

50. ročník matematické olympiády na středních školách

Leo Boček (editor); Karel Horák (editor); Tomáš Pitner (editor); Jaromír Šimša (editor); Jaroslav Švrček (editor); Pavel Töpfer (editor): 50. ročník matematické olympiády na středních školách. Zpráva o řešení úloh ze soutěže konané ve školním roce 2000/2001. 42. mezinárodní matematická olympiáda. 13. mezinárodní olympiáda v informatice. (Czech). Praha: Jednota českých matematiků a fyziků, 2001.

Persistent URL: <http://dml.cz/dmlcz/405023>

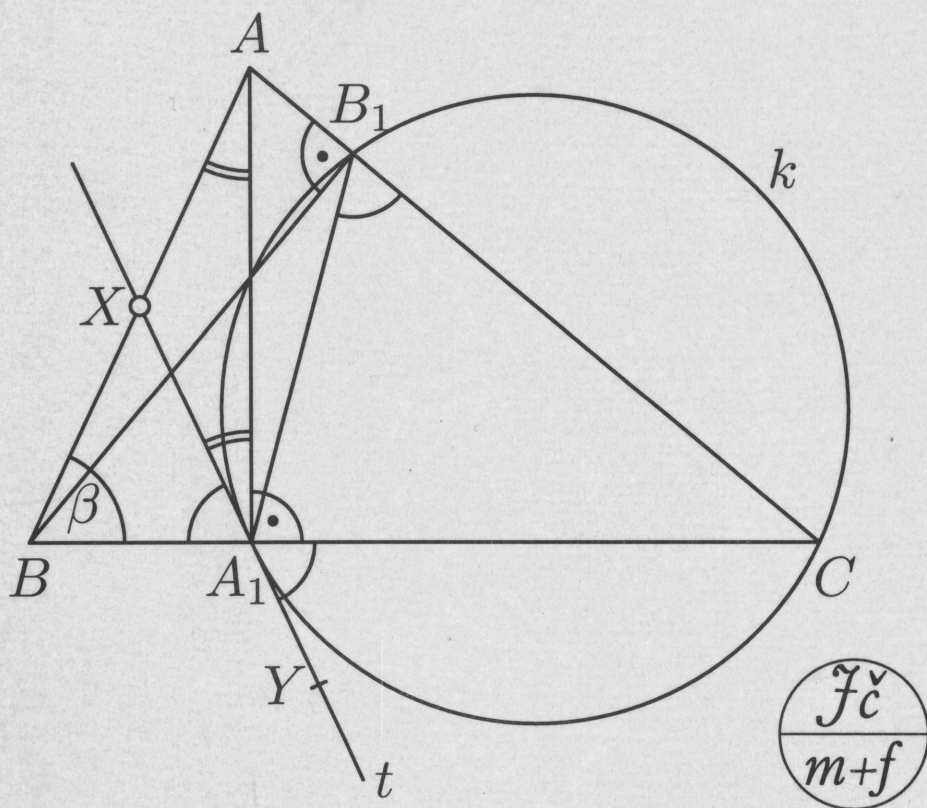
Terms of use:

Institute of Mathematics of the Czech Academy of Sciences provides access to digitized documents strictly for personal use. Each copy of any part of this document must contain these *Terms of use*.



This document has been digitized, optimized for electronic delivery and stamped with digital signature within the project *DML-CZ: The Czech Digital Mathematics Library* <http://dml.cz>

50. ročník matematické olympiády na středních školách



PADESÁTÝ ROČNÍK MATEMATICKÉ OLYMPIÁDY NA STŘEDNÍCH ŠKOLÁCH

Zpráva o řešení úloh ze soutěže konané ve školním roce 2000/2001

42. MEZINÁRODNÍ MATEMATICKÁ OLYMPIÁDA
TŘINÁCTÁ MEZINÁRODNÍ OLYMPIÁDA V INFORMATICE

JEDNOTA ČESKÝCH MATEMATIKŮ A FYZIKŮ PRAHA

Obsah

O průběhu 50. ročníku matematické olympiády	5
Výsledky celostátního kola 50. ročníku MO kategorie A	10
Výsledky celostátního kola 50. ročníku MO kategorie P	11
Nejúspěšnější řešitelé II. kola MO v kategoriích A, B, C a P	12
Kategorie C	25
Texty úloh	25
Řešení úloh	28
Kategorie B	37
Texty úloh	37
Řešení úloh	40
Kategorie A	51
Texty úloh	51
Řešení úloh	56
Kategorie P	83
Texty úloh	83
Řešení úloh	101
Přípravná soustředění před 42. MMO	137
Úlohy zadáné na přípravném soustředění	138
Mezinárodní střetnutí česko-slovensko-polské	141
Texty soutěžních úloh	142
Řešení úloh	143
42. mezinárodní matematická olympiáda	150
Texty soutěžních úloh	153
Řešení úloh	154
Osmý ročník Středoevropské olympiády v informatice	163
13. mezinárodní olympiáda v informatice	165
Texty soutěžních úloh	167

Za přispění spolupracovníků zpracovali
doc. RNDr. Leo Boček, CSc., RNDr. Karel Horák, CSc.,
RNDr. Tomáš Pitner, Dr., doc. RNDr. Jaromír Šimša, CSc.,
RNDr. Jaroslav Švrček, CSc., doc. RNDr. Pavel Töpfer, CSc.

© Karel Horák, editor, 2001

ISBN 80-7015-828-X

O průběhu 50. ročníku matematické olympiády

Jubilejní — 50. ročník matematické olympiády měl v podstatě stejný průběh jako ty bezprostředně předcházející ročníky. I ten 50. začal už s předstihem výběrem všech úloh, který zajišťují tři úlohové komise, jedna pro kategorie A, B, C (střední školy), druhá pro kategorie Z5–Z9 (základní školy a nižší třídy víceletých gymnázií), třetí pro kategorii P (programování). Následuje příprava letáků s úlohami I. kola a komentářů k nim. Komentáře, to jsou texty pro učitele, které obsahují nejen řešení všech úloh I. kola, ale též návodné úlohy, kterými může učitel žákům pomoci najít řešení soutěžních úloh MO. Není možné zde uvést všechny pracovníky, kteří se podíleli na přípravě 50. ročníku MO. Uvedme pouze důležitou informaci, že úlohy připravujeme společně se slovenskými kolegy, tedy „federálně“. Z české strany se na přípravě nejvíce podíleli doc. dr. J. Šimša, CSc., dr. K. Horák, CSc., doc. dr. P. Töpfer, CSc., dr. L. Hozová, doc. dr. M. Volfová, CSc., Mgr. P. Leischner, Mgr. M. Krejčová, doc. dr. L. Boček, CSc., a další.

Pořadateli soutěže Matematická olympiáda jsou Ministerstvo školství, mládeže a tělovýchovy ČR, Jednota českých matematiků a fyziků a Matematický ústav Akademie věd ČR. Soutěž řídí Ústřední výbor MO, který je jmenován na návrh JČMF Ministerstvem školství, mládeže a tělovýchovy. Ve školním roce 2000–2001 řešil ÚV MO hlavně přípravu termínů příštího ročníku MO, přípravu soustředění úspěšných řešitelů a přípravu celostátních kol. V této věci nám velmi pomohla Střední průmyslová škola sdělovací techniky v Praze, která se ujala organizace III. kol MO kategorií A a P, a dále Fakulta tělesné výchovy a sportu, která nám pronajala prostory pro vlastní soutěž i ubytování soutěžících. Delší zasedání ÚV MO se konalo 4. dubna 2001 při zakončení celostátního kola MO kategorie A. Jeho první část měla slavnostní ráz, konala se v Karolinu. Členy ÚV MO, hosty i soutěžící přivítal rektor Univerzity Karlovy prof. ing. Ivan Wilhelm, CSc. Další projevy přednesli ředitel odboru MŠMT dr. Karel Tomek, ředitel MÚ AV doc. dr. Karel Segeth, CSc., předseda JČMF prof. dr. Jaroslav Kurzweil, DrSc., děkan MFF UK prof. dr. Ivan Netuka, DrSc., ředitelka SPŠST ing. M. Plocková a další

hosté. Matematický ústav AV věnoval každému účastníku celostátního kola MO kategorie A knihu S. Singha „Velká Fermatova věta“, soutěžící v kategorii P odměnil knihami zase Ústav informatiky AV ČR. Zástupce MŠMT dr. Tomek předal děkovný dopis pana ministra řadě učitelů a dalších pracovníků, kteří se dlouhodobě věnovali práci v MO. Další pracovníci obdrželi děkovný dopis od ÚV JČMF. Předseda ÚV MO poděkoval oběma ústavům Akademie věd za knižní dary pro soutěžící, a Střední průmyslové škole sdělovací techniky za práci při zajištění celostátních kol MO. Matematicko-fyzikální fakultě poděkoval za vydání brožurky „Padesát let Matematické olympiády“, kterou fakulta věnovala všem soutěžícím i pracovníkům v MO. Dík patří i všem sponzorům, kterými byly ČEZ, Společnost SuSe, Česká pojišťovna, Český rozhlas, Precheza Přerov, Česká vědecko-technická společnost a Společnost O. Borůvky. Posledně jmenovaná věnovala Matematické olympiádě vedle knižních darů i podstatnou finanční částku, která nám umožnila vydat tuto ročenku. První tři soutěžící v každé kategorii obdrželi odměnu od MŠMT ve formě knižních poukázek, všechny ostatní ceny věnovali úspěšným řešitelům sponzoři. Hlavním bodem druhé části zasedání ÚV MO bylo vypracování návrhu na složení ÚV MO pro další čtyřleté funkční období. Vypracovaný návrh byl schválen ÚV JČMF i MŠMT, takže od 51. ročníku MO je předsedou ÚV MO doc. dr. J. Šimša, CSc., místopředsedy dr. J. Švrček, CSc., doc. dr. P. Tlustý, CSc., a doc. dr. P. Töpfer, CSc., jednatelem dr. K. Horák, CSc. Byli též jmenováni předsedové krajských výborů MO podle nového státoprávního uspořádání.

Celostátními koly MO soutěž v příslušném roce vlastně nekončí. Družstva České republiky se pak ještě účastní mezinárodních soutěží v matematice a informatice. O výsledcích našich účastníků se dočtete v dalších částech této brožurky.

Příprava každého ročníku MO je úkolem ÚV MO a úlohových komisí. Matematická olympiáda by však nemohla existovat, kdyby se jí nevěnovali učitelky a učitelé, profesorky a profesori na základních a středních školách, kteří nad své pracovní povinnosti a bez nároku na odměnu vedou své žáky k účasti v MO, pomáhají jim překonat první potíže při řešení úloh I. kola, opravují úlohy. ÚV MO jim upřímně děkuje za jejich práci pro MO.

Dosavadní ÚV MO skončil 50. ročníkem MO své funkční období. Byl jmenován v roce 1997 na čtyřleté funkční období ve složení:

Předseda:

Doc. RNDr. Leo Boček, CSc., MFF UK Praha

Místopředsedové:

Doc. RNDr. Jaromír Šimša, CSc., Matematický ústav AV ČR Brno

Doc. RNDr. Václav Sedláček, CSc., Fakulta informatiky MU Brno

RNDr. Jiří Binder, CSc., Pedagogická fakulta UK Praha

Tajemníci:

RNDr. Karel Horák, CSc., Matematický ústav AV ČR, Praha

Doc. RNDr. Pavel Töpfer, CSc., MFF UK Praha

Mgr. Monika Barešová, Dr., Matematický ústav AV ČR Praha

Zástupce MŠMT ČR: Mgr. Vlasta Marková

Další členové:

Prof. RNDr. Miroslav Fiedler, DrSc., MÚ AV ČR Praha

PhDr. Libuše Hozová, Matematický ústav Slezské univerzity Opava

RNDr. Dag Hrubý, Gymnázium Jevíčko

Prof. RNDr. Milan Koman, CSc., Pedagogická fakulta UK Praha

Doc. RNDr. Jan Kratochvíl, CSc., MFF UK Praha

Mgr. Marie Krejčová, Základní škola Jihlava

Mgr. Pavel Leischner, Pedagogická fakulta Jihočeské univerzity České
Budějovice

RNDr. Jaroslav Švrček, CSc., Přírodovědecká fakulta UP Olomouc

Doc. PhDr. Marta Volfová, CSc., Pedagogická fakulta Univerzity Hradec
Králové

RNDr. Antonín Vrba, CSc., Pedagogická fakulta UK Praha

RNDr. Jaroslav Zhouf, Gymnázium Praha

Předsedové regionálních výborů MO:

PhDr. Ivan Bušek, Pedagogické centrum Praha

RNDr. Šárka Gergelitsová, Gymnázium Benešov

Doc. RNDr. Pavel Tlustý, CSc., Pedagogická fakulta Jihočeské univerzity
České Budějovice

RNDr. Jiří Potůček, CSc., Pedagog. fakulta Západočeské univerzity Plzeň

Mgr. Pavla Hofmanová, Pedagogická fakulta UJEP Ústí n. L.

Prof. RNDr. Bohdan Zelinka, DrSc., Pedagogická fakulta Technické uni-
verzity Liberec

RNDr. Jan Žížka, Gymnázium Přelouč

Mgr. Jan Beneš, Gymnázium Jihlava

RNDr. Jiří Herman, Dr., Gymnázium Brno

RNDr. Jan Chudárek, Gymnázium Zlín

Doc. RNDr. Vladimír Vlček, CSc., Přírodovědecká fakulta Olomouc

V průběhu funkčního období požádali o uvolnění z funkce člena ÚV MO v souvislosti s přechodem na jiné zaměstnání doc. Sedláček a dr. Binder, jejich povinnosti převzali tajemníci doc. Töpfer a dr. Barešová. Se změnami na MŠMT vystřídal Mgr. Markovou dr. Václav Müller.

Přejeme novému ÚV MO hodně úspěchů a matematické olympiádě mnoho úspěšných účastníků po celou dobu aspoň padesáti dalších ročníků.

Tabulka 1

Počty žáků středních škol soutěžících v I. kole 50. ročníku MO

Oblast	Kategorie								Celkem	
	A		B		C		P			
	S	U	S	U	S	U	S	U	S	U
Praha	65	37	54	14	82	57	19	19	220	127
Střední Čechy	104	33	91	39	111	54	26	25	332	151
Jižní Čechy	55	28	61	27	97	45			213	100
Západní Čechy	35	28	39	22	52	33	6	6	132	89
Liberec	22	7	36	20	24	11	2	2	84	40
Ústí n. Labem	4	3	31	27	42	29	2	2	79	61
Východní Čechy	39	26	26	18	89	62			154	106
Brno	101	50	95	38	173	122			369	210
Jihlava	42	33	27	18	52	35	10	7	131	93
Zlín	76	31	33	10	46	20	5	5	160	66
Severní Morava	167	45	197	32	267	65	26	26	657	168
ČR	710	321	690	265	1 035	533	96	92	2 531	1 211

Tabulka 2

Počty žáků středních škol soutěžících v II. kole 50. ročníku MO

Oblast	Kategorie								Celkem	
	A		B		C		P			
	S	U	S	U	S	U	S	U	S	U
Praha	36	22	11	3	54	38	19	12	120	75
Střední Čechy	34	4	34	18	48	16	25	9	141	47
Jižní Čechy	22	2	22	10	40	18			84	30
Západní Čechy	28	8	20	14	31	17	6	1	85	40
Liberec	8	4	7	3	23	7	4	0	42	14
Ústí n. Labem	7	2	17	6	11	4	2	2	37	14
Východní Čechy	24	9	17	5	52	24			93	38
Brno	45	10	35	9	110	55			190	74
Jihlava	31	7	15	2	33	9	6	0	85	18
Zlín	27	2	11	7	19	15	5	3	62	27
Severní Morava	45	13	32	6	65	32	26	3	168	54
ČR	307	83	221	83	486	235	93	30	1 107	431

S ... počet všech soutěžících

U ... počet úspěšných řešitelů

Výsledky celostátního kola 50. ročníku MO
kategorie A

Vítězové

1.-2.	<i>Jan Herman</i> , 4/4, G tř. Kpt. Jaroše 14, Brno	41 b.
	<i>Martin Tancer</i> , 3/4, G Ch. Dopplera, Praha 5	41 b.
3.	<i>Josef Křišťan</i> , 7/7, G Mikulášské nám. 23, Plzeň	39 b.
4.-5.	<i>Tomáš Protivínský</i> , 3/4, G tř. Kpt. Jaroše 14, Brno	36 b.
	<i>Ondřej Suchý</i> , 7/7, G Mikulášské nám. 23, Plzeň	36 b.
6.-7.	<i>Martin Káldy</i> , 2/4, G Ch. Dopplera, Praha 5	33 b.
	<i>Marek Sulovský</i> , 4/4, G tř. Kpt. Jaroše 14, Brno	33 b.
8.	<i>Jan Kynčl</i> , 6/6, G Kostelní 259, Jilemnice	31 b.
9.	<i>Jiří Koula</i> , 4/4, G U Libeňského zámku 1, Praha 8	29 b.
10.-11.	<i>Pavel Čížek</i> , 2/4, G a OA, Kralupy n. Vlt.	28 b.
	<i>Jaroslav Hájek</i> , 3/4, GMK, Bílovec	28 b.

Další úspěšní řešitelé

12.-13.	<i>Marek Krčál</i> , 2/4, G tř. Kpt. Jaroše 14, Brno	27 b.
	<i>Ondřej Kreml</i> , 4/4, GMK, Bílovec	27 b.
14.	<i>Ondřej Kurka</i> , 7/8, G Ch. Dopplera, Praha 5	26 b.
15.	<i>Václav Flaška</i> , 8/8 Svob. cheb. škola, Cheb	25 b.
16.-17.	<i>Pavel Kůs</i> , 4/4, G Ch. Dopplera, Praha 5	24 b.
	<i>Rudolf Stolař</i> , 4/4, G tř. Kpt. Jaroše 14, Brno	24 b.
18.-19.	<i>Petr Jelínek</i> , 8/8, G Parlérova 2/118, Praha 6	23 b.
	<i>David Šálek</i> , 6/6, G Na Vítězné pláni 1160, Praha 4	23 b.
20.-21.	<i>Tomáš Hanzák</i> , 3/4, G nám. Dr. E. Beneše 1573, Kladno	22 b.
	<i>Martin Sikora</i> , 4/4, GMK, Bílovec	22 b.
22.-26.	<i>Petr Götz</i> , 4/4, GMK, Bílovec	20 b.
	<i>Martin Holík</i> , 4/4, GMK, Bílovec	20 b.
	<i>Martin Motl</i> , 4/4, GMK, Bílovec	20 b.
	<i>Ondřej Šerý</i> , 8/8, G a sport. škola, Kladno	20 b.
	<i>Pavel Vališ</i> , 4/4, G a OA, Kralupy n. Vlt.	20 b.

Výsledky celostátního kola 50. ročníku MO
kategorie P

Vítězové

1. <i>Pavel Čížek</i> , 6/8, Dvořákovo G a OA, Kralupy n. Vlt.	33 b.
2. <i>Miloslav Trmač</i> , 8/8, Biskupské G, Brno	31 b.
3. <i>Roman Krejčík</i> , 4/4, G Ch. Dopplera, Praha 5	29 b.
4. <i>Marek Sulovský</i> , 7/7, G Brno, tř. Kpt. Jaroše	28 b.
5. <i>Eva Ondráčková</i> , 7/7, G Tišnov	25 b.
6. <i>Jiří Danihelka</i> , 2/4, SPŠ Čapkova, Písek	22 b.
7. <i>Jiří Svoboda</i> , 7/8, G Ch. Dopplera, Praha 5	20 b.
8. <i>Jiří Fink</i> , 4/4, SPŠE Ječná, Praha 2	19 b.

Další úspěšní řešitelé

9. <i>Petr Matas</i> , 7/7, G Klatovy, Národních mučedníků	17 b.
10.–11. <i>Martin Beránek</i> , 7/7, G Praha 4, Ohradní	16 b.
<i>Robert Poch</i> , 4/4, SPŠST Panská, Praha 1	16 b.
12. <i>Jiří Koula</i> , 4/4, G Praha 8, U Libeňského zámku	15 b.
13.–16. <i>Josef Cibulka</i> , 3/4, Akademické G, Praha 1, Štěpánská	13 b.
<i>Martin Hamrle</i> , 3/4, G Pelhřimov, Jirsíkova	13 b.
<i>Jiří Štěpánek</i> , 5/8, G Brno, tř. Kpt. Jaroše	13 b.
<i>Ondřej Zajíček</i> , 4/4, SPŠS Chrudim	13 b.

Nejúspěšnější řešitelé II. kola MO
v kategoriích A, B, C a P

Z každého kraje a z každé kategorie je uvedeno nejvýše prvních deset řešitelů. Označení G znamená gymnázium, M, resp. MF zaměření studijního oboru 01 Matematika, resp. 02 Matematika a fyzika.

—— Praha ——

Kategorie A

- 1.-2. *František Havlík*, 7.E, G Sladkovského nám., Praha 3
Ondřej Kurka, 7.M, GChD, Zborovská, Praha 5
3. *David Šálek*, 6.C, G Na Vít. pláni, Praha 4
- 4.-7. *Lenka Havrdová*, VII, G Písnická, Praha 4
Petr Jelínek, VIII, GJK, Parlářova, Praha 6
Martin Káldy, 2.C, GChD, Zborovská, Praha 5
Václav Vlasák, 6.A, Arcibiskup. G Korunní, Praha 2
- 8.-9. *David Chodounský*, 4.C, GChD, Zborovská, Praha 5
Martin Tancer, 3.C, GChD, Zborovská, Praha 5
- 10.-12. *Martin Klimeš*, 3.A, G Botičská, Praha 2
Jiří Koula, 4.E, G U Libeň. zámku, Praha 8
Pavel Kůs, 4.C, GChD, Zborovská, Praha 5

Kategorie B

1. *Martin Káldy*, 2.C, GChD, Zborovská, Praha 5
2. *Ondřej Čertík*, 6.M, GChD, Zborovská, Praha 5
3. *Jan Kadlec*, 2.C, GChD, Zborovská, Praha 5

Kategorie C

1. *Alexandr Kazda*, 5.A, G Nad Alejí, Praha 6
- 2.–4. *Václav Potoček*, 1.M, SPŠST Panská, Praha 1
Daniel Balaš, 1.E, G Arabská, Praha 6
Petr Škoda, 5.B, G Ústavní, Praha 8
- 5.–7. *Michal Rameš*, 1.L, SPŠST Panská, Praha 1
Alena Drábková, 3.A, G Na Vítězné pláni, Praha 4
David Navara, 1.C, G Zborovská, Praha 5
8. *Martin Pastrňák*, 1.M, SPŠST Panská, Praha 1
- 9.–13. *Lenka Homolková*, V.B, Akad. G Korunní, Praha 2
Mariana Svobodová, V.B, Akad. G Korunní, Praha 2
Jiří Kukačka, T.A, G Postupická, Praha 4
Vojtěch Křesala, V.B, G Parlérova, Praha 6
Štefan Alakša, V.B, G Parlérova, Praha 6

Kategorie P

1. *Jiří Svoboda*, 7.M, GChD, Zborovská, Praha 5
- 2.–3. *Martin Beránek*, VII, G Ohradní, Praha 4
- 2.–3. *Robert Poch*, 4.K, SPŠST Panská 3, Praha 1
4. *Jiří Koula*, 4.E, G U Libeňského zámku, Praha 8
5. *Jiří Fink*, P4a, SPŠE Ječná, Praha 2
6. *Roman Krejčík*, 4.C, GChD, Zborovská, Praha 5
7. *Josef Cibulka*, 3.B, Akademické G, Štěpánská, Praha 1
8. *Dan Lessner*, 2/4, G Bernarda Bolzana, V Holešovičkách, Praha
9. *František Havlůj*, 7/7, G Sladkovského nám., Praha 3
10. *Martin Hinner*, 4.G, G Arabská, Praha

———— Střední Čechy ————

Kategorie A

1. *Ondřej Šerý*, 8.C (8), G a sport. škola Kladno
2. *Tomáš Hanzák*, 3.B (4), G Kladno
3. *Pavel Vališ*, 4.A (4), G Kralupy
- 4.–5. *Pavel Čížek*, 6. (8), G Kralupy
Irena Váňová, 4.G (4), G Sedlčany

- 6.-8. *Tomáš Petr*, 7.A (7), G Benešov
Martin Doubek, 3.A (4), G Kladno
Oto Havle, 4.A (4), G Kladno
9. *Ondřej Chochola*, 3.A (4), G Kladno

Kategorie B

1. *Jan Lamač*, G Mnichovo Hradiště
2. *Pavel Čížek*, G Kralupy
- 3.-4. *Jiří Paleček*, G Kladno
Pavel Brom, G Mladá Boleslav
5. *Martina Vermachová*, G a sport. škola Kladno
6. *Marek Šmíd*, G Brandýs

Kategorie C

1. *Jakub Vilhelm*, G Beroun
2. *Jan Kotera*, G Kralupy
3. *Ondřej Šindelář*, G Kralupy
- 4.-5. *Eva Patáková*, G Dobříš
Petr Bílý, G Slaný
6. *Jan Mejšnar*, G Mladá Boleslav
- 7.-9. *Barbora Řezáčová*, G Beroun
Jan Váňa, G Kutná Hora
Lenka Prchalová, G Kutná Hora
- 10.-15. *Jaroslav Kuneš*, G Benešov
Miroslav Frantes, G Benešov
Michal Kašpar, G Kolín
Stanislava Strachová, G Poděbrady
Kateřina Průchová, G Příbram
Jan Procházka, GJP, Mladá Boleslav

Kategorie P

1. *Pavel Čížek*, 6.G, Dvořákovo G a OA, Benešov
- 2.-4. *Jiří Šofka*, VII, GOA Benešov
Zdeněk Bulan, VII.A, G Benešov, Husova
Jakub Maršík, VII.B, G Benešov, Husova
5. *Pavel Šafrata*, VII.B, G Benešov, Wagnerovo nám.

—— Jižní Čechy ——

Kategorie A

1. *Jan Hejda*, G České Budějovice, Jírovcova
- 2.–3. *Tomáš Dolejšek*, G České Budějovice, Jírovcova
Radek Mlada, G Pelhřimov
- 4.–5. *Richard Chudoba*, G České Budějovice, Jírovcova
Dana Bartošová, G J. Hradec

Kategorie B

1. *Jan Kouba*, G Č. Krumlov
2. *Jiří Danielka*, SPŠ Písek
3. *Milan Straka*, G Strakonice

Kategorie C

- 1.–2. *Radek Mlada*, G Pelhřimov
Martin Hadrava, G Soběslav
3. *Ondřej Prašnička*, G České Budějovice, Jírovcova
- 4.–5. *Karel Princ*, G České Budějovice, Jírovcova
Jan Šmrha, G Pelhřimov
- 6.–8. *Jiří Bernas*, SPŠ Písek
Barbora Hošková, G České Budějovice, Jírovcova
Zdeněk Tichý, G Pelhřimov
- 9.–11. *Martin Pilát*, G České Budějovice, Česká
Miroslav Janíček, G Tábor
Jakub Rada, G České Budějovice, Česká

Kategorie P

1. *Martin Hamrle*, G Pelhřimov
2. *Jiří Danihelka*, A2S, SPŠ Písek
3. *Tomáš Lusk*, 4.B, G České Budějovice, Jírovcova

Kategorie A

1. *Martin Setvín*, septima M, 1. G Plzeň
2. *Ondřej Suchý*, septima M, 1. G Plzeň
- 3.–4. *Václav Flaška*, oktáva, Svob. chebská škola, Cheb
Josef Křišťan, septima M, 1. G Plzeň
5. *Martin Hrba*, septima M, G Sušice
6. *David Mareček*, sexta M, 1. G Plzeň
- 7.–8. *Jiří Ajgl*, sexta M, 1. G Plzeň
Pavel Křeček, septima M, 1. G Plzeň

Kategorie B

1. *Klára Šteklová*, sexta, 1. G Plzeň
2. *Josef Mládek*, sexta, 1. G Plzeň
3. *Barbora Vostracká*, sexta, 1. G Plzeň
- 4.–6. *Luboš Matásek*, sexta, 1. G Plzeň
Michal Bareš, sexta, 1. G Plzeň
Jiří Ajgl, sexta, 1. G Plzeň
- 7.–8. *Jiří Reitspies*, sexta, 1. G Plzeň
Jaroslav Svoboda, sexta, 1. G Plzeň
9. *Ladislav Lenc*, sexta, 1. G Plzeň
- 10.–11. *David Mareček*, sexta, 1. G Plzeň
Michal Kvíz, sexta, 1. G Plzeň

Kategorie C

- 1.–2. *Petr Cvachovec*, kvinta, 1. G Plzeň
Petr Havránek, kvinta, 1. G Plzeň
3. *Petra Flajtingrová*, kvinta, Masarykovo G, Plzeň
4. *Jaroslav Nunvář*, 1.r., G Ostrov n. Ohří
- 5.–6. *Jan Bartošek*, kvinta, 1. G Plzeň
Jari Fazekas, 1.r., ISŠTE Sokolov
7. *Zdeňka Fartáková*, kvinta, G Sušice
- 8.–11. *Jiří Kohout*, kvinta, 1. G Plzeň
Tereza Naarová, kvinta, 1. G Plzeň
Ludmila Schwarzová, kvinta, 1. G Plzeň

Kategorie P

1. *Petr Matas*, septima B, G Klatovy

—— **Liberec** ——

Kategorie A

1. *Martin Vitouš*, 7/8, G Liberec, Partyzánská

Kategorie B

1. *Karel Pupík*, sexta B, GaSG, Jilemnice

Kategorie C

1. *Marek Kříklán*, 1B, GFXŠ, Liberec
2. *Jana Divišová*, kvinta, G Mimoň
3. *Michal Kollert*, kvinta A, GFXŠ, Liberec
4. *Jan Breuer*, A1, SPŠSE, Liberec
5. *Martin Svoboda*, kvinta A, GFXŠ, Liberec
6. *Lenka Šíkolová*, kvinta A, GaSG, Jilemnice
7. *Miroslav Klož*, kvinta B, GFXŠ, Liberec
8. *Jiří Schejbal*, kvinta A, G Turnov
9. *Jan Jebavý*, kvinta B, G Turnov
10. *Veronika Zrníková*, kvinta B, GFXŠ, Liberec

—— **Ústí nad Labem** ——

Kategorie A

1. *Miroslav Šulc*, sexta, G Ústí nad Labem, Stavbařů
2. *Jan Ondřích*, 3.E, SPŠ Ústí nad Labem
3. *Jiří Binder*, V7A, G Ústí nad Labem, Jateční
4. *Lucie Plochová*, 7F, G Teplice, Čs. Dobrovolců

Kategorie B

- 1.-2. *Nelly Vostrá*, kvinta C, G Děčín, Komenského nám.
Vít Šípál, 6.B, G Ústí nad Labem, Jateční
3. *Tomáš Kozelek*, sexta, G Kadaň

Kategorie C

1. *Jaromír Pufler*, G Teplice, Čs. Dobrovolců
2. *Jan Veselý*, G Teplice, Čs. Dobrovolců
3. *Martin Suchan*, G V. Hlavatého, Louny
- 4.-5. *Lenka Sarnová*, G Most
Jakub Strnad, SPŠ Chomutov
- 6.-9. *Petr Thür*, G Chomutov
Jan Rusňák, G Chomutov
Miroslav Řehoř, G Teplice, Čs. Dobrovolců
Marek Hilovský, G Děčín, Komenského nám.
- 10.-11. *Marek Bartes*, G Teplice, Čs. Dobrovolců
Jana Ferencová, G Ústí nad Labem, Stavbařů

Kategorie P

1. *Bohumír Kubík*, 3.C, G Rumburk

—— Východní Čechy ——

Kategorie A

1. *Miroslav Hejna*, G Rychnov n. K.
2. *Ladislav Benda*, GJKT, Hradec Králové
- 3.-4. *Jan Hošek*, GJKT, Hradec Králové
Jan Prachař, G Rychnov n. K.
- 5.-7. *Václav Jára*, GJKT, Hradec Králové
Jan Nožka, G Lanškroun
Jiří Eliášek, G Trutnov
- 8.-9. *Miloš Doubek*, GJKT, Hradec Králové
Martin Kabrhel, G Litomyšl

Kategorie B

1. *Petr Pošta*, G Pardubice, Dašická
- 2.–3. *Jan Prachař*, G Rychnov n. K.
Dana Chromíková, G Pardubice, Dašická
- 4.–5. *Miroslav Hejna*, G Rychnov n. K.
Petr Rezek, G Litomyšl

Kategorie C

1. *Jan Moláček*, GJKT, Hradec Králové
- 2.–3. *Romana Kubátová*, GJKT, Hradec Králové
Jan Ondruš, G Rychnov n. K.
4. *Jan Zelený*, G Náchod
5. *Jindřich Šedek*, G Náchod
6. *Petr Voltr*, GJKT, Hradec Králové
7. *Pavel Kladivo*, G Polička
8. *Martin Holeček*, G Náchod
- 9.–10. *Jan Langr*, G Jaroměř
Eva Jílková, G B. Němcové, Hradec Králové

Kategorie P

1. *Ondřej Zajíček*, 4/4, SPŠS Chrudim
2. *Roman Hocke*, 6/6, G Pardubice, Dašická
3. *Jakub Ďurovec*, 6/6, G Pardubice, Dašická

———— **Brno** ————

Kategorie A

- 1.–3. *Marek Krčál*, 2.A M, G Brno, tř. Kpt. Jaroše
Eva Ondráčková, septima A/7, G Tišnov
Tomáš Protivínský, 3.A M, G Brno, tř. Kpt. Jaroše
- 4.–5. *Jan Herman*, 4.A M, G Brno, tř. Kpt. Jaroše
Marek Sulovský, 4.A M, G Brno, tř. Kpt. Jaroše
6. *Rudolf Stolař*, 4.A M, G Brno, tř. Kpt. Jaroše
- 7.–8. *Miroslav Frost*, sextima A/7, G Brno, Elgartova
Jan Trojek, 4.C, G Břeclav

- 9.-10. *Jakub Chaloupka*, septima A/7, G Brno, Křenová
Vítězslav Kala, 1.A M, G Brno, tř. Kpt. Jaroše

Kategorie B

1. *Jan Hladký*, 2.A M, G Brno, tř. Kpt. Jaroše
2.-3. *Veronika Trnková*, 2.A M, G Brno, tř. Kpt. Jaroše
Jiří Vlácil, 2.A M, G Brno, tř. Kpt. Jaroše
4.-5. *Jan Orava*, sexta A, G Kyjov
Jan Zahradník, 2.A M, G Brno, tř. Kpt. Jaroše
6.-7. *Robert Ganian*, sexta B, G Brno-Komín
Antonín Pavelka, 2.A M, G Brno, tř. Kpt. Jaroše
8. *Helena Grulichová*, sexta B, G Břeclav
9. *Alena Šimečková*, sexta A, G Kyjov

Kategorie C

- 1.-8. *Alena Drášilová*, 1.A M, G Brno, tř. Kpt. Jaroše
Sven Dražan, 1.A M, G Brno, tř. Kpt. Jaroše
Tomáš Gregorovič, kvinta, G Šlapanice
Tomáš Hebelka, kvinta, G Brno, Vídeňská
Petr Chalupa, 1.A M, G Brno, tř. Kpt. Jaroše
Jan Novotný, 1.A M, G Brno, tř. Kpt. Jaroše
Alena Robotková, 1.A M, G Brno, tř. Kpt. Jaroše
Michal Rychnovský, 4.ag M, G Brno, tř. Kpt. Jaroše
9. *Pavel Troubil*, 1.A M, G Brno, tř. Kpt. Jaroše
10.-13. *Jiří Hutárek*, 4.ag M, G Brno, tř. Kpt. Jaroše
Vítězslav Kala, 1.A M, G Brno, tř. Kpt. Jaroše
Jan Křetínský, kvinta B, G Brno, Žižkova
Martin Vejnar, 4.ag M, G Brno, tř. Kpt. Jaroše

Kategorie P

1. *Miloslav Trmač*, oktáva B, Biskupské G Brno
2. *Marek Sulovský*, 4.A, G Brno, tř. Kpt. Jaroše
3. *Roman Margold*, 4.A, G Brno, tř. Kpt. Jaroše
4.-6. *Eva Ondráčková*, septima A, G Tišnov
Jiří Štěpánek, 1.A, G Brno, tř. Kpt. Jaroše
Pavel Troubil, 1.A, G Brno, tř. Kpt. Jaroše

7. *Jan Herman*, 4.A, G Brno, tř. Kpt. Jaroše
8. *Martin Vejnar*, 4.ag, G Brno, tř. Kpt. Jaroše
9. *Jan Franců*, 4.A, G Brno, tř. Kpt. Jaroše

—— Jihlava ——

Kategorie A

1. *Zdeněk Černý*, 4.A, G Žďár nad Sázavou
- 2.–3. *Pavel Janda*, septima, G Telč
Martin Koláček, 4.C, G Třebíč
4. *Jaromír Šír*, 3.C, G Nové Město na M.
- 5.–7. *Jiří Lipovský*, sexta, G Bystřice nad Pern.
Josef Novák, V7A, G Velké Meziříčí
Leoš Přikryl, 4.C, G Jihlava

Kategorie B

- 1.–2. *Martin Hána*, sexta, G Žďár nad Sázavou
Jiří Lipovský, sexta, G Bystřice n. P.

Kategorie C

1. *Josef Janák*, V5.A, G Velké Meziříčí
2. *Michal Holub*, 1.B, G Třebíč
3. *Dominik Macáš*, kvinta, G Bystřice nad Pern.
- 4.–5. *Michal Janků*, 1.D, G Znojmo
Aleš Povalač, 1.B, G Třebíč
- 6.–7. *Luděk Gregor*, 1.B, G Nové Město na M.
Karel Veselý, 1.D, G Znojmo
- 8.–9. *Jan Dočekal*, 1.D, G Znojmo
Petra Mácová, kvinta D, G Telč

Kategorie P

1. *Jaroslav Tykal*, 4.C, G Jihlava

Kategorie A

1. *Pavel Kočica*, G Uherský Brod
2. *Radek Vitovják*, SPŠ Zlín

Kategorie B

1. *Radovan Polanský*, GJAK, Uherský Brod
2. *Pavel Česka*, SPŠ Zlín
3. *Radek Krejčířík*, G Uherské Hradiště
4. *Kamil Langer*, G Otrokovice
- 5.-6. *Ondřej Kunčar*, G Kroměříž
Jan Vavrys, GJAK, Uherský Brod
7. *Martin Cetkovský*, G Zlín, Lesní čtvrť

Kategorie C

- 1.-5. *Kristína Čapková*, GJAK, Uherský Brod
Martin Dungl, G Kroměříž
František Konopecký, G Holešov
Adam Kosík, G Uherské Hradiště
Milan Pečeňa, G Zlín, Lesní čtvrť
- 6.-7. *Přemysl Habáň*, G Uherské Hradiště
Jan Olšina, G Kroměříž
8. *Stanislav Sigmund*, G Kroměříž
9. *Michal Hurbiš*, GJAK, Uherský Brod
- 10.-15. *Milan Cellar*, G Hodonín
Martin Horáček, G Zlín, Lesní čtvrť
Monika Hradilová, G Holešov
Michal Krejčí, GJAK, Uherský Brod
Petr Kudela, G Zlín TGM
Vít Kutálek, G Uherské Hradiště

Kategorie A

1. *Jaroslav Hájek*, 3.C, GMK, Bílovec
- 2.–5. *Ondřej Kreml*, 4.C, GMK, Bílovec
Patrik Hudec, 4.C, GMK, Bílovec
Martin Holík, 4.C, GMK, Bílovec
Nartin Sikora, 4.C, GMK, Bílovec
- 6.–7. *Petr Götz*, 4.C, GMK, Bílovec
Martin Motl, 4.D, GMK, Bílovec
- 8.–9. *Lucia Jarešová*, 3.A, GJŠ, Přerov
Marek Klus, 4.A, pol. G, Český Těšín
10. *Minh Ha*, 4.C, GMK, Bílovec

Kategorie B

1. *Václav Cviček*, 4.A, GPB, Frýdek-Místek
2. *Pavel Ludvík*, 2.C, GMK, Bílovec
3. *Tomáš Staněk*, VI.A, G Ostrava, Volgogradská
- 4.–5. *Marek Schmidt*, VI.D, G Karviná
Michal Straka, 6.A, GJW, Prostějov
6. *Václav Mácha*, 2.C, GMK, Bílovec

Kategorie C

1. *Jana Hrudíková*, V.C, GJŠ, Přerov
2. *Tomáš Gavenčiak*, 1.C, GMK, Bílovec
- 3.–5. *Jan Matoušek*, 5.C, GJW, Prostějov
Michal Tkadlec, V.B, GJŠ, Přerov
Pavel Paloncý, 1.A, G Šumperk
6. *Zuzana Králová*, V.B, G Ostrava Hrabůvka
7. *Matěj Hořínek*, 5.A, MOG, Bruntál
- 8.–9. *Ondřej Májek*, 1.C, GMK, Bílovec
Michal Kliment, V.B, G Orlová-Lutyně
- 10.–13. *Aleš Havel*, V.A, G Frenštát p. Radh.
Tomáš Babinec, V.A G Frenštát p. Radh.
Zdeněk Jahn, 3A/6, RG Prostějov, Studentská
Veronika Saňková, 1.A, G Hranice nad Mor.

Kategorie P

1. *Michal Malohlava*, 4.D, GMK, Bílovec
2. *Tomáš Gavenčíak*, 4.D, GMK, Bílovec
3. *Norbert Požár*, 8.A, MOG, Bruntál

Kategorie C

Texty úloh

C – I – 1

Najděte všechna trojmístná čísla n taková, že poslední trojčíslí čísla n^2 je shodné s číslem n .
(J. Zhouf)

C – I – 2

Sestrojte lichoběžník, jsou-li dány délky 9 cm a 12 cm jeho úhlopříček, délka 8 cm střední příčky a vzdálenost 2 cm středů úhlopříček.
(E. Kováč)

C – I – 3

Najděte všechny dvojice přirozených čísel a, b , pro které platí

$$n(a, b) + D(a, b) = 63,$$

kde $n(a, b)$ značí nejmenší společný násobek a $D(a, b)$ největší společný dělitel čísel a, b .
(L. Boček)

C – I – 4

Dokažte, že pro délky a, b, c stran libovolného trojúhelníku platí

$$\frac{(a^2 + b^2)c^2 - (a^2 - b^2)^2}{abc^2} \leq 2.$$

Pro které trojúhelníky nastane v předchozím vztahu rovnost?

(J. Šimša)

C – I – 5

Třicet maturantů jednoho gymnázia si podalo přihlášku k dalšímu studiu na některou ze šesti fakult Českého vysokého učení technického. Využili

možnost podat více přihlášek, a tak polovina žáků podala přihlášku aspoň na tři fakulty, třetina si podala přihlášku na více než tři fakulty. Na fakultu architektury se s ohledem na talentovou přijímací zkoušku nehlásil nikdo. Dokažte, že na některou ze zbývajících pěti fakult se přihlásilo méně než dvacet studentů. (P. Hliněný)

C – I – 6

Do dané kružnice s poloměrem r vepište lichoběžník $ABCD$ s kratší základnou CD a průsečíkem úhlopříček E tak, aby platilo $|BC| = |CD|$ a $|AE| = r$. (P. Leischner)

C – S – 1

Najděte všechny trojice a, b, c přirozených čísel, pro které současně platí

$$n(ab, c) = 2^8, \quad n(bc, a) = 2^9, \quad n(ca, b) = 2^{11},$$

kde $n(x, y)$ značí nejmenší společný násobek přirozených čísel x a y . (P. Černek)

C – S – 2

V rovině je dán čtverec $ABCD$. Kružnice k prochází body A, B a dotýká se přímky CD . Označme M ($M \neq B$) průsečík kružnice k a strany BC . Určete poměr $|CM| : |BM|$. (J. Švrček)

C – S – 3

Pro která dvojmístná čísla n je číslo $n^3 - n$ dělitelné stem? (J. Zhouf)

C – II – 1

Najděte všechny dvojice přirozených čísel a, b , pro které platí

$$a + b + D(a, b) + n(a, b) = 50,$$

kde $D(a, b)$ značí největší společný dělitel a $n(a, b)$ nejmenší společný násobek přirozených čísel a, b . (J. Šimša)

C – II – 2

Kružnice $k(S, r)$ a $l(O, R)$ se vně dotýkají v bodě T . Jejich společná tečna v bodě T protíná jejich vnější společnou tečnu v bodě M . Dokažte, že trojúhelník SOM je pravoúhlý, a vyjádřete jeho obsah pomocí poloměrů r, R daných kružnic. (P. Leischner)

C – II – 3

Najděte všechny dvojice kladných čísel x, y , které jsou řešením soustavy rovnic

$$x \cdot y_{10} = 195,6,$$

$$y \cdot x_{10} = 241,7.$$

Zápis z_{10} značí číslo, které vznikne zaokrouhlením čísla z na desítky.

(S. Bednářová)

C – II – 4

Sestrojte trojúhelník ABC takový, že výška a těžnice z vrcholu C dělí těžnici z vrcholu A na tři shodné úsečky, je-li dána délka strany AB a velikost výšky z vrcholu C . (J. Földes)

C – I – 1

Budeme hledat nejdříve ta čísla n , pro která je poslední číslice čísla n^2 totožná s poslední číslicí čísla n . V tom případě se musí poslední číslice čísla n rovnat některé z číslic 0, 1, 5 nebo 6.

Vezměme například 6 a označme b , a předcházející číslice čísla n , tedy $n = 100b + 10a + 6$, $n^2 = 10\,000b^2 + 2\,000ab + 1\,200b + 100a^2 + 120a + 36$. Tato dvě čísla se shodují v posledním dvojčíslí právě tehdy, jestliže se $20a + 36$ rovná $10a + 6$ až na celý násobek čísla 100, tedy $10a + 30$ má být násobek 100, což platí pouze pro $a = 7$. Je tedy $n = 100b + 76$, $n^2 = 10\,000b^2 + 15\,200b + 5\,776$. Tato dvě čísla se shodují v posledních třech číslicích, právě když se $200b + 776$ rovná $100b + 76$ až na násobek čísla 1 000, to znamená, že $b + 7$ má být celý násobek čísla 10, proto $b = 3$. Jedním řešením je číslo $n = 376$. Podobně bychom dostali další řešení $n = 625$, zatímco předpoklad, že poslední číslice je 0 nebo 1, nevede k cíli.

Výhodnější je ale postup založený na dělitelnosti — dvě čísla se shodují v posledních třech číslicích, právě když je jejich rozdíl dělitelný číslem 1 000. V našem případě má být číslo $n^2 - n = n(n - 1)$ dělitelné číslem $1\,000 = 2^3 \cdot 5^3$. Čísla n a $n - 1$ jsou nesoudělná a menší než 1 000, proto musí být jedno dělitelné číslem 125 a druhé osmi.

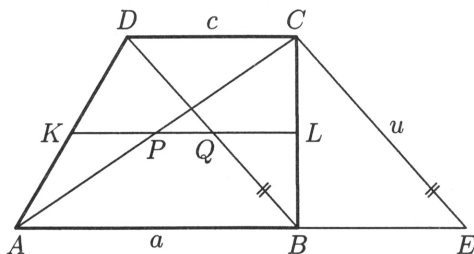
První možnost je tedy: n je *lichým* násobkem 125, takže se rovná některému z čísel 125, 375, 625, 875, a současně je $n - 1$ násobek osmi, proto $n = 625$.

Druhá možnost: n je násobek 8 (tedy sudé) a $n - 1$ lichý násobek 125, proto $n = 376$, neboť z čísel 126, 376, 626, 876 je pouze číslo 376 násobek osmi.

Poznámka. Při tomto postupu hraje významnou roli nesoudělnost dvou čísel. Připomeňte si pojem největšího společného dělitele čísel a , b a ověřte, že ten dělí také každé číslo tvaru $ka + lb$, kde k , l jsou celá. Existují-li tedy celá k , l tak, že $ka + lb = 1$, jsou čísla a , b nesoudělná. Je-li d největší společný dělitel čísel a , b , je $a = dp$, $b = dq$, kde p , q jsou nesoudělná a číslo $n = dpq$ je nejmenším společným násobkem čísel a , b .

C - 1 - 2

Zvolme označení podle obr. 1, KP je střední příčka v trojúhelníku ACD ,



Obr. 1

proto $|KP| = \frac{1}{2}|DC|$, obdobně $|QL| = \frac{1}{2}|DC|$, $|PL| = \frac{1}{2}|AB|$, takže $|PQ| = \frac{1}{2}(a - c) = 2\text{ cm}$. Protože $|KL| = \frac{1}{2}(a + c) = 8\text{ cm}$, je $a = 10\text{ cm}$, $c = 6\text{ cm}$. Nejdříve sestrojíme trojúhelník AEC podle věty *sss*, na úsečce AE pak bod B , jím vedeme rovnoběžku s CE . Ta protne přímkou vedenou bodem C rovnoběžně s AE v bodě D .

C - 1 - 3

Využijeme to, co jsme uvedli v závěru řešení 1. úlohy. Je $a = Dp$, $b = Dq$, $n = Dpq$, kde D je největší společný dělitel, n nejmenší společný násobek čísel a , b a čísla p , q jsou nesoudělná. Podle textu úlohy má platit $D(1 + pq) = 63$, takže máme tyto možnosti (bez újmy na obecnosti předpokládáme, že $a \leq b$):

D	pq	(p, q)	(a, b)
1	62	$(1, 62), (2, 31)$	$(1, 62), (2, 31)$
3	20	$(1, 20), (4, 5)$	$(3, 60), (12, 15)$
7	8	$(1, 8)$	$(7, 56)$
9	6	$(1, 6), (2, 3)$	$(9, 54), (18, 27)$
21	2	$(1, 2)$	$(21, 42)$

Úloha má 8 řešení, nerozlišujeme-li pořadí čísel a, b .

C – I – 4

Uvedenou nerovnost upravíme na ekvivalentní tvar

$$0 \leq (a^2 - b^2)^2 - (a^2 - 2ab + b^2)c^2,$$

tj.

$$0 \leq (a - b)^2 \cdot [(a + b)^2 - c^2].$$

Protože $a + b > c$, platí tato nerovnost pro délky libovolného trojúhelníku, rovnost nastane, právě když $a = b$, tedy pro rovnoramenné trojúhelníky se základnou c .

C – I – 5

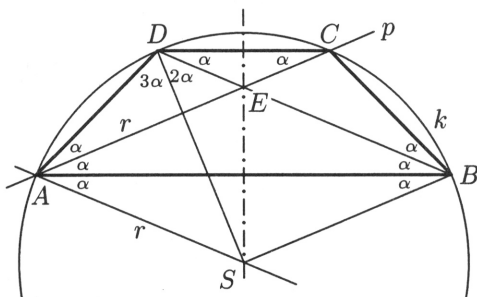
Nejdříve odhadneme, kolik přihlášek celkem maturanti podali. Polovina, tj. 15 studentů, podala jednu nebo dvě přihlášky. Z druhé poloviny podalo 10 studentů aspoň 4, tedy 4 nebo 5 přihlášek, a zbývajících pět studentů podalo přihlášku právě na tři fakulty. Celkem podali nejvýše $15 \cdot 2 + 5 \cdot 3 + 10 \cdot 5 = 95$ přihlášek. Proto nemohli podat na každou fakultu aspoň 20 přihlášek, to by jich muselo být aspoň 100.

Následující rozpis pro případ ukazuje situaci, kdy na každou z pěti fakult bylo podáno právě 19 přihlášek: A znamená, že žák v příslušném sloupci podal přihlášku na fakultu uvedenou v řádku, znak – znamená, že přihlášku nepodal. Studenti jsou rozděleni do tří skupin, první je složena z 15 studentů, kteří podali dvě přihlášky. Následuje skupina pěti studentů s třemi přihláškami, třetí skupina má 10 členů, z nichž každý podal 5 přihlášek:

student	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1. fakulta:	A	A	A	A	A	A	–	–	–	–	–	–	–	–	–
2. fakulta:	–	–	–	A	A	A	A	A	A	–	–	–	–	–	–
3. fakulta:	–	–	–	–	–	–	A	A	A	A	A	A	–	–	–
4. fakulta:	–	–	–	–	–	–	–	–	–	A	A	A	A	A	A
5. fakulta:	A	A	A	–	–	–	–	–	–	–	–	–	A	A	A

student	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1. fakulta	A	A	A	–	–	A	A	A	A	A	A	A	A	A	A
2. fakulta	–	A	A	A	–	A	A	A	A	A	A	A	A	A	A
3. fakulta	–	–	A	A	A	A	A	A	A	A	A	A	A	A	A
4. fakulta	A	–	–	A	A	A	A	A	A	A	A	A	A	A	A
5. fakulta	A	A	–	–	A	A	A	A	A	A	A	A	A	A	A

Předpokládejme, že jsme již lichoběžník sestrojili (obr. 2), přímka SE je



Obr. 2

nutně jeho osou souměrnosti. Označíme-li α velikost úhlu ACD , mají stejnou velikost i úhly BDC , ABD a CAB , jak plyne ze souměrnosti lichoběžníku podle přímky SE a z rovnoběžnosti přímek CD a AB . Protože $|BC| = |CD|$, je trojúhelník BCD rovnoramenný, a proto se α rovnají i velikosti úhlů CBD a CAD . A protože $|AE| = |AS|$ a AB je kolmá na SE , rovnají se α také velikosti úhlů SAB a SBA . Z rovnoramenného trojúhelníku ASD plyne, že úhly SAD a SDA mají velikost 3α , velikost úhlu SDB je 2α (trojúhelník SDB je také rovnoramenný). Z trojúhelníku ACD pak plyne, že $8\alpha = 180^\circ$, $\alpha = 22,5^\circ$. Tím už je dána *konstrukce*: zvolíme na dané kružnici libovolně bod A , jím vedeme přímku p svírající s přímkou AS úhel $2\alpha = 45^\circ$, p protne kružnici k v bodě C různém od A . Na úsečce AC zvolíme bod E , $|AE| = r$. Body B, D sestrojíme jako body souměrně sdružené k bodům A, C podle přímky SE . Jiná volba bodu A by vedla pouze k řešení, které by vzniklo otočením řešení již sestrojeného. Podobně volba druhé přímky vedené bodem A pod úhlem 45° s přímkou AS vede k řešení souměrně sdruženému k sestrojenému podle přímky AS .

C - S - 1

Jsou-li čísla a, b, c řešením úlohy, jsou to dělitelé mocnin dvou, a tedy sama mocniny čísla 2, $a = 2^r$, $b = 2^s$, $c = 2^t$, kde r, s, t jsou celá nezáporná čísla. Z rovnosti $n(ab, c) = 2^8$ plyne, že čísla $t, r + s$ se rovnají nejvýše 8, přičemž aspoň jedno z nich se rovná 8. Podobně se čísla $s + t, r$

rovnají nejvýše devíti a aspoň jedno z nich je rovno devíti. Dále se jedno z čísel $r + t$, s rovná 11 a žádné z nich není větší než 11. Nemůže však platit $s = 11$, protože $s + t \leq 9$, takže $r + t = 11$. Nemůže být $r = 9$, neboť má platit $r + s \leq 8$. Proto $s + t = 9$. Dále máme dvě možnosti:

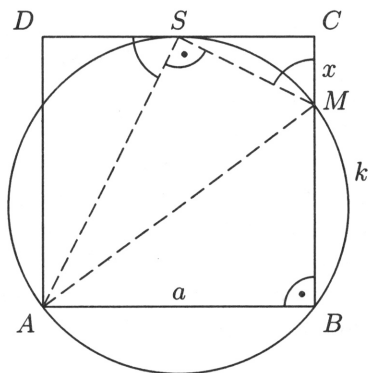
1) $t = 8$, odkud $r = 3$, $s = 1$, $a = 2^3$, $b = 2$, $c = 2^8$,

2) $r + s = 8$, odkud plyne $t = 6$, $r = 5$, $s = 3$, tedy $a = 2^5$, $b = 2^3$, $c = 2^6$.

Úloha má dvě řešení.

C – S – 2

Protože střed kružnice k leží na ose strany AB , která je zároveň osou i protější strany CD , dotýká se kružnice k úsečky CD v jejím středu S (obr. 3). Protože úhel ABM je pravý, je AM průměrem kružnice k ,



Obr. 3

a proto je pravý i úhel ASM . Odtud plyne, že $|\angle DSA| = 90^\circ - |\angle CSM| = |\angle SMC|$, proto jsou trojúhelníky SMC a ASD podobné, takže $|CM| : |CS| = |DS| : |DA|$. Označíme-li $a = |DA|$ a $x = |CM|$, je $x : \frac{1}{2}a = \frac{1}{2}a : a$, tedy $x = \frac{1}{4}a$. Proto $|CM| : |BM| = 1 : 3$.

Dodejme, že rovnost $x = \frac{1}{4}a$ lze odvodit i z Pythagorovy věty pro trojúhelníky AMB , AMS :

$$|AB|^2 + |BM|^2 = |AM|^2 = |AS|^2 + |SM|^2,$$

takže

$$a^2 + (a - x)^2 = (a^2 + (\frac{1}{2}a)^2) + ((\frac{1}{2}a)^2 + x^2),$$

odkud po úpravě

$$x = \frac{1}{4}a.$$

Poznámka. Známe-li pojem mocnosti bodu ke kružnici, můžeme napsat $|CM| \cdot |CB| = |CS|^2$, odkud ihned plyne $|CM| = \frac{1}{4}a$.

C – S – 3

Je-li číslo $n^3 - n = n(n-1)(n+1)$ dělitelné číslem $100 = 2^2 \cdot 5^2$, musí být jedno z čísel $n-1$, n , $n+1$ dělitelné číslem 25, protože ze tří po sobě jdoucích čísel může být nejvýše jedno dělitelné pěti. Dále musí být buď číslo n dělitelné čtyřmi (čísla $n-1$, $n+1$ jsou pak lichá), nebo musí být číslo n liché (čísla $n-1$, $n+1$ jsou sudá a jejich součin je dělitelný čtyřmi). Máme tedy tyto možnosti:

$n = 25$ vyhovuje, neboť je liché,

$n = 75$ vyhovuje, neboť je liché,

$n = 50$ nevyhovuje, neboť je sudé, ale není dělitelné čtyřmi,

$n-1 = 25$, $n = 26$ nevyhovuje, neboť je sudé, ale není dělitelné čtyřmi,

$n-1 = 50$, $n = 51$ vyhovuje, neboť je liché,

$n-1 = 75$, $n = 76$ vyhovuje, neboť je dělitelné čtyřmi,

$n+1 = 25$, $n = 24$ vyhovuje, neboť je dělitelné čtyřmi,

$n+1 = 50$, $n = 49$ vyhovuje, neboť je liché,

$n+1 = 75$, $n = 74$ nevyhovuje, neboť je sudé, ale není dělitelné čtyřmi,

$n+1 = 100$, $n = 99$ vyhovuje, neboť je liché.

Úloha má sedm řešení.

C – II – 1

Položme $a = Dk$, $b = Dl$, kde $D = D(a, b)$ je největší společný dělitel čísel a , b , takže čísla k , l jsou nesoudělná. Je pak $n = n(a, b) = Dkl$ a má platit $D(k+l+1+kl) = 50$, tedy $(1+k)(1+l)D = 50$. Najdeme proto všechny rozklady čísla 50 na součin tří přirozených čísel D , $1+k$, $1+l$, z nichž poslední dvě jsou větší než 1. Bez újmy na obecnosti můžeme

předpokládat, že $a \leq b$, tj. $k \leq l$. Dostaneme tak tyto možnosti:

D	$1+k$	$1+l$	k	l	a	b
1	2	25	1	24	1	24
1	5	10	4	9	4	9
5	2	5	1	4	5	20

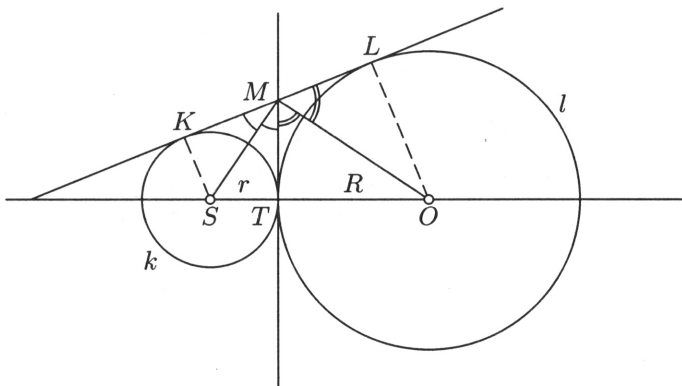
Pro $D = 2$ dostaneme $k = l = 4$, ale k, l mají být nesoudělná.

Pro $D = 10, 25$ nebo 50 dostaneme $k = 0$, což nevede k žádnému řešení.

Úloha má šest řešení: $\{a, b\} = \{1, 24\}$, $\{a, b\} = \{4, 9\}$, $\{a, b\} = \{5, 20\}$.

C – II – 2

Označme K, L body dotyku té společné tečny obou kružnic, na které leží také bod M a která je různá od společné tečny v bodě T (obr. 4). Ze souměrnosti podle přímky MS plyne shodnost úhlů KMS a TMS a ze souměrnosti podle přímky OM plyne shodnost úhlů LMO a TMO . Součet těchto čtyř úhlů je 180° , proto $|\sphericalangle SMO| = |\sphericalangle SMT| + |\sphericalangle TMO| = 90^\circ$. Tím je vyřešena první část úlohy.



Obr. 4

Užitím Pythagorovy věty pro trojúhelníky SOM , STM a OTM dostaneme pro výšku $v = |TM|$ trojúhelníku SOM rovnost

$$(r + R)^2 = (r^2 + v^2) + (R^2 + v^2),$$

odkud $v^2 = Rr$. (Tento vztah plyne i přímo z Eukleidovy věty pro trojúhelník SOM .)

Obsah trojúhelníku SOM je tedy $\frac{1}{2}(R+r)\sqrt{Rr}$.

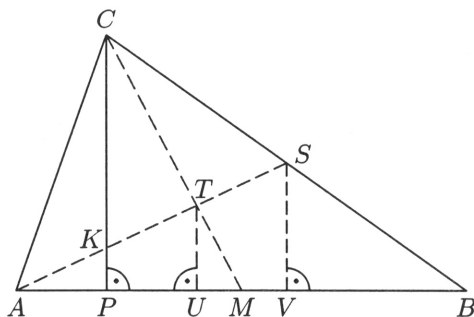
C – II – 3

Jsou-li x, y řešením, musí být $x \geq 5$ a $y \geq 5$, jinak by se x_{10} nebo y_{10} rovnalo nule. Protože $y_{10} \geq 10$, je $x = 195,6 : y_{10} \leq 19,56$, takže x_{10} se rovná 10 nebo 20. V prvním případě je $y = 24,17$, $y_{10} = 20$ a $x = 9,78$, v druhém případě je $y = 12,085$, $y_{10} = 10$ a $x = 19,56$. Úloha má právě dvě řešení:

$$(x, y) = (19,56; 12,085) \quad \text{a} \quad (x, y) = (9,78; 24,17).$$

C – II – 4

Předpokládejme, že trojúhelník ABC splňuje podmínky úlohy. Označme S střed strany BC , M střed strany AB , T těžiště trojúhelníku, P patu výšky vedené bodem C , K průsečík těžnice AS a výšky CP . Protože těžiště T dělí úsečku AS v poměru $2 : 1$, tj. platí $|AT| = 2|TS|$, musí být bod K středem úsečky AT (obr. 5). Z rovnosti $|AK| = |KT| = |TS|$ navíc plyne, že $|AP| = |PU| = |UV|$, kde U, V jsou kolmé průměty bodů T, S na přímku AB . Jelikož S je střed strany BC , je V střed úsečky PB . Proto $|AP| = \frac{1}{5}|AB|$. Odtud již plyne *konstrukce*: Sestrojíme úsečku AB dané délky, na ní bod P tak, aby $|AP| = \frac{1}{5}|AB|$. Bodem P vedeme kolmici k AB , na ni nanese od bodu P danou výšku a dostaneme tak bod C , a tím i trojúhelník ABC .



Obr. 5

Důkaz správnosti konstrukce. V sestrojeném trojúhelníku ABC uvažujme těžnice CM a AS , těžiště T a průsečík K úseček AS, CP . Označme

U, V kolmé průměty bodů T, S do přímky AB . Protože $|CT| = 2|TM|$, je $|PU| = 2|UM|$, a proto $|PU| = \frac{2}{3}|PM|$. Označíme-li $c = |AB|$, je $|AP| = \frac{1}{5}c$, $|PV| = \frac{1}{2}(|AB| - |AP|) = \frac{2}{5}c$, $|PM| = \frac{1}{2}c - \frac{1}{5}c = \frac{3}{10}c$, $|PU| = \frac{2}{3}|PM| = \frac{1}{5}c$ a $|UV| = |PV| - |PU| = \frac{1}{5}c$. Protože $|AP| = |PU| = |UV|$, je také $|AK| = |KT| = |TS|$. Tím je správnost konstrukce dokázána.

Kategorie B

Texty úloh

B – I – 1

Řešte v oboru kladných čísel soustavu rovnic

$$3x + y_{10} = 598,6,$$

$$x_{10} + 2y = 723,4,$$

v níž x_{10} a y_{10} označují po řadě čísla x a y zaokrouhlená na desítky.
(S. Bednářová)

B – I – 2

Na povrchu krychle $ABCDEFGH$ je sestrojena lomená čára složená ze čtyř shodných úseček ve stěnách $ABFE$, $BCGF$, $CDHG$ a $GHEF$, která vychází z vrcholu A a končí ve vrcholu E . Určete, v jakém poměru dělí tato lomená čára hranu CG .
(J. Zhouf)

B – I – 3

Do každého pole čtvercové tabulky $n \times n$ vepíšeme jedno z čísel $1, 2, \dots, n$ tak, aby v každém řádku i v každém sloupci byla buď všechna čísla stejná, nebo všechna různá. Příkladem pro $n = 5$ je následující tabulka

5	4	1	2	3
3	3	3	3	3
4	1	2	5	3
1	2	5	4	3
2	5	4	1	3

Označme S součet všech čísel tabulky. Kolik různých hodnot S pro dané n existuje?
(J. Šimša)

B – I – 4

Nechť k je kružnice opsaná trojúhelníku ABC , D je průsečík těžnice na stranu AB s kružnicí k . Tečny ke kružnici k v bodech A, B, C, D vytvářejí čtyřúhelník $PQRS$. Zjistěte, pro které trojúhelníky ABC je čtyřúhelník $PQRS$ tětiový. (J. Földes)

B – I – 5

Určete všechny polynomy P , které pro každé reálné číslo x splňují rovnost

$$P(2x) = 8P(x) + (x - 2)^2.$$

(P. Černek)

B – I – 6

Sestrojte trojúhelník ABC s obsahem 18 cm^2 a následující vlastností: obvod každého pravoúhelníku $KLMN$, jehož vrcholy K, L leží na úsečce BC a body M, N po řadě na úsečkách AC, AB , je roven třem pětinaám obvodu trojúhelníku ABC . (S. Bednářová)

B – S – 1

Najděte všechna trojmístná čísla n , jejichž druhá mocnina končí stejným trojčíslem jako druhá mocnina čísla $3n - 2$. (J. Šimša)

B – S – 2

Je dán tětiový čtyřúhelník $ABCD$. Označme E průsečík přímek BC a AD . Leží-li průsečík úhlopříček AC a BD na ose úhlu AEB , je trojúhelník ABE rovnoramenný. Dokažte. (E. Kováč)

B – S – 3

Určete mnohočleny P a Q takové, že pro všechna reálná čísla x platí

$$Q(x^2) = (x + 1)^4 - x(P(x))^2.$$

(P. Černek)

B – II – 1

Určete všechna reálná čísla p taková, že pro libovolná kladná čísla x, y platí nerovnost

$$\frac{x^3 + py^3}{x + y} \geq xy.$$

(J. Bábela)

B – II – 2

Je dán trojúhelník ABC . Sestrojte rovnoběžník $KLMN$ tak, aby jeho vrcholy K a L ležely na straně AB , vrchol M na straně BC , vrchol N na straně AC a aby trojúhelníky AKN , LBM a NMC měly stejné obsahy.

(J. Šimša)

B – II – 3

Určete všechna přirozená čísla n , pro která je podíl

$$\frac{(n^2)_{10}}{(n_{10})^2}$$

celé číslo. Zápis z_{10} značí číslo, které vznikne zaokrouhlením čísla z na desítky.

(S. Bednářová)

B – II – 4

Najděte všechny ostroúhlé trojúhelníky ABC , jejichž těžiště T splývá s průsečíkem výšek trojúhelníku PQR , přičemž body P, Q, R jsou po řadě průsečíky polopřímek opačných k polopřímkám TA, TB, TC s kružnicí opsanou trojúhelníku ABC .

(J. Földes)

Řešení úloh

B – I – 1

Nechť

$$x = x_{10} + m, \quad y = y_{10} + n, \quad -5 \leq m, n \leq 5, \quad (1)$$

$$a = 3x_{10} + y_{10}, \quad b = x_{10} + 2y_{10}. \quad (2)$$

Čísla a, b jsou násobky deseti a původní soustavu rovnic můžeme přepsat ve tvaru

$$a = 598,6 - 3m, \quad b = 723,4 - 2n. \quad (3)$$

Čísla m, n jsou z intervalu $\langle -5, 5 \rangle$, proto $a \in \{590, 600, 610\}$ a $b \in \{720, 730\}$. Dále ze (2) dostáváme

$$x_{10} = \frac{1}{5}(2a - b), \quad y_{10} = \frac{1}{5}(3b - a). \quad (4)$$

Vidíme, že čísla $2a - b$ a $3b - a$ musejí být dělitelná padesáti, a proto přicházejí v úvahu jen dvojice $[a, b] = [590, 730]$, $[a, b] = [610, 720]$. Nalezené hodnoty čísel a, b postupně dosadíme do (4) a (3). Pomocí (1) určíme x a y :

V prvním případě je $x_{10} = 90$, $y_{10} = 320$, $m = \frac{43}{15}$, $n = -3,3$, $x = \frac{1393}{15} = 92,8\bar{6}$ a $y = 316,7$, ve druhém $x_{10} = 100$, $y_{10} = 310$, $m = -3,8$, $n = 1,7$, $x = 96,2$ a $y = 311,7$.

B – I – 2

Označme K, L, M body dané lomené čáry, jež po řadě leží na úsečkách BF, CG, GH . Délku hrany krychle položíme rovnu jedné a patu kolmice z bodu K na hranu CG označme P (obr. 6). Pravoúhlé trojúhelníky AKB, KLP a EMH se shodují v přeponách AK, KL a EM a v jednotkových odvěsnách AB, KP a EH . Jsou tedy podle věty Ssu shodné a platí $|BK| = |LP| = |MH| = u$. Z pravoúhlých trojúhelníků LMG a ABK ($|GL| = 1 - 2u$, $|GM| = 1 - u$) vyjádříme pomocí Pythagorovy věty druhé mocniny délek jejich přepon a porovnáme je: $1 + u^2 = (1 - u)^2 + (1 - 2u)^2$.

Rovnice má jediný kořen menší než 1: $u = \frac{1}{4}(3 - \sqrt{5})$. Poměr $|CL| : |LG| = 2u : (1 - 2u) = \frac{1}{2}(\sqrt{5} - 1)$ je roven poměru zlatého řezu.

Pro polohu bodu L na hraně CG je ještě jedna možnost, znázorněná na stejné části sítě krychle na obr. 7. Zřejmě jsou pak body C a L totožné

Dosazením daného n a postupně všech možných hodnot čísel a, c do vztahů (5), (6) a (7) dostaneme celkem $2n + 1$ součtů, z toho n součtů typu (6) je navzájem různých a n součtů typu (7) je navzájem různých. Musíme tedy ještě vyšetřit, zda není možná pro nějaké hodnoty čísel a, c rovnost součtů (5) a (6), nebo (5) a (7), nebo (6) a (7).

V prvním případě z rovnice $\frac{1}{2}n^2(n+1) = na + \frac{1}{2}n(n^2 - 1)$ zjistíme, že rovnost nastane pro $a = \frac{1}{2}(n+1)$, to znamená, jen když n je liché.

Ve druhém případě dojdeme analogicky k závěru, že (5) a (7) se rovnají opět jen pro n liché a $c = \frac{1}{2}(n+1)$.

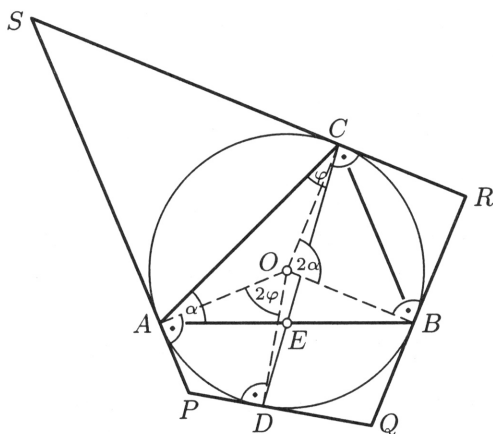
Ve třetím případě upravíme rovnici $na + \frac{1}{2}n(n^2 - 1) = n^2c$ na tvar $2a - 1 = n(2c - n)$, z něhož plyne, že pokud taková dvě čísla $a, c \in \{1, 2, \dots, n\}$ existují, je číslo n nutně liché a číslo $2a - 1$ je jeho násobkem. Je však $2a - 1 \leq 2n - 1$, proto může být jedině $2a - 1 = n$ a $2c - n = 1$. Odtud $a = \frac{1}{2}(n+1) = c$.

Shrnutím všech tří situací můžeme konstatovat, že pro n sudé je všech $2n + 1$ součtů S různých, kdežto pro n liché se mezi těmito součty vyskytují právě tři stejné.

Odpověď: Součet S všech čísel tabulky nabývá buď $2n + 1$ hodnot (když n je sudé), nebo $2n - 1$ hodnot (když n je liché).

B - I - 4

Nechť O je střed kružnice opsané trojúhelníku ABC . Při označení podle obr. 8 jsou úhly PAO a PDO pravé a velikost středového úhlu AOD



Obr. 8

je dvojnásobkem velikosti příslušného obvodového úhlu ACD . Ve čtyřúhelníku $APDO$ je tedy $|\sphericalangle APD| = 180^\circ - 2|\sphericalangle ACD|$. Analogicky $|\sphericalangle BRC| = 180^\circ - 2|\sphericalangle BAC|$. Čtyřúhelník $PQRS$ je tětiový, právě když $|\sphericalangle SPQ| + |\sphericalangle SRQ| = 180^\circ$, tj. $180^\circ - 2|\sphericalangle ACD| + 180^\circ - 2|\sphericalangle BAC| = 180^\circ$. Odtud vychází pro to, aby čtyřúhelník $PQRS$ byl tětiový, nutná a postačující podmínka $|\sphericalangle ACD| + |\sphericalangle BAC| = 90^\circ$. Při označení podle obr. 8 to znamená $|\sphericalangle ACE| + |\sphericalangle EAC| = 90^\circ$, tj. těžnice CE je kolmá na AB , jak je vidět z trojúhelníku ACE . Je tedy $|AC| = |BC|$, protože trojúhelníky AEC a BEC jsou shodné podle věty *sus*.

Závěr: Čtyřúhelník $PQRS$ je tětiový jen tehdy, je-li trojúhelník ABC rovnoramenný se základnou AB .

B – I – 5

Stupeň polynomu P je aspoň dva. Nechť nejprve $P(x) = ax^2 + bx + c$. Dosazením tohoto vyjádření do vztahu v zadání dostáváme

$$4ax^2 + 2bx + c = (8a + 1)x^2 + (8b - 4)x + 8c + 4.$$

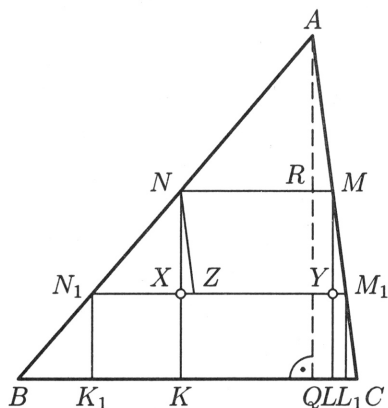
Porovnáním koeficientů u stejných mocnin x na levé a pravé straně dostaneme $4a = 8a + 1$, $2b = 8b - 4$ a $c = 8c + 4$. Odtud $a = -\frac{1}{4}$, $b = \frac{2}{3}$, $c = -\frac{4}{7}$.

Je-li dále stupeň n polynomu P větší než dva, zjistíme analogicky, že jeho člen $a_n x^n$ s nejvyšší mocninou x splňuje vztah $2^n a_n = 8a_n$, tedy $n = 3$, přičemž $a_n \neq 0$ je libovolné. Koeficienty mnohočlenu P u mocnin x^2 , x^1 a x^0 vyjdou stejně jako v předchozí situaci.

Závěr: Úloze vyhovují všechny mnohočleny $P(x) = ax^3 - \frac{1}{4}x^2 + \frac{2}{3}x - \frac{4}{7}$, kde a je libovolné reálné číslo.

B – I – 6

Uvažujme dva pravoúhelníky $KLMN$, $K_1L_1M_1N_1$ vepsané do trojúhelníku ABC uvedeným způsobem. Nechť $|KL| < |K_1L_1|$. Označme Z průsečík rovnoběžky s AC vedené bodem N s úsečkou N_1M_1 , Q patu výšky z vrcholu A na stranu BC a X , Y průsečíky hranice pravoúhelníku $KLMN$ s úsečkou N_1M_1 (obr. 9). Obvody obou pravoúhelníků jsou si rovny, právě když $|N_1X| + |YM_1| = |NX|$. To je ekvivalentní s podmínkou $|NX| = |N_1Z|$, neboť $|XZ| = |YM_1|$. Trojúhelníky BCA , N_1ZN i NMA si jsou podobné, proto $a = v_a$. A protože $S = a \cdot \frac{1}{2}v_a = 18 \text{ cm}^2$, plyne odtud rovnost $a = v_a = 6 \text{ cm}$.



Obr. 9

Obvod pravoúhelníku $KLMN$ je $2|NM| + 2|KN| = 2|AR| + 2|RQ| = 2v_a = 2a = 12$ cm. Obvod $2s$ trojúhelníku ABC je proto $\frac{5}{3} \cdot 12$ cm = 20 cm. Odtud $b + c = 2s - a = 14$ cm.

Máme tedy sestrojit trojúhelník ABC , je-li dáno a , v_a , $b + c$.

Uvedeme několik postupů řešení.

1. možnost: Snadno vypočteme $s = 10$ (cm), $s - a = 4$, $s - b = 10 - b$, $s - c = b - 4$. Po dosazení do Heronova vzorce pro obsah trojúhelníku ABC dostaneme $\sqrt{40 \cdot (10 - b) \cdot (b - 4)} = 18$, což vede po úpravě na kvadratickou rovnici $10b^2 - 140b + 481 = 0$. Jejím řešením obdržíme

$$b = 7 + \frac{3}{\sqrt{10}}, \quad c = 7 - \frac{3}{\sqrt{10}}, \quad \text{nebo} \quad b = 7 - \frac{3}{\sqrt{10}}, \quad c = 7 + \frac{3}{\sqrt{10}}.$$

Obě řešení vyhovují a snadno je ze známých délek stran sestrojíme. Délku $d = 3/\sqrt{10}$ nalezneme eukleidovsky jako čtvrtou geometrickou úměrnou tak, že vztah přepíšeme na tvar $d : 1 = 3 : \sqrt{10}$. Nejdrříve ovšem sestrojíme $\sqrt{10}$ např. pomocí Eukleidovy věty o výšce.

2. možnost: Nechť $k(O, r)$ je kružnice vepsaná trojúhelníku ABC a T její bod dotyku se stranou AC (obr. 10). Pravoúhlý trojúhelník AOT můžeme sestrojit, neboť známe délky jeho odvěsen $|AT| = x = s - a = 4$ cm, $|TO| = r = S/s = 1,8$ cm, dále kružnici $k(O, r)$ a nad přeponou AO ještě jeden pravoúhlý trojúhelník s odvěsnou AE délky $v_a - r = 4,2$ cm. (Tento trojúhelník zřejmě existuje — výpočtem délky přepony trojúhelníku AOT pomocí Pythagorovy věty lze ověřit, že $|AO| > v_a - r$.)

The diagram shows a triangle with vertices labeled U , B , and C . A dashed line, labeled σ , passes through the triangle. The angle at vertex U is labeled $\frac{\alpha}{2}$. The angle at vertex B is labeled $\frac{\alpha}{2}$. The angle at vertex A is labeled α .

3. možnost: Úsečka CU na obr. 11 má délku $b + c$. Trojúhelník UBA je tedy rovnoramenný se základnou UB , a proto $|\sphericalangle BUC| = \frac{1}{2}\alpha$. Tento úhel umíme sestrojit podle předchozího postupu, neboť je to úhel OAT na obr. 10. Sestrojíme tedy nejprve trojúhelník CUB , ve kterém známe $|BC|$, $|CU|$ a $|\sphericalangle CUB|$. Bod A je pak průsečík úsečky CU s osou strany BU . Konstrukce vede opět na dvě řešení.

Jestliže číslo n vyhovuje podmínke úlohy, existuje prirodzené číslo k také, že

45

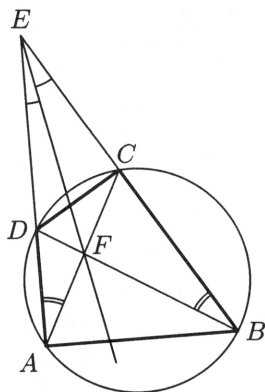
Levou stranu uvedené rovnosti rozložíme na součin podle vzorce pro rozdíl čtverců a po snadné úpravě dostaneme

$$(2n - 1)(n - 1) = 250k = 2 \cdot 5^3 \cdot k.$$

Čísla $2n - 1$ a $n - 1$ jsou nesoudělná (je $2n - 1 = 2(n - 1) + 1$), přitom první z nich je liché, takže druhé musí být sudé. Hledáme tedy *lichá* trojmístná čísla n taková, že buď $n - 1$ je násobkem 250, nebo $2n - 1$ je lichým násobkem čísla 125. V prvním případě dostaneme $n \in \{251, 501, 751\}$ a ve druhém vidíme, že musí být $2n - 1 \in \{375, 625, 875, 1125, 1375, 1625, 1875\}$, tedy (protože n je liché) $n \in \{313, 563, 813\}$. Celkem má úloha uvedených šest řešení.

B – S – 2

Označme F průsečík úhlopříček AC a BD (obr. 12). Jestliže je přímka EF osou úhlu AEB , jsou úhly AEF a BEF shodné. Navíc jsou



Obr. 12

shodné i úhly EAF a EBF , neboť jsou to obvodové úhly příslušné téže těživě CD . Trojúhelníky AFE a BFE se shodují ve společné straně EF , jsou tedy shodné podle věty *usu*, $|AE| = |BE|$, a trojúhelník ABE je tudíž rovnoramenný.

B – S – 3

Předpokládejme, že P je mnohočlen stupně n . Členy polynomu $Q(x^2)$ obsahují jen sudé mocniny x a polynom $x \cdot P^2(x)$ je lichého stupně $2n+1$. Protože má platit $Q(x^2) = (x+1)^4 - x \cdot P^2(x)$, vidíme, že nemůže být $2n+1 > 4$. Je tedy $n \leq 1$ a

$$P(x) = ax + b, \quad Q(x^2) = (x+1)^4 - x(ax+b)^2. \quad (1)$$

Po úpravě dostaneme $Q(x^2) = x^4 + (4-a^2)x^3 + (6-2ab)x^2 + (4-b^2)x + 1$. Koeficienty při lichých mocninách x jsou rovny nule, proto $a, b \in \{-2, 2\}$ a $Q(x^2) = x^4 + (6-2ab)x^2 + 1$. Dosazením každé ze čtyř možných dvojic čísel a, b do (1) nalezneme všechna čtyři řešení úlohy:

$$P(x) = 2x + 2 \text{ a } Q(x) = x^2 - 2x + 1,$$

$$P(x) = 2x - 2 \text{ a } Q(x) = x^2 + 14x + 1,$$

$$P(x) = -2x + 2 \text{ a } Q(x) = x^2 + 14x + 1,$$

$$P(x) = -2x - 2 \text{ a } Q(x) = x^2 - 2x + 1.$$

B – II – 1

Nerovnost je pro kladná x, y zřejmě ekvivalentní se vztahem

$$y^2(py - x) - x^2(y - x) \geq 0.$$

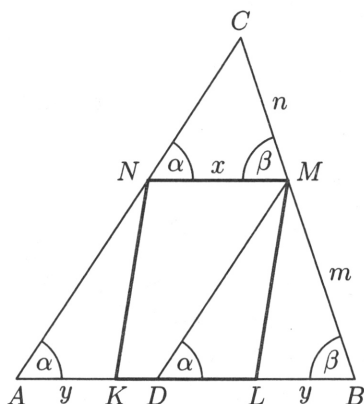
Je-li $p \geq 1$, dostáváme

$$y^2(py - x) - x^2(y - x) \geq y^2(y - x) - x^2(y - x) = (x + y)(y - x)^2 \geq 0.$$

Je-li však $p < 1$, neplatí daná nerovnost například pro $x = y = 1$.

Závěr: Daná nerovnost je splněna pro každá dvě kladná čísla x, y , právě když $p \geq 1$.

Předpokládejme, že rovnoběžník $KLMN$ má požadované vlastnosti. V posunutí o vektor $\vec{NM}$ (obr. 13) je obrazem trojúhelníku AKN troj-



Obr. 13

úhelník DLM . Vzniklý trojúhelník DBM má mít (dle zadání) dvakrát větší obsah než trojúhelník NMC a je tomuto trojúhelníku podobný (věta uu). Koeficient k podobnosti, která převádí trojúhelník DBM na trojúhelník NMC , je odmocninou z podílu obsahů těchto trojúhelníků a zároveň podílem délek libovolných dvou v podobnosti si odpovídajících úseček: $k = \sqrt{2} = \frac{|BM|}{|MC|}$. Z podmínky rovnosti obsahů trojúhelníků LBM , DLM a AKN , jejichž výšky na strany LB , DL a AK jsou shodné, navíc plyne $|LB| = |DL| = |AK|$.

Odtud plyne konstrukce: Na úsečce BC sestrojíme bod M tak, aby $|BM| : |MC| = \sqrt{2} : 1$. Rovnoběžka s přímkou AC vedená bodem M protne úsečku AB v bodě D . Vrchol N nalezneme jako průsečík úsečky AC s přímkou, která prochází bodem M rovnoběžně s AB . Bod L sestrojíme jako střed úsečky DB , bod K je pak obrazem bodu L v posunutí o vektor $\vec{MN}$.

Výsledkem konstrukce je rovnoběžník $KLMN$ (jediný pro každý trojúhelník ABC), o němž se snadno přesvědčíme, že má požadované vlastnosti.

Jiné řešení. Z rovností obsahů trojúhelníků AKN a LBM se shod-

nými výškami na strany AK a LB plyne shodnost těchto stran. Označme (obr. 13) $|AB| = c$, $|NM| = |KL| = x$, $|BM| = m$ a $|MC| = n$ a $|LB| = |AK| = y$; zřejmě $c = 2y + x$, takže $y = \frac{1}{2}(c - x)$. Trojúhelníky NMC a LBM mají stejné obsahy, je tedy $\frac{1}{2}xn \sin \beta = \frac{1}{2}ym \sin \beta$ neboli $xn = ym$. Po dosazení za y a úpravě máme

$$x(m + 2n) = mc. \quad (1)$$

Z podobnosti trojúhelníků NMC a ABC plyne úměra $x : c = n : (m + n)$ neboli

$$nc = x(m + n), \quad (2)$$

takže z rovnosti součinu levých a součinu pravých stran vztahů (1) a (2) dostaneme $m = n\sqrt{2}$, tj. $\frac{|BM|}{|MC|} = m : n = \sqrt{2}$. Odtud vyplývá konstrukce podobně jako v předchozím řešení.

B – II – 3

Položme $m = \frac{(n^2)_{10}}{(n_{10})^2}$ a $n = 10k + r$, kde k je celé nezáporné a r je poslední číslice čísla n , tj. $r \in \{0, 1, 2, \dots, 9\}$. Zřejmě je m celé pro všechna n , která mají $r = 0$ a $k > 0$. Pokud je $k = 0$ a $r \in \{1, 2, 3, 4\}$, není zlomek m definován.

Nechť je $k > 0$ a $r \in \{1, 2\}$. Pak $m = \frac{100k^2 + 20kr}{100k^2} = 1 + \frac{r}{5k}$, což není celé číslo.

Pro $k > 0$ a $r = 3$ platí $m = \frac{100k^2 + 60k + 10}{100k^2} = 1 + \frac{6k + 1}{10k^2}$. Čítecel posledního zlomku není na rozdíl od jmenovatele dělitelný deseti, tedy m není celé číslo.

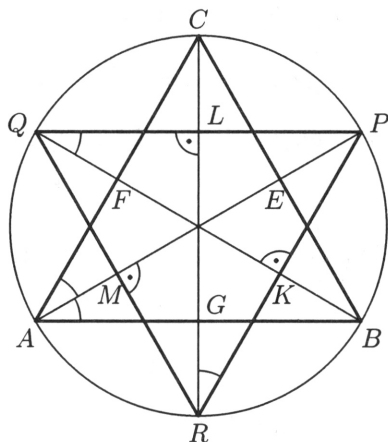
Je-li $k > 0$ a $r = 4$, máme $m = \frac{100k^2 + 80k + 20}{100k^2} = 1 + \frac{4k + 1}{5k^2}$. Odtud $m = 2$ pro $k = 1$. Pro $k > 1$ je $\frac{4k + 1}{5k^2} = \frac{4k + 1}{(4k + k)k} < \frac{4k + 1}{(4k + 1)k} = \frac{1}{k}$, a tak m nemůže být celé číslo.

Je-li konečně $r \in \{5, 6, 7, 8, 9\}$, dostáváme

$$m = \frac{100k^2 + 20kr + (r^2)_{10}}{100(k + 1)^2} < \frac{100k^2 + 200k + 100}{100(k + 1)^2} = 1.$$

Závěr: m je celé číslo pro všechna přirozená čísla n , jejichž dekadický zápis končí číslicí 0, a pro $n = 14$.

Pravoúhlé trojúhelníky LRP a KQP na obr. 14 jsou podobné, protože



Obr. 14

mají společný ostrý úhel při vrcholu P . Využijeme-li navíc, že obvodové úhly příslušné témuž oblouku jsou shodné, dostáváme $|\sphericalangle CAP| = |\sphericalangle CRP| = |\sphericalangle LRP| = |\sphericalangle KQP| = |\sphericalangle BQP| = |\sphericalangle BAP|$. Bod E je střed úsečky BC , tětivy CP a BP příslušné shodným obvodovým úhlům ACP a BAP jsou shodné. Trojúhelník CEP je tedy shodný s trojúhelníkem BEP podle věty *sss*, tudíž úhly AEB a BEP stejně jako úhly AEB a AEC jsou shodné (a pravé). Odtud plyne i shodnost trojúhelníků AEC a AEB podle věty *sus*. Je tedy $|AC| = |AB|$. Analogicky zjistíme, že $|AB| = |BC|$.

Závěr: Daným podmínkám vyhovují jen rovnostranné trojúhelníky ABC .

Poznámka. Rovnost $|CP| = |BP|$ lze dokázat i jinak, například na základě poznatku, že obrazy ortocentra v souměrnostech podle stran trojúhelníku leží na kružnici trojúhelníku opsané. Těžiště T trojúhelníku ABC je zároveň ortocentrem trojúhelníku PQR . Proto jsou obrazy úsečky TP v osových souměrnostech podle přímek PQ a PR po řadě úsečky CP a BP , které jsou tudíž shodné.

Kategorie A

Texty úloh

A – I – 1

V urně jsou jen bílé a černé kuličky, jejichž počet zaokrouhlen na stovky je 1 000. Pravděpodobnost vytažení dvou černých kuliček je o $\frac{17}{43}$ větší než pravděpodobnost vytažení dvou bílých kuliček. Kolik bílých a kolik černých kuliček je v urně? (Pravděpodobnost vytažení kterékoli kuličky je stejná.) (P. Černek)

A – I – 2

Nechť a_1, a_2 jsou přirozená čísla a nechť pro každé přirozené $n \geq 2$ je číslo a_{n+1} o 1 větší než největší lichý dělitel součtu $a_n + a_{n-1}$. Dokažte, že posloupnost $a_1, a_2, a_3, \dots$ je od určitého členu počínaje periodická. Najděte všechny takové posloupnosti, jež jsou periodické už od prvního členu. (J. Bábela)

A – I – 3

V rovině je dán ostroúhlý trojúhelník ABC . Paty výšek z vrcholů A, B označme po řadě A_1, B_1 . Tečny kružnice opsané trojúhelníku CA_1B_1 sestrojené v bodech A_1, B_1 se protínají v bodě M . Dokažte, že kružnice opsané trojúhelníkům AMB_1, BMA_1, CA_1B_1 procházejí jedním bodem. (J. Švrček)

A – I – 4

V oboru reálných čísel řešte soustavu nerovnic

$$\sin x + \cos y \geq \sqrt{2},$$

$$\sin y + \cos z \geq \sqrt{2},$$

$$\sin z + \cos x \geq \sqrt{2}.$$

(J. Švrček)

A – I – 5

Najděte všechny funkce $f: \mathbb{R} \rightarrow \mathbb{R}$ takové, že pro všechna reálná čísla x , y platí

$$x^2 + y^2 + 2f(xy) = f(x+y) \cdot (f(x) + f(y)).$$

(E. Kováč)

A – I – 6

Sestrojte lichoběžník $ABCD$, jsou-li dány délky jeho ramen $|BC| = 4,5$ cm, $|DA| = 3$ cm a velikost 75° úhlu, který svírají přímky BC a AD , platí-li navíc $|AB| \cdot |CD| = |AC|^2$.

(E. Kováč)

A – S – 1

Najděte všechna reálná čísla p , pro která má soustava nerovnic

$$25 + 2x^2 \leq 13y + 10z - p,$$

$$25 + 3y^2 \leq 6z + 10x,$$

$$25 + 4z^2 \leq 6x + 5y + p$$

s neznámými x , y , z řešení v oboru reálných čísel.

(J. Švrček)

A – S – 2

Je dán lichoběžník $ABCD$ se základnou AB délky a , v němž oba úhly ABC , ADB jsou pravé. Na straně AB leží bod M tak, že úsečka MD je kolmá na AC a úsečka MC je kolmá na BD . Určete délky ostatních stran lichoběžníku.

(J. Zhouf)

A – S – 3

Najděte všechna čtyřmístná čísla $\overline{abcd}$, která jsou dělitelná každým z dvojčíselných čísel $\overline{ab}$, $\overline{bc}$, $\overline{cd}$, jejichž číslice a , b , c , d jsou liché a ne všechny stejné. (J. Šimša)

A – II – 1

Najděte nejmenší čtyřmístné číslo n , pro něž má soustava

$$\begin{aligned}x^3 + y^3 + y^2x + x^2y &= n, \\x^2 + y^2 + x + y &= n + 1\end{aligned}$$

pouze celočíselná řešení.

(J. Zhouf)

A – II – 2

Určete všechna reálná čísla s a t , pro která je grafem funkce

$$f(x) = \frac{x^2 - 4x + s}{t|x - 1| + x + 7}$$

lomená čára složená ze dvou polopřímek.

(P. Černek)

A – II – 3

Je dána kružnice $k(S, r)$ a na ní body M , N takové, že úhel MSN je ostrý. Libovolným bodem X menšího z oblouků MN vedme rovnoběžku s přímkou MS a označme Y její průsečík s úsečkou SN . Sestrojte takový bod X , pro který je obsah trojúhelníku SXY maximální. (P. Černek)

A – II – 4

Určete všechny funkce $f: \mathbb{R} \rightarrow \mathbb{R}$ takové, že pro všechna reálná čísla x a y platí

$$f(x^2 + f(y)) = (x - y)^2 \cdot f(x + y).$$

(P. Calábek)

A – III – 1

Určete všechny mnohočleny P takové, že pro všechna reálná čísla x platí

$$(P(x))^2 + P(-x) = P(x^2) + P(x).$$

(P. Calábek)

A – III – 2

V rovině je dán trojúhelník PQX , kde $|PQ| = 3$ cm, $|PX| = 2,6$ cm, $|QX| = 3,8$ cm. Sestrojte pravoúhlý trojúhelník ABC tak, aby se jemu vepsaná kružnice dotýkala přepony AB v bodě P , odvěsny BC v bodě Q a aby bod X ležel na přímce AC .

(J. Šimša)

A – III – 3

Najděte všechny trojice reálných čísel a, b, c , pro které je množinou řešení nerovnice

$$\sqrt{2x^2 + ax + b} > x - c$$

s neznámou x množina $(-\infty, 0) \cup (1, \infty)$.

(P. Černek)

A – III – 4

V jistém jazyce je n písmen. Skupina písmen (napsaných za sebou) je slovo, právě když se mezi žádnými dvěma stejnými písmeny nenacházejí dvě stejná písmena. Určete počet všech slov maximální délky.

(K. Černeková)

A – III – 5

Z papíru byl vystřížen rovnoramenný lichoběžník $C_1AB_2C_2$ s kratší základnou B_2C_2 . Patu kolmice ze středu D ramena C_1C_2 na základnu AC_1 označíme B_1 . Po přehnutí papíru podél úseček DB_1 , AD a AC_2 se body C_1 , C_2 přemístily v prostoru do jednoho bodu C a body B_1 , B_2 do bodu B . Vznikl tak model čtyřstěnu $ABCD$ s objemem 64 cm^3 . Určete délky stran původního lichoběžníku.

(P. Leischner)

A – III – 6

Jsou dána přirozená čísla $a_1, a_2, \dots, a_n$ a funkce $f: \mathbb{Z} \rightarrow \mathbb{R}$ taková, že $f(x) = 1$ pro každé celé $x < 0$ a

$$f(x) = 1 - f(x - a_1) f(x - a_2) \dots f(x - a_n) \quad (1)$$

pro každé celé $x \geq 0$. Dokažte, že existují přirozená čísla s a t taková, že pro každé celé $x > s$ platí $f(x + t) = f(x)$. (P. Kaňovský)

Řešení úloh

A – I – 1

Nechť je v urně n kuliček, z toho b bílých (a $n - b$ černých). Potom pravděpodobnost vytažení dvou bílých kuliček je rovna podílu

$$\frac{\binom{b}{2}}{\binom{n}{2}} = \frac{b(b-1)}{n(n-1)},$$

zatímco pravděpodobnost vytažení dvou černých kuliček podílu

$$\frac{\binom{n-b}{2}}{\binom{n}{2}} = \frac{(n-b)(n-b-1)}{n(n-1)}.$$

Podle zadání úlohy platí rovnost

$$\frac{(n-b)(n-b-1)}{n(n-1)} = \frac{b(b-1)}{n(n-1)} + \frac{17}{43},$$

kteřou lze algebraickými úpravami zjednodušit do tvaru $43b = 13n$ (pro $n \notin \{0, 1\}$ jde o ekvivalentní rovnice). Odtud vzhledem k nesoudělnosti čísel 13 a 43 plyne, že přirozená čísla n a b jsou tvaru $n = 43k$ a $b = 13k$, kde k je vhodné přirozené číslo. Podle zadání pro číslo n platí odhady $950 \leq n < 1050$, z nichž zjistíme, že $k \in \{23, 24\}$. Pro $k = 23$ vychází $n = 989$ a $b = 299$ (tehdy $n - b = 690$), zatímco hodnotě $k = 24$ odpovídá $n = 1032$ a $b = 312$ (tehdy $n - b = 720$).

Odpověď: Úloha má dvě řešení — v urně je buď 299 bílých a 690 černých, nebo 312 bílých a 720 černých kuliček.

A – I – 2

Zvolíme-li přirozená čísla a_1, a_2 libovolně, jsou všechny následující členy $a_3, a_4, \dots$ zkoumané posloupnosti jednoznačně určeny rekurentním předpisem

$$a_{n+1} = 1 + (a_n + a_{n-1})^* \quad (n = 2, 3, \dots), \quad (1)$$

kde a^* značí největší lichý dělitel přirozeného čísla a . Vypočtěme pro několik „počátečních“ dvojic a_1, a_2 tolik prvních členů a_n , ze kterých už bude jasné, kde začíná a jak vypadá perioda dotyčné posloupnosti.¹ Několik příkladů uvádíme v následující tabulce.

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	...
1	1	2	4	4	2	4	4	2	4	...
2	1	4	6	6	4	6	6	4	6	...
1	2	4	4	2	4	4	2	4	4	...
1	3	2	6	2	2	2	2	2	2	...
2	2	2	2	2	2	2	2	2	2	...
3	1	2	4	4	2	4	4	2	4	...
1	4	6	6	4	6	6	4	6	6	...
2	3	6	10	2	4	4	2	4	4	...
3	2	6	2	2	2	2	2	2	2	...
4	1	6	8	8	2	6	2	2	2	...

(Pro větší hodnoty a_1, a_2 se perioda často objeví „později“, jak ukazuje příklad posloupnosti 30, 31, 62, 94, 40, 68, 28, 4, 2, 4, 4, 2, ...)

Posloupnost vytvořená podle předpisu (1) je od jistého členu, řekněme a_n , periodická, právě když existuje takový index m , že platí

$$m > n, \quad a_m = a_n \quad \text{a} \quad a_{m+1} = a_{n+1}. \quad (2)$$

Vlastní řešení úlohy zahájíme tak, že dokážeme čtyři tvrzení (T1) až (T4), která platí pro každou zkoumanou posloupnost $\{a_n\}$ a která lze vypočítat z příkladů uvedených v tabulce.

(T1) Číslo a_n je sudé pro každé $n \geq 3$.

Důkaz (T1) je triviální: protože je číslo a^* liché pro každé a , je pravá strana rovnosti v (1) sudá.

(T2) Nerovnost $a_n \leq \max\{a_{n-1}, a_{n-2}\}$ platí pro každé $n \geq 5$.

Důkaz (T2): Protože pro sudé a platí $a^* \leq \frac{1}{2}a$ a pro každé $n \geq 5$ jsou podle (T1) obě čísla a_{n-1}, a_{n-2} sudá, platí pro takové n odhad

$$a_n = 1 + (a_{n-1} + a_{n-2})^* \leq 1 + \frac{a_{n-1} + a_{n-2}}{2} \leq 1 + \max\{a_{n-1}, a_{n-2}\}$$

¹ Je téměř nemyslitelné řešit takové neobvyklé úlohy bez podobného úvodního experimentování. Proto experimenty patří k výzkumu v matematice stejně jako v jiných přírodních vědách.

(neboť aritmetický průměr dvou čísel nepřevyšuje větší z nich). Zjistili jsme, že sudé číslo a_n nepřevyšuje liché číslo $1 + \max\{a_{n-1}, a_{n-2}\}$, tudíž nepřevyšuje ani číslo o 1 menší, číslo $\max\{a_{n-1}, a_{n-2}\}$.

Z dokázaného tvrzení (T2) plyne, že každá zkoumaná posloupnost $\{a_n\}$ má největší člen (a ten se navíc rovná jednomu z čísel a_1, a_2, a_3, a_4). Ukažme, jak snadno již odtud plyne tvrzení o periodičnosti posloupnosti $\{a_n\}$: značí-li M její největší člen, je každá z nekonečně mnoha dvojic (a_n, a_{n+1}) ($n = 1, 2, 3, \dots$) rovna jedné z M^2 dvojic (a, b) , kde $a, b \in \{1, 2, \dots, M\}$. Proto existují dva různé indexy, řekněme $n < m$, pro které platí $(a_n, a_{n+1}) = (a_m, a_{m+1})$, tj. podmínka (2). Pak ovšem indukcí z (1) plyne, že $a_{n+k} = a_{m+k}$ pro každé $k \geq 0$. Proto je posloupnost $\{a_n\}$ periodická. První část úlohy je tak vyřešena.

Zdůrazněme ještě, že tvrzení (T2) neznamena, že posloupnost $\{a_n\}$ je od některého členu nerostoucí (vyvracejí to příklady z naší tabulky). Platí ale:

(T3) *Existuje index n_0 (závislý na posloupnosti $\{a_n\}$) takový, že pro každé $n \geq n_0$ platí rovnost $\max\{a_{n-1}, a_{n-2}\} = \max\{a_n, a_{n-1}\}$.*

Důkaz (T3): Položme $b_n = \max\{a_n, a_{n-1}\}$ pro každé $n \geq 4$. Podle (T2) pro každé $n \geq 5$ platí $a_n \leq b_{n-1}$, což spolu s triviální rovností $a_{n-1} \leq b_{n-1}$ vede k závěru, že $b_n \leq b_{n-1}$. Posloupnost přirozených čísel $b_4, b_5, b_6, \dots$ je tedy nerostoucí, proto je od jistého členu, řekněme b_{n_0} , konstantní. Důkaz (T3) je hotov.

(T4) *Pro každé $n \geq n_0$, kde n_0 je index z (T3), platí implikace: jestliže $a_n > a_{n+1}$, pak $a_n - a_{n+1} = 2$.*

Důkaz (T4): Pokud $a_n > a_{n+1}$ pro některé $n \geq n_0$, pak $a_n \geq a_{n+1} + 2$ podle (T1) a z rovnosti $\max\{a_{n+1}, a_n\} = \max\{a_{n+2}, a_{n+1}\}$ obsažené v (T3) vyplývá, že $a_n = a_{n+2}$, což podle (1) znamená, že $a_n = 1 + (a_n + a_{n+1})^*$. Číslo $a_n - 1$ je tedy dělitelem (většího) čísla $a_n + a_{n+1}$, a tak platí nerovnost $a_n + a_{n+1} \geq 2(a_n - 1)$, odkud $a_n \leq a_{n+1} + 2$. Protože platí i obrácená nerovnost (viz výše), je důkaz (T4) ukončen.

S pomocí tvrzení (T3) a (T4) teď dokončíme řešení úlohy. (Ukáže se, že všechny možné periodické skupiny členů lze vyčíst z naší tabulky.) Uvažujme i nadále libovolnou zkoumanou posloupnost $\{a_n\}$ a jí příslušný index n_0 z (T3). Mohou nastat dva případy:

- (i) nerovnost $a_n \leq a_{n+1}$ platí pro každé $n \geq n_0$,
- (ii) pro některé $n \geq n_0$ platí $a_n > a_{n+1}$.

V případě (i) z (T3) plyne indukcí, že $a_n = a_{n_0}$ pro každé $n \geq n_0$. Možnou hodnotu $c = a_{n_0}$ najdeme podle (1) z rovnosti $c = 1 + (2c)^*$. Protože $(2c)^* = c^*$, dostáváme $c^* = c - 1$. Číslo $c - 1$ je však dělitelem čísla c zřejmě jen pro $c = 2$. Zkoumaná posloupnost je tedy tvaru

$$a_1, a_2, \dots, a_{n_0-1}, 2, 2, 2, \dots \quad (3)$$

V případě (ii) z nerovnosti $a_n > a_{n+1}$ podle (T4) plyne $a_n = 2d$ a $a_{n+1} = 2d - 2$ pro vhodné celé $d > 1$ (připomeňme, že a_n je sudé podle (T1)). Podle předpisu (1) pak dostáváme

$$\begin{aligned} a_{n+2} &= 1 + (2d + 2d - 2)^* = 1 + (2d - 1) = 2d, \\ a_{n+3} &= 1 + (2d - 2 + 2d)^* = 1 + (2d - 1) = 2d, \\ a_{n+4} &= 1 + (2d + 2d)^* = 1 + d^*. \end{aligned}$$

Pro $d > 1$ ovšem platí $2d > 1 + d \geq 1 + d^*$, a tak $a_{n+3} > a_{n+4}$. To opět podle (T4) znamená, že $a_{n+3} - a_{n+4} = 2$, neboli $(2d) - (1 + d^*) = 2$, odkud $2d - 3 = d^* \leq d$, takže $d \leq 3$. V případě $d = 2$ je zkoumaná posloupnost tvaru

$$a_1, a_2, \dots, a_{n-1}, 4, 2, 4, 4, 2, 4, 4, 2, \dots, \quad (4)$$

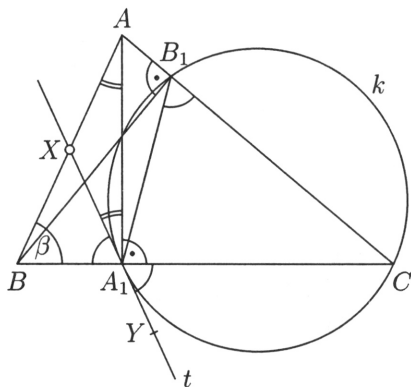
zatímco pro $d = 3$ má tvar

$$a_1, a_2, \dots, a_{n-1}, 6, 4, 6, 6, 4, 6, 6, 4, \dots \quad (5)$$

Dokázali jsme, že *každá* posloupnost ze zadání úlohy je jednoho z tvarů (3), (4), (5). Odtud již plyne, že periodické od prvního členu jsou právě ty posloupnosti, které mají dvojici prvních členů (a_1, a_2) rovnu jedné ze sedmi dvojic $(2, 2)$, $(2, 4)$, $(4, 2)$, $(4, 4)$, $(4, 6)$, $(6, 4)$ a $(6, 6)$.

A – I – 3

Označme k kružnici opsanou trojúhelníku CA_1B_1 . V první části řešení ukážeme, že bod M z textu úlohy je středem strany AB . Protože trojúhelník ABC je ostroúhlý, leží paty A_1 , B_1 příslušných výšek uvnitř odpovídajících stran. S ohledem na symetrii dané situace stačí uvažovat



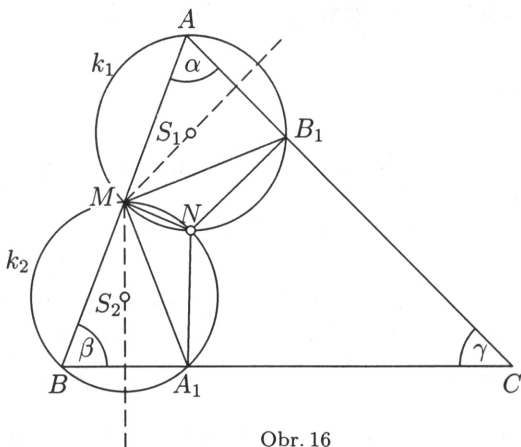
Obr. 15

jen tečnu t ke kružnici k sestrojenou v bodě A_1 , označit X její průsečík s přímkou AB a dokázat rovnost $|AX| = |BX|$ (obr. 15).

Označme ještě Y libovolný vnitřní bod polopřímky opačné k polopřímce A_1X . Protože jsou oba úhly AA_1B a BB_1A pravé, je čtyřúhelník ABA_1B_1 tětíkový, a tak platí $|\sphericalangle AB_1A_1| = 180^\circ - |\sphericalangle ABA_1| = 180^\circ - \beta$, kde jako obvykle $\beta = |\sphericalangle ABC|$. Proto má obvodový úhel A_1B_1C v kružnici k nad tětívou A_1C velikost $|\sphericalangle A_1B_1C| = 180^\circ - |\sphericalangle AB_1A_1| = 180^\circ - (180^\circ - \beta) = \beta$, stejnou velikost má i příslušný úsekový úhel YA_1C .² Protože úhly XA_1B a YA_1C jsou vrcholové, dohromady dostáváme $|\sphericalangle XA_1B| = |\sphericalangle YA_1C| = |\sphericalangle A_1B_1C| = \beta$ (tyto shodné úhly jsou na obr. 15 vyznačeny obloučky). Zároveň platí i $|\sphericalangle XA_1A| = |\sphericalangle XAA_1| = 90^\circ - \beta$. Zjistili jsme, že tečna t protne přímkou AB v takovém bodě X , pro který jsou trojúhelníky BA_1X a A_1AX rovnoramenné, tj. $|BX| = |A_1X| = |AX|$.

Dokázali jsme, že bod M (průsečík tečen ke kružnici k s body dotyku A_1 a B_1) splývá se středem strany AB . Označme nyní k_1 a k_2 kružnice opsané po řadě trojúhelníkům AMB_1 a BMA_1 a S_1, S_2 jejich středy (obr. 16). Jedním průsečíkem kružnic k_1 a k_2 je bod M , druhý průsečík označme N . Protože body S_1, S_2 leží v polorovině ABC , leží v ní i bod N , neboť je souměrně sdružený s M podle středné S_1S_2 . Naší úlohou je dokázat, že bod N leží na jedné kružnici s body A_1, B_1 a C .

² K pojmu úsekového úhlu a jeho shodnosti s obvodovým úhlem viz S. Horák: *Kružnice*, ŠMM 16, MF, Praha 1966, str. 3–7.



Obr. 16

Jak už víme, je trojúhelník BA_1M rovnoramenný, a protože

$$|\sphericalangle BB_1M| = 90^\circ - \alpha < \beta = |\sphericalangle BA_1M|$$

(tato nerovnost je ekvivalentní tomu, že $\gamma < 90^\circ$), leží bod B_1 vně kružnice k_2 . To znamená, že kružnice k_2 musí protínat kružnici k_1 v tom jejím oblouku nad tětivou MB_1 , který odpovídá obvodovému úhlu $180^\circ - \alpha$. Analogicky zjistíme, že bod A_1 leží vně kružnice k_1 , takže průsečík N leží na oblouku kružnice k_2 příslušného tětivě MA_1 a obvodovému úhlu $180^\circ - \beta$. Protože zároveň

$$\begin{aligned} |\sphericalangle B_1NM| + |\sphericalangle A_1NM| &= (180^\circ - \alpha) + (180^\circ - \beta) = \\ &= 360^\circ - (\alpha + \beta) = 180^\circ + \gamma > 180^\circ, \end{aligned}$$

musí bod N ležet uvnitř trojúhelníku A_1B_1M (přímka A_1B_1 tedy oděluje body C a N). Protože $\alpha + \beta + \gamma = 180^\circ$ (kde $\gamma = |\sphericalangle A_1CB_1|$), plyne odtud, že ve čtyřúhelníku A_1CB_1N je součet vnitřních úhlů u protilehlých vrcholů C a N roven 180° , a tak je tento čtyřúhelník skutečně tětivový.

A - I - 4

Sečtením všech tří daných nerovnic dostaneme nerovnost

$$(\sin x + \cos x) + (\sin y + \cos y) + (\sin z + \cos z) \geq 3\sqrt{2}. \quad (1)$$

Na druhou stranu pro každé reálné číslo t platí³

$$\begin{aligned}\sin t + \cos t &= \sqrt{2} \left(\frac{\sqrt{2}}{2} \sin t + \frac{\sqrt{2}}{2} \cos t \right) = \\ &= \sqrt{2} \left(\sin t \cos \frac{\pi}{4} + \cos t \sin \frac{\pi}{4} \right) = \\ &= \sqrt{2} \sin \left(t + \frac{\pi}{4} \right) \leq \sqrt{2},\end{aligned}\tag{2}$$

neboť $\sin \left(t + \frac{1}{4}\pi \right) \leq 1$; přitom rovnost v druhém řádku (2) nastane, právě když $t + \frac{1}{4}\pi = \frac{1}{2}\pi + 2k\pi$, neboli $t = \frac{1}{4}\pi + 2k\pi$ pro některé celé číslo k . Sečtením tří odhadů (2) pro $t = x$, $t = y$ a $t = z$ dostaneme nerovnost

$$(\sin x + \cos x) + (\sin y + \cos y) + (\sin z + \cos z) \leq 3\sqrt{2}.\tag{3}$$

Zdůrazněme, že nerovnost (3) platí pro *libovolnou trojici* reálných čísel (x, y, z) , zatímco opačná nerovnost (1) platí pro *každé řešení* původní soustavy. Tak zjišťujeme, že nerovnost (1) může být splněna jedině jako rovnost, což se podle předchozího stane, právě když čísla x, y, z budou tvaru

$$x = \frac{\pi}{4} + 2k_1\pi, \quad y = \frac{\pi}{4} + 2k_2\pi, \quad z = \frac{\pi}{4} + 2k_3\pi \quad (k_1, k_2, k_3 \in \mathbb{Z}).$$

Dosazením do původní soustavy se snadno přesvědčíme, že každá taková trojice čísel (x, y, z) je skutečně řešením, platí pro ni totiž

$$\sin x = \cos x = \sin y = \cos y = \sin z = \cos z = \frac{\sqrt{2}}{2}.$$

(Zmínka o zkoušce byla nutná, protože nerovnost (1) je pouze *důsledkem* zadané soustavy: nemohli jsme vyloučit, že pro některou trojici čísel (x, y, z) platí (1), avšak neplatí některá ze tří původních nerovností.)

Poznámka. Nerovnost (2) můžeme snadno získat z nerovnosti mezi aritmetickým a kvadratickým průměrem:

$$\sin t + \cos t \leq |\sin t| + |\cos t| \leq 2\sqrt{\frac{\sin^2 t + \cos^2 t}{2}} = \sqrt{2}.$$

³ Provedeme úpravu, kterou běžně používáme při řešení rovnic typu $a \sin t + b \cos t = c$. Jiná možnost je použít nerovnost mezi aritmetickým a kvadratickým průměrem, viz poznámku na konci řešení.

K řešení úlohy lze místo (1) využít i jiných důsledků dané soustavy. Přepíšme například první z daných nerovnic do tvaru $\sin x \geq \sqrt{2} - \cos y$. Platí-li tato nerovnost, platí i umocněná nerovnost $\sin^2 x \geq (\sqrt{2} - \cos y)^2$, neboť $\sqrt{2} - \cos y > 0$ pro každé $y \in \mathbb{R}$. Sečtením tří nerovností

$$\sin^2 x \geq (\sqrt{2} - \cos y)^2, \quad \sin^2 y \geq (\sqrt{2} - \cos z)^2, \quad \sin^2 z \geq (\sqrt{2} - \cos x)^2$$

dostaneme po úpravách nerovnost

$$0 \geq (1 - \sqrt{2} \cos x)^2 + (1 - \sqrt{2} \cos y)^2 + (1 - \sqrt{2} \cos z)^2,$$

ze které už plyne vše potřebné.

Řešení úlohy můžeme zahájit ještě jedním způsobem. Umocněme nejříve každou ze tří daných nerovnic na druhou (jde o důsledkovou úpravu, neboť menší [pravá] strana nerovnice je kladná) a pak je sečtěme. Po snadné úpravě vyjde nerovnost

$$2 \sin x \cos y + 2 \sin y \cos z + 2 \sin z \cos x \geq 3.$$

O obecné platnosti opačné nerovnosti se přesvědčíme tak, že každý ze tří členů na levé straně odhadneme shora pomocí „klasické“ nerovnosti $2uv \leq u^2 + v^2$ (jež je ostrá v případě $u \neq v$):

$$\begin{aligned} 2 \sin x \cos y + 2 \sin y \cos z + 2 \sin z \cos x &\leq \\ &\leq (\sin^2 x + \cos^2 y) + (\sin^2 y + \cos^2 z) + (\sin^2 z + \cos^2 x) = 3. \end{aligned}$$

(Tak odvodíme nutné podmínky $\sin x = \cos y$, $\sin y = \cos z$, $\sin z = \cos x$, za kterých je řešení původní soustavy již triviální úlohou.)

A – I – 5

Vypíšeme přehled všech výsledků dosazení, které budeme dále potřebovat (písmeno a značí libovolné reálné číslo):

$$\begin{array}{ll} \text{(D1)} & x = 0, y = 0: \quad f(0) = (f(0))^2, \\ \text{(D2)} & x = 0, y = a: \quad a^2 + 2f(0) = f(a) \cdot (f(a) + f(0)), \\ \text{(D3)} & x = a, y = a: \quad a^2 + f(a^2) = f(a) \cdot f(2a), \\ \text{(D4)} & x = a, y = -a: \quad 2a^2 + 2f(-a^2) = f(0) \cdot (f(a) + f(-a)), \\ \text{(D5)} & x = 1, y = 1: \quad 1 = f(1) \cdot (f(2) - 1). \end{array}$$

Z (D1) plyne, že číslo $f(0)$ je řešením rovnice $t = t^2$, je to tedy číslo $t = 0$ nebo $t = 1$. Diskutujeme obě možnosti odděleně.

(i) Příklad $f(0) = 0$. Dosadíme-li $f(0) = 0$ do (D2), dostaneme $a^2 = (f(a))^2$, odkud $f(a) = a$ nebo $f(a) = -a$ pro každé $a \in \mathbb{R}$. Zdůrazníme, že zatím nevíme, zda pro všechna x platí stejný z obou předpisů $f(x) = x$ resp. $f(x) = -x$ (víme jen, že existuje $A \subset \mathbb{R}$ tak, že $f(x) = x$ pro každé $x \in A$ a $f(x) = -x$ pro každé $x \in \mathbb{R} \setminus A$).

Ukažme nejprve, že předpis $f(x) = x$ platí pro všechna nekladná x : dosadíme-li $f(0) = 0$ do (D4), dostaneme rovnost $f(-a^2) = -a^2$, takže $f(x) = x$ pro všechna ta reálná čísla x , která se dají zapsat ve tvaru $x = -a^2$ s vhodným $a \in \mathbb{R}$, a to jsou všechna nekladná x . Nyní zdůvodníme, proč $f(x) = x$ rovněž pro všechna kladná x : kdyby totiž pro některé $a \neq 0$ neplatilo $f(a^2) = a^2$, platilo by podle předchozího odstavce $f(a^2) = -a^2$, a tak bychom podle (D3) měli $f(a)f(2a) = 0$, což je ale ve sporu s tím, že obě čísla $f(a)$ a $f(2a)$ jsou nenulová (jsou to čísla $\pm a$ resp. $\pm 2a$, předpoklad ale byl, že $a \neq 0$). Tím je dokázáno, že $f(x) = x$ pro každé x . Dosazením se snadno ověří, že taková funkce je skutečně řešením naší úlohy.

(ii) Příklad $f(0) = 1$. Po dosazení $f(0) = 1$ do (D2) dostaneme pro hodnotu $f(a)$ kvadratickou rovnici $(f(a))^2 + f(a) - a^2 - 2 = 0$. Jejím řešením zjistíme, že

$$f(a) \in \left\{ \frac{-1 + \sqrt{4a^2 + 9}}{2}, \frac{-1 - \sqrt{4a^2 + 9}}{2} \right\} \quad \text{pro každé } a \in \mathbb{R}.$$

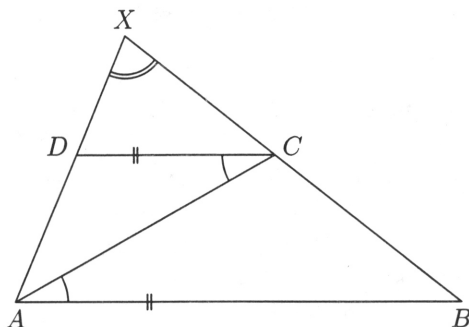
Speciálně pro $a = 1$ a pro $a = 2$ vychází

$$f(1) \in \left\{ \frac{-1 + \sqrt{13}}{2}, \frac{-1 - \sqrt{13}}{2} \right\} \quad \text{a} \quad f(2) \in \{2, -3\},$$

odkud plyne, že číslo $f(1)$ je iracionální, zatímco číslo $f(2) - 1$ je racionální a různé od nuly. Proto je součin $f(1) \cdot (f(2) - 1)$ iracionální, což je ve sporu s (D5). Hledaná funkce f splňující podmínku $f(0) = 1$ tudíž neexistuje.

Odpověď: Danou funkcionální rovnici splňuje jediná funkce $f: \mathbb{R} \rightarrow \mathbb{R}$, a to funkce určená předpisem $f(x) = x$ pro každé $x \in \mathbb{R}$.

Rovnost ze zadání přepíšme jako $|AB| : |CA| = |AC| : |CD|$. Tato úměra spolu s rovností střídavých úhlů BAC a ACD (obr. 17) znamená, že



Obr. 17

trojúhelníky BAC a ACD jsou podobné podle věty *sus*. Protože strany BC a AD si v této podobnosti odpovídají a podle zadání platí $|BC| : |AD| = 3 : 2$, platí rovněž $|AB| : |CA| = 3 : 2$ a $|AC| : |CD| = 3 : 2$. Vynásobením posledních dvou rovností dostaneme $|AB| : |CD| = 9 : 4$, takže základna AB lichoběžníku $ABCD$ je delší než základna CD . Proto se protínají polopřímky AD a BC , jejich průsečík označme X . Podle zadání má úhel CXD velikost 75° nebo 105° .

Trojúhelníky ABX a DCX jsou podobné podle věty *uu* a podle předchozího platí $|AB| = \frac{9}{4}|DC|$, proto rovněž $|AX| = \frac{9}{4}|DX|$. Dosadíme-li sem za $|AX|$ součet $|AD| + |DX|$, vyjde $|DX| = \frac{4}{5}|AD| = 2,4 \text{ cm}$. Analogicky $|CX| = \frac{4}{5}|BC| = 3,6 \text{ cm}$.

Konstrukce:

1. Trojúhelník CDX : $|DX| = 2,4 \text{ cm}$, $|CX| = 3,6 \text{ cm}$, $\sphericalangle DXC \in \{75^\circ, 105^\circ\}$,
2. bod A : A leží na polopřímce opačné k DX , $|AD| = 3 \text{ cm}$,
3. bod B : B leží na polopřímce opačné k CX , $|BC| = 4,5 \text{ cm}$.

Nyní je třeba ukázat, že takto sestrojený čtyřúhelník $ABCD$ má všechny požadované vlastnosti (tj. provést *důkaz správnosti* konstrukce). Předně body 2 a 3 konstrukce zaručují, že strany BC a AD mají předepsané délky. Podle bodu 1 svírají přímky BC a AD předepsaný úhel

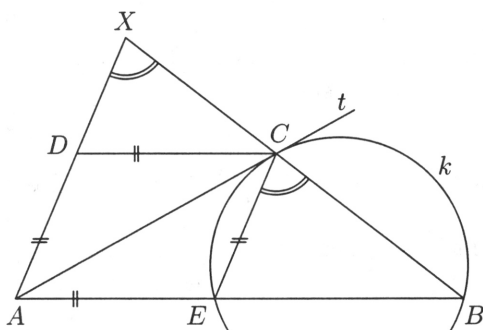
a platí

$$|AX| = |AD| + |DX| = 5,4 \text{ cm} \quad \text{a} \quad |BX| = |BC| + |CX| = 8,1 \text{ cm},$$

a tak je poměr $|AX| : |BX|$ roven $2 : 3$ stejně jako poměr $|DX| : |CX|$. Trojúhelníky ABX a DCX jsou tudíž stejnolehle, proto jsou úsečky AB a CD rovnoběžné. Tak jsme ověřili, že $ABCD$ je lichoběžník se základnami AB , CD . Z rovnosti poměrů $|AX| : |BX| = 2 : 3 = |CX| : |AX|$ plyne podobnost trojúhelníků ABX a CAX (se společným úhlem při vrcholu X), takže úhly ABX a CAX (neboli úhly ABC a CAD) jsou shodné. Protože jsou shodné rovněž (střídavé) úhly BAC a ACD , podle věty *uu* zjišťujeme, že jsou podobné trojúhelníky ABC a CAD . Proto platí $|AB| : |CA| = |AC| : |CD|$, tudíž $|AB| \cdot |CD| = |AC|^2$. (Poslední rovnost lze dokázat také tak, že délky úseček AB , CD a AC vyjádříme podle kosinové věty pro trojúhelníky ABX , CDX resp. ACX .) Všechny požadované vlastnosti sestrojeného čtyřúhelníku $ABCD$ jsou tak ověřeny.

Druhé řešení. Zadaná rovnost $|AB| \cdot |CD| = |AC|^2$ evokuje otázku, zda nelze úlohu řešit pomocí známého tvrzení o mocnosti bodu ke kružnici. Ukažme, že je tomu skutečně tak.

Předpokládejme nejprve, že $|AB| > |CD|$ a označme X průsečík polopřímek AD a BC . Vhodným bodem E základny AB doplníme trojúhelník ACD na rovnoběžník $AECD$ (obr. 18).⁴ Všimněme si, že v trojúhelníku BCE známe délky stran BC , EC ($|EC| = |AD|$) a velikost úhlu BCE



Obr. 18

⁴ To je dosti obvyklý obrat v situaci, kdy jsou dány délky ramen lichoběžníku a úhel, který tato ramena svírají.

($|\sphericalangle BCE| = |\sphericalangle CXD|$). Opišme tomuto trojúhelníku kružnici a označme ji k . Protože $|AE| = |CD|$, lze zadanou rovnost $|AB| \cdot |CD| = |AC|^2$ zapsat jako $|AB| \cdot |AE| = |AC|^2$. To znamená, že přímka AC je tečnou ke kružnici k . Tím je rozbor případu $|AB| > |CD|$ ukončen. V případě $|AB| < |CD|$ stačí v rozboru provést dvě změny: bod X je průsečíkem polopřímek DA a CB , bod E leží na polopřímce opačné k BA (nikoliv na základně AB).

Konstrukce:

1. Trojúhelník BCE : $|BC| = 4,5 \text{ cm}$, $|EC| = 3 \text{ cm}$, $|\sphericalangle BCE| \in \{75^\circ, 105^\circ\}$,
2. kružnice k opsaná trojúhelníku BCE ,
3. tečna t ke kružnici k v bodě C ,
4. bod A : $A \in BE \cap t$,
5. bod D : $AECD$ je rovnoběžník.

(Protože dle zadání platí $|BC| > |EC|$, padne bod A při konstrukci podle bodu 4 na polopřímku opačnou k EB . Bod E bude tedy náležet úsečce AB , takže nastane případ $|AB| > |CD|$.)

Při důkazu správnosti konstrukce opět využijeme mocnost bodu ke kružnici (v opačném „směru“, než tomu bylo v rozboru): protože je přímka AC tečnou kružnice k , platí rovnost $|AB| \cdot |AE| = |AC|^2$. Zbytek důkazu je triviální.

A – S – 1

Sečtením všech tří nerovnic dostaneme nerovnost

$$75 + 2x^2 + 3y^2 + 4z^2 \leq 16x + 18y + 16z,$$

z níž po „doplnění na druhé mocniny“ vychází

$$2(x - 4)^2 + 3(y - 3)^2 + 4(z - 2)^2 \leq 0.$$

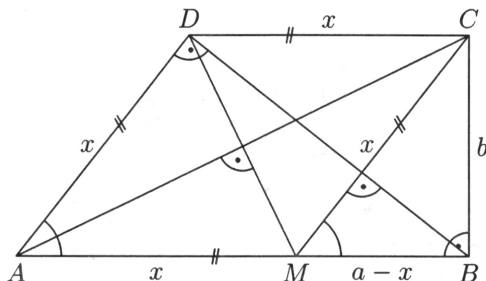
Tato nerovnost, jež je důsledkem dané soustavy nerovnic, zřejmě platí jedině tehdy, když jsou základy všech tří dvojmocí na levé straně nerovnosti rovny nule, tedy když $x = 4$, $y = 3$, $z = 2$. Daná soustava má proto (při zvoleném p) nejvýše jedno řešení, a to právě vypsanou trojici čísel. Zjistíme nyní, pro kterou hodnotu parametru p se skutečně jedná o řešení. Po dosazení hodnot $x = 4$, $y = 3$, $z = 2$ do dané soustavy dostaneme trojici nerovností

$$57 \leq 59 - p, \quad 52 \leq 52, \quad 41 \leq 39 + p.$$

Z první nerovnosti vychází podmínka $p \leq 2$, ze třetí podmínka $p \geq 2$. Číslo $p = 2$ je tedy jediná hodnota p , pro kterou má daná soustava v oboru reálných čísel řešení.

A – S – 2

Úsečky MC a AD jsou rovnoběžné (obě jsou totiž kolmé k úsečce BD , obr. 19); protože jsou rovnoběžné i úsečky AM a DC , je čtyřúhelník



Obr. 19

$AMCD$ rovnoběžník. Je to dokonce kosočtverec, neboť jeho úhlopříčky jsou dle zadání navzájem kolmé. Označme proto $x = |CD| = |DA| = |AM| = |MC|$, zřejmě $a > x$. Pak $|MB| = a - x$ a ze shodnosti souhlasných úhlů DAM a CMB plyne podobnost pravoúhlých trojúhelníků ABD a MCB , takže platí úměra $|AD| : |AB| = |MB| : |MC|$, neboli $x : a = (a - x) : x$. Odtud pro neznámou délku x vychází kvadratická rovnice $x^2 + ax - a^2 = 0$, která má jediné kladné řešení $x = \frac{\sqrt{5} - 1}{2} a$.

Tak jsme vypočetli (shodné) délky základny CD a ramena AD daného lichoběžníku; zbývá určit délku ramena BC . Podle Pythagorovy věty pro trojúhelník CMB dostáváme

$$\begin{aligned} |BC| &= \sqrt{|MC|^2 - |MB|^2} = \sqrt{x^2 - (a - x)^2} = \sqrt{a(2x - a)} = \\ &= a\sqrt{\sqrt{5} - 2}, \end{aligned}$$

neboť $2x - a = a(\sqrt{5} - 2)$.

A – S – 3

Z vyjádření $\overline{abcd} = 100 \cdot \overline{ab} + \overline{cd}$ plyne, že podmínky dělitelnosti čísla $\overline{ab}$ a $\overline{cd}$ jsou splněny, právě když $\overline{cd} \mid 100 \cdot \overline{ab}$ a $\overline{ab} \mid \overline{cd}$, tedy právě když $\overline{cd} = k \cdot \overline{ab}$, kde přirozené číslo k je některý dělitel čísla 100. Protože obě čísla $\overline{ab}$ a $\overline{cd}$ jsou dle zadání lichá a dvojciferná, plyne odtud, že buď $k = 1$, nebo $k = 5$. Rozlišíme oba případy, přitom s ohledem na vyjádření $\overline{abcd} = 10 \cdot \overline{bc} + (1000a + d)$ budeme místo podmínky $\overline{bc} \mid \overline{abcd}$ zkoumat ekvivalentní podmínku

$$\overline{bc} \mid (1000a + d). \quad (\text{P})$$

(Tato úprava není nutná, jen poněkud zjednodušuje další zápisy.)

a) Je-li $\overline{cd} = \overline{ab}$, platí $c = a$ a $d = b$, takže podmínka (P) se zapíše ve tvaru $(10b + a) \mid (1000a + b)$. Protože

$$1000 \cdot (10b + a) - (1000a + b) = 9999 \cdot b = 11 \cdot 9 \cdot 101 \cdot b$$

a 101 je prvočíslo (tudíž je s číslem $10b + a$ nesoudělné), dostáváme ekvivalentní podmínku $(10b + a) \mid (11 \cdot 9b)$. Odtud s ohledem na zřejmou nerovnost $10b + a > 9b$ plyne, že číslo $10b + a$ má číslo 11 ve svém rozkladu na prvočinitele. Podmínka $11 \mid (10b + a)$ je však splněna, jen když se číslice a a b rovnají, pak by však z rovnosti $\overline{cd} = \overline{ab}$ vyplývalo, že číslo $\overline{abcd}$ má *všechny* číslice stejné. O takových číslech podle zadání úlohy neuvažujeme.

b) Z rovnosti $\overline{cd} = 5 \cdot \overline{ab}$ ihned určíme (liché) cifry $a = 1$ a $d = 5$, po jejich dosazení po úpravě vyjde rovnost $b = 2c - 9$, takže jsou tři možnosti:

$$c = 5 \text{ a } b = 1, \quad c = 7 \text{ a } b = 5, \quad c = 9 \text{ a } b = 9.$$

Podmínka (P) má nyní tvar $\overline{bc} \mid 1005$, z čísel 15, 57 a 99 je však pouze číslo 15 dělitelem čísla 1005 ($1005 = 3 \cdot 5 \cdot 67$), proto nutně $b = 1$ a $c = 5$.

Odpověď: Hledané číslo je jediné, a to 1155.

A – II – 1

Předpokládejme, že parametr n je přirozené číslo, a řešme danou soustavu v oboru reálných čísel. Levá strana první rovnice je rovna $(x^2 + y^2)(x + y)$, a tak pro čísla $s = x + y$ a $t = x^2 + y^2$ platí $t \cdot s = n$ a $t + s = n + 1$. Čísla s , t jsou tedy kořeny kvadratické rovnice $w^2 - (n + 1)w + n = 0$.

Z rozkladu $w^2 - (n+1)w + n = (w-n)(w-1)$ vidíme, že $\{s, t\} = \{1, n\}$. Dvojice (x, y) je tedy řešením původní soustavy, právě když je řešením jedné ze soustav

$$\begin{cases} x + y = 1, \\ x^2 + y^2 = n \end{cases} \quad \text{nebo} \quad \begin{cases} x + y = n, \\ x^2 + y^2 = 1. \end{cases} \quad (1)$$

V prvním případě jsou čísla x, y kořeny kvadratické rovnice $z^2 + (1 - z)^2 = n$. Jejím řešením dostaneme

$$\{x, y\} = \left\{ \frac{1 + \sqrt{2n-1}}{2}, \frac{1 - \sqrt{2n-1}}{2} \right\}. \quad (2)$$

Podobně z druhé soustavy v (1) plyne, že čísla x, y jsou kořeny kvadratické rovnice $z^2 + (n - z)^2 = 1$. Její diskriminant $D = 4(2 - n^2)$ je nezáporný jedině pro $n = 1$ (připomeňme, že n je přirozené), pak jsou ovšem obě soustavy v (1) totožné, takže žádné další řešení kromě (2) neexistuje.

Zjistili jsme, že pro každé přirozené číslo n jsou všechna řešení původní soustavy v oboru *reálných* čísel popsána vztahem (2). Jsou to celá čísla, právě když je hodnota $2n - 1$ druhou mocninou (lichého) přirozeného čísla. Je-li číslo n čtyřmístné, pak $n \geq 1000$, a tak $2n - 1 \geq 1999$. Protože $44^2 = 1936$ a $45^2 = 2025$, hledané číslo n určíme z rovnice $2n - 1 = 2025$. Zřejmě $n = 1013$.

Poznamenejme, že v první části řešení můžeme postupovat i takto: do první rovnice $(x^2 + y^2)(x + y) = n$ lze dosadit za první činitel vyjádření $x^2 + y^2 = n + 1 - (x + y)$ z druhé rovnice. Tak získáme pro neznámou $s = x + y$ kvadratickou rovnici $(n + 1 - s)s = n$, jejíž kořeny jsou $s_1 = 1$ a $s_2 = n$.

A - II - 2

Předpokládejme, že s a t jsou (pevná) čísla požadované vlastnosti. Graf funkce f je sjednocením grafů funkcí f_1 a f_2 , které jsou určeny vzorci

$$f_1(x) = \frac{x^2 - 4x + s}{(1-t)x + (t+7)}, \quad f_2(x) = \frac{x^2 - 4x + s}{(t+1)x + (7-t)} \quad (1)$$

a mají definiční obory $D(f_1) = (-\infty, 1)$, $D(f_2) = (1, \infty)$ (polopřímky bez jakýchkoliv vyloučených bodů, neboť hodnota $f(x)$ dle popisu grafu f

existuje pro každé $x \in \mathbb{R}$). Čísla s a t určíme z podmínky, že grafy funkcí f_1 a f_2 jsou polopřímky (takže jde o lineární funkce). Zřejmě platí $t \neq \pm 1$ (jinak by graf jedné z funkcí f_1, f_2 byl částí paraboly), proto můžeme lineární funkce ze jmenovatelů zlomků v (1) zapsat ve tvaru

$$(1-t)x + (t+7) = (1-t)(x-x_1), \quad \text{kde} \quad x_1 = \frac{t+7}{t-1}, \quad (2a)$$

a

$$(t+1)x + (7-t) = (t+1)(x-x_2), \quad \text{kde} \quad x_2 = \frac{t-7}{t+1}. \quad (2b)$$

Výhodu těchto rozkladů oceníme při vyjadřování podmínky, že obě funkce f_1 a f_2 jsou lineární. Předtím však poznamenejme, že hodnota $f_1(x)$ existuje pro každé $x \leq 1$ a hodnota $f_2(x)$ pro každé $x \geq 1$, právě když čísla x_1, x_2 z rozkladů (2) splňují podmínku

$$x_2 < 1 < x_1. \quad (3)$$

Vzorce (1) určují lineární funkce f_1 a f_2 , právě když je kvadratický mnohočlen $x^2 - 4x + s$ dělitelný (beze zbytku) každým z lineárních mnohočlenů $(x-x_1)$ a $(x-x_2)$. Protože však podle (3) platí $x_1 \neq x_2$, lze podmínku z předchozí věty vyjádřit rovností mnohočlenů

$$x^2 - 4x + s = (x-x_1)(x-x_2). \quad (4)$$

Poznamenejme, že za podmínky (4) budou předpisy pro funkce f_1, f_2 tvaru

$$f_1(x) = \frac{x-x_2}{1-t} \quad \text{a} \quad f_2(x) = \frac{x-x_1}{t+1};$$

podmínka (3) zaručí, že polopřímky, které jsou grafy f_1 a f_2 , neleží na téže přímce (kdyby ležely, nebyla by grafem f lomená čára): osu x totiž protne jak polopřímka $y = f_1(x)$ (v bodě $[x_2, 0]$), tak i polopřímka $y = f_2(x)$ (v bodě $[x_1, 0]$).

Podmínka (4) je s ohledem na (2) ekvivalentní s dvojicí rovnic

$$4 = \frac{t+7}{t-1} + \frac{t-7}{t+1} \quad \text{a} \quad s = \frac{t+7}{t-1} \cdot \frac{t-7}{t+1} = \frac{t^2-49}{t^2-1},$$

ze kterých určíme neznámé hodnoty s a t . Úpravou první rovnice vyjde $t^2 = 9$, možné hodnoty t jsou tedy ± 3 ; podle druhé rovnice jim odpovídá stejná hodnota $s = -5$. Je-li $t = 3$, platí podle (2) $x_1 = 5$ a $x_2 = -1$

(podmínka (3) je tehdy splněna), je-li $t = -3$, pak $x_1 = -1$ a $x_2 = 5$ (podmínka (3) splněna není). Řešením úlohy je tedy jediná dvojice $(s, t) = (-5, 3)$.

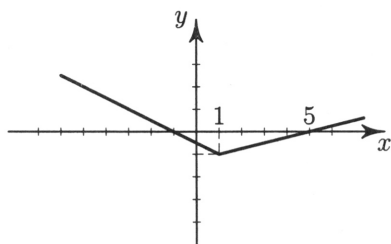
Přestože jsme celé řešení vedli tak, že zkouška nutná není, provedme ji jak pro dvojici $(s, t) = (-5, 3)$, tak pro dvojici $(s, t) = (-5, -3)$. Pro první dvojici vychází

$$f(x) = \frac{x^2 - 4x - 5}{3|x - 1| + x + 7} = \begin{cases} \frac{(x+1)(x-5)}{-2(x-5)} = -\frac{x+1}{2} & (x \leq 1), \\ \frac{(x+1)(x-5)}{4(x+1)} = \frac{x-5}{4} & (x \geq 1), \end{cases}$$

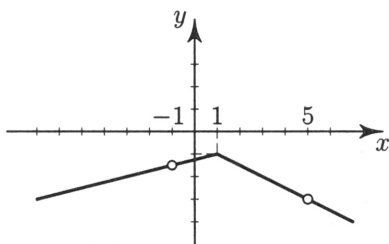
takže opravdu jde o řešení (obr. 20a); dvojici $(s, t) = (-5, -3)$ odpovídá funkce

$$f(x) = \frac{x^2 - 4x - 5}{-3|x - 1| + x + 7} = \begin{cases} \frac{(x+1)(x-5)}{4(x+1)} = \frac{x-5}{4} & (-1 \neq x \leq 1), \\ \frac{(x+1)(x-5)}{-2(x-5)} = -\frac{x+1}{2} & (5 \neq x \geq 1), \end{cases}$$

jejímž grafem je lomená čára bez dvou bodů (obr. 20b), takže dvojici $(s, t) = (-5, -3)$ nelze považovat za řešení.



Obr. 20a



Obr. 20b

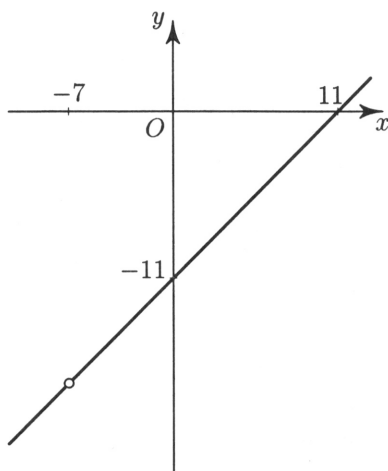
Poznámka. Podmínku linearitu obou funkcí f_1 a f_2 (aniž zavedeme čísla x_1, x_2) můžeme vyjádřit i takto: mnohočlen $x^2 - 4x + s$ je dělitelný jak mnohočlenem $(1 - t)x + (t + 7)$, tak i mnohočlenem $(t + 1)x + (7 - t)$, je tedy dělitelný i jejich součinem, tudíž platí rovnost mnohočlenů

$$((1 - t)x + (t + 7))((t + 1)x + (7 - t)) = (1 - t^2)(x^2 - 4x + s)$$

(koeficient $(1 - t^2)$ se zjistí porovnáním kvadratických členů). Tento závěr je však korektní, jen když jsou oba lineární mnohočleny *nesoudělné*; v našem řešení je tato nesoudělnost zaručena podmínkou (3). Soudělným mnohočlenům ze jmenovatelů zlomků v (1) odpovídá „řešení“ $(s, t) = (-77, 0)$ s příslušnou funkcí

$$f(x) = \frac{x^2 - 4x - 77}{x + 7} = x - 11 \quad (x \in \mathbb{R}, x \neq -7),$$

jejíž graf (obr. 20c) je sice složen ze dvou polopřímek, ale jejich společný počátek do grafu f nepatří (navíc tyto polopřímky svírají přímý úhel, takže jejich sjednocení není lomená čára).



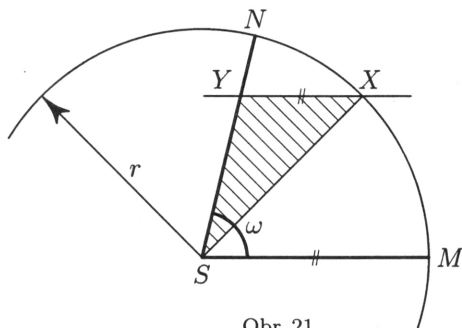
Obr. 20c

A – II – 3

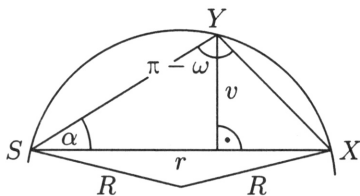
Ve všech řešeních značíme $\omega = |\sphericalangle MSN|$.

Řešení 1. Z rovnoběžnosti přímek SM a XY plyne rovnost $|\sphericalangle SYX| = \pi - \omega$ (obr. 21). Proto se všechny uvažované trojúhelníky SXY shodují nejen v délce strany SX (rovné poloměru r kružnice k) ale i ve velikosti protilehlého vnitřního úhlu SYX . Kružnice opsané všem trojúhelníkům SXY mají tedy též poloměr R (rovný $r/(2 \sin \omega)$), a bod Y leží vždy na tom jejich oblouku, ze kterého je tětiva SX délky r vidět pod úhlem $\pi - \omega$. Výška v ke straně SX trojúhelníku SYX tudíž zřejmě nepřevyšuje

výšku kruhové úseče z obr. 22, přitom je rovna této výšce, právě když platí $|SY| = |XY|$. Proto má ze všech trojúhelníků SXY největší obsah právě ten, který má shodné strany SY a XY . Jeho vnitřní úhel α u vrcholu S je shodný s vnitřním úhlem u vrcholu X , a tak platí $2\alpha + (\pi - \omega) = \pi$, odkud $\alpha = \frac{1}{2}\omega$, což znamená, že polopřímka SX je osou úhlu MSN . Průsečík této osy s kružnicí k proto určuje hledaný bod X .



Obr. 21



Obr. 22

Dodejme, že maximální výšku v trojúhelníku SXY ke straně SX lze určit i jiným postupem (bez úvah o kruhové úseči): označíme-li Y_0 patu výšky z vrcholu Y , $\alpha = |\sphericalangle XSY|$ a $\beta = |\sphericalangle SXY|$, potom platí

$$v \cdot \cotg \alpha + v \cdot \cotg \beta = |SY_0| + |XY_0| = |SX| = r,$$

odkud s ohledem na to, že $\alpha + \beta = \omega$ a že funkce $\cotg$ je v intervalu $(0, \frac{1}{2}\pi)$ konvexní, vychází odhad

$$v = \frac{r}{\cotg \alpha + \cotg \beta} \leq \frac{r}{2 \cotg \frac{\alpha + \beta}{2}} = \frac{r}{2 \cotg \frac{\omega}{2}} = \frac{r}{2} \operatorname{tg} \frac{\omega}{2}.$$

Řešení 2. Označme $p = |XY|$ a $q = |SY|$. Podle kosinové věty pro trojúhelník SXY platí $r^2 = p^2 + q^2 + 2pq \cdot \cos \omega$, neboť $|\sphericalangle SYX| = \pi - \omega$. Vypsanou rovnost upravíme do tvaru $r^2 = (p - q)^2 + 2pq(1 + \cos \omega)$, z něhož už snadno odhadneme velikost součinu pq shora:

$$pq = \frac{r^2 - (p - q)^2}{2(1 + \cos \omega)} \leq \frac{r^2}{2(1 + \cos \omega)},$$

přitom rovnost nastane, právě když $p = q$. To je i podmínka, za které je obsah $\frac{1}{2}pq \sin \omega$ trojúhelníku SXY maximální. Došli jsme tak ke stejnému závěru jako při prvním řešení.

neboli

$$(x - y)^2 \cdot f(x + y) = (x + y)^2 \cdot f(y - x).$$

Dosaďme sem při libovolném $t \in \mathbb{R}$ hodnoty $x = \frac{1}{2}(t - 1)$ a $y = \frac{1}{2}(t + 1)$, které jsou zvoleny tak, aby platilo $x + y = t$ a $y - x = 1$. Dostaneme vztah $f(t) = t^2 \cdot f(1)$ pro každé $t \in \mathbb{R}$. Funkce f je tedy nutně tvaru $f(x) = cx^2$, přitom neznámý koeficient c zjistíme tak, že dosadíme do rovnosti ze zadání (a tak vlastně současně provedeme i *zkoušku*): rovnice $c(x^2 + cy^2)^2 = (x - y)^2 \cdot c(x + y)^2$ je ekvivalentní s rovnicí $c(c + 1)y^2(2x^2 + (c - 1)y^2) = 0$, jež je splněna pro libovolná x a y , právě když $c = 0$ nebo $c = -1$ (např. dosazení $x = y = 1$ vede k podmínce $c(c + 1)^2 = 0$, takže nutně $c \in \{0, -1\}$).

Úloha má právě dvě řešení: nulovou funkci $f_1(x) = 0$ a funkci $f_2(x) = -x^2$.

A – III – 1

Je-li mnohočlen P konstantní, tedy $P(x) = a$, pak číslo a splňuje dle zadání podmínku $a^2 + a = a + a$, takže $a = 0$ nebo $a = 1$. Oba mnohočleny $P(x) = 0$ a $P(x) = 1$ jsou řešením úlohy.

Je-li stupeň n mnohočlenu P kladný, pak $P(x) = ax^n + Q(x)$, kde a je číslo různé od nuly a Q je mnohočlen stupně nejvýše $n - 1$. Porovnáme-li v rovnosti

$$(ax^n + Q(x))^2 + a(-x)^n + Q(-x) = ax^{2n} + Q(x^2) + ax^n + Q(x) \quad (1)$$

koeficienty u nejvyšší mocniny x^{2n} , dostaneme podmínku $a^2 = a$, ze které plyne $a = 1$ (připomeňme, že $a \neq 0$). Rovnost (1) po dosazení hodnoty $a = 1$ upravíme do ekvivalentního tvaru

$$2x^n Q(x) + (Q(x))^2 - Q(x^2) = [1 - (-1)^n]x^n + Q(x) - Q(-x). \quad (2)$$

Připustíme, že mnohočlen Q má kladný stupeň k ($k < n$). Pak na levé straně (2) stojí mnohočlen stupně alespoň $n + k$, což je sporu s tím, že na pravé straně (2) je mnohočlen stupně nejvýše n . Proto je Q konstantní mnohočlen, tedy $Q(x) = b$ pro vhodné číslo b . Po dosazení do (2) dostáváme podmínku $2bx^n + b^2 - b = [1 - (-1)^n]x^n$, která je splněna pro každé x , právě když $2b = 1 - (-1)^n$ a zároveň $b^2 - b = 0$. Pro sudé n vychází jediné $b = 0$ (takže $P(x) = x^n$), pro liché n vyjde $b = 1$ (takže $P(x) = x^n + 1$).

Odpověď: Hledané mnohočleny jsou konstanty 0 a 1, jednočleny x^2 , x^4 , x^6 , ... a dvojčleny $x + 1$, $x^3 + 1$, $x^5 + 1$, ...

Jiné řešení. Přičteme-li $P(-x)$ k oběma stranám dané rovnosti, dostaneme rovnost $(P(x))^2 + 2P(-x) = P(x^2) + P(x) + P(-x)$, na jejíž pravé straně je sudá funkce proměnné x . Proto je sudá i funkce na levé straně: pro každé x platí $(P(x))^2 + 2P(-x) = (P(-x))^2 + 2P(x)$, neboli

$$(P(x) - P(-x)) \cdot (P(x) + P(-x) - 2) = 0.$$

Jeden z obou činitelů na levé straně poslední rovnosti je tedy nulový mnohočlen. Pokud platí identicky $P(x) - P(-x) = 0$, redukuje se rovnost ze zadání úlohy na $(P(x))^2 = P(x^2)$; pokud platí identicky $P(x) + P(-x) - 2 = 0$, pak pro mnohočlen Q definovaný rovností $Q(x) = P(x) - 1$ platí $Q(-x) = -Q(x)$ a dosazením a snadnou úpravou zjistíme, že rovnost ze zadání přejde do tvaru $(Q(x))^2 = Q(x^2)$.

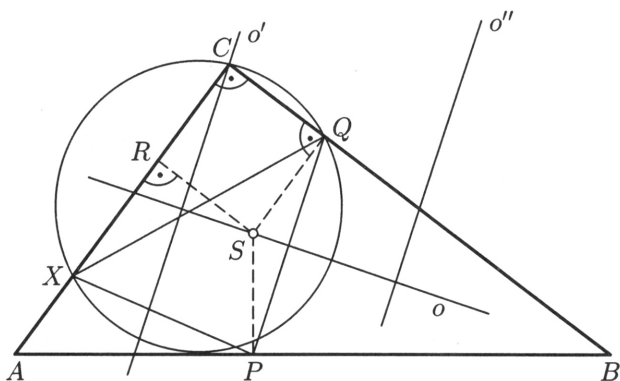
Shrňme-li tedy oba případy, zjistíme, že v každém z nich máme určit mnohočlen R , který je sudou nebo lichou funkcí a splňuje pro každé x rovnost $(R(x))^2 = R(x^2)$. Hledejme taková R nejdříve mezi jednočleny: po dosazení $R(x) = ax^n$ zjistíme, že je buď $a = 0$, nebo $a = 1$ a $n \geq 0$ libovolné. Pripusťme, že R není jednočlen, tedy $R(x) = ax^n + bx^k + S(x)$, kde a, b jsou čísla různá od nuly, $n > k$ a S je nulový mnohočlen nebo mnohočlen stupně nejvýše $k - 1$. Porovnáme-li v rovnosti

$$(ax^n + bx^k + S(x)) \cdot (ax^n + bx^k + S(x)) = ax^{2n} + bx^{2k} + S(x)$$

koefficienty členů s mocninou x^{n+k} , dostaneme rovnost $2ab = 0$, která je ve sporu s tím, že $a \neq 0$ a $b \neq 0$. Proto podmínku $(R(x))^2 = R(x^2)$ splňují pouze mnohočleny R rovné 0, 1, x , x^2 , x^3 , ...

A – III – 2

Označme ještě R bod dotyku s odvěsnou AC a S střed zmíněné kružnice (obr. 24). Protože $SQCR$ je čtverec a bod S leží na ose o úsečky PQ , leží bod C na přímce o' , která je obrazem osy o v otočení kol bodu Q o pravý úhel. Vrchol C proto sestrojíme jako průsečík přímky o' s Thaletovou kružnicí τ nad průměrem QX . Zbytek *konstrukce* je zřejmý.



Obr. 24

Úloha má (pro dané body P, Q, X) jediné řešení. I když osu o můžeme kol bodu Q otočit o pravý úhel dvěma způsoby, jedna z otočených přímk o'' kružnici τ vůbec neprotne; druhá z nich má s kružnicí τ sice dva společné body, ale jednomu z nich odpovídá takový trojúhelník ABC , že místo kružnice vepsané má požadované vlastnosti kružnice *připsaná* přeponě AB (bod Q jejího dotyku s přímkou BC neleží na odvěsně BC , ale na jejím prodloužení za vrchol B).

A – III – 3

Označme $(*)$ danou nerovnici a $K = (-\infty, 0) \cup (1, \infty)$ příslušnou množinu (všech) řešení. Z toho, že 0 patří do K , plyne pro b podmínka $b \geq 0$ a zároveň $\sqrt{b} > -c$. Kdyby však platilo $b > 0$, byl by výraz $\sqrt{2x^2 + ax + b}$ definován v některém okolí bodu $x = 0$ a z (ostré) nerovnosti $(*)$ pro $x = 0$ by plynula její platnost i pro malá kladná čísla x , což je ve sporu s tvarem množiny K . Proto musí být $b = 0$ a z nerovnosti $\sqrt{b} > -c$ plyne podmínka $c > 0$.

Protože $\sqrt{2x^2 + ax + b} = \sqrt{x(2x + a)}$ a protože množina K obsahuje všechna čísla $x > 1$, platí pro taková x nerovnost $2x + a \geq 0$, která znamená, že $a \geq -2$. Protože $1 \notin K$, nerovnost $\sqrt{2 + a} > 1 - c$ neplatí, její levá strana však má díky nerovnosti $a \geq -2$ smysl. Proto naopak platí $\sqrt{2 + a} \leq 1 - c$, odkud plyne podmínka $c \leq 1$. Kdyby platila ostrá nerovnost $\sqrt{2 + a} < 1 - c$, nerovnost $\sqrt{x(2x + a)} < x - c$ by byla splněna nejen pro $x = 1$, ale také pro $x = 1 + \varepsilon$ s dostatečně malým $\varepsilon > 0$, což

je ve sporu s tím, že $1 + \varepsilon \in K$. To znamená, že $\sqrt{2+a} = 1 - c$, odkud $a = (1 - c)^2 - 2 = c^2 - 2c - 1$.

Shrňme výsledky našich úvah: zjistili jsme, že každá vyhovující trojice čísel (a, b, c) je nutně tvaru $(c^2 - 2c - 1, 0, c)$, kde $0 < c \leq 1$. Ukažme nyní, že obráceně každá trojice popsaného tvaru má požadované vlastnosti. Řešme proto v oboru reálných čísel nerovnici

$$\sqrt{x(2x+a)} > x - c, \quad (1)$$

pro pevně zvolené $c \in (0, 1)$ a odpovídající $a = c^2 - 2c - 1$.

Z nerovností $0 < c \leq 1$ a vyjádření $a = (1 - c)^2 - 2$ plyne, že $-2 \leq a < -1$. Pro každé $x \leq 0$ tudíž platí $2x + a < 0$, takže levá strana (1) má smysl a je nezáporná, zatímco pravá strana (1) je pro takové x záporná (neboť $x - c \leq -c < 0$). Proto celý interval $(-\infty, 0)$ patří do množiny řešení (1). Nepatří tam však žádné číslo x z intervalu $(0, -\frac{1}{2}a)$, neboť pro ně nemá smysl levá strana (1). Zbývá tedy vyřešit nerovnici (1) na intervalu $(-\frac{1}{2}a, \infty)$. Zdůvodníme předtím, že pro krajní bod $-\frac{1}{2}a$ platí odhady $c \leq -\frac{1}{2}a \leq 1$. Skutečně, horní odhad okamžitě plyne z toho, že $a \geq -2$, dolní odhad se snadno odvodí ze zřejmé nerovnosti $c^2 \leq 1$:

$$-\frac{1}{2}a = -\frac{1}{2}(c^2 - 2c - 1) = c + \frac{1}{2}(1 - c^2) \geq c.$$

Pro každé $x \in (-\frac{1}{2}a, \infty)$ platí tedy $x \geq c$, a proto jsou obě strany nerovnice (1) nezáporné. Po umocnění obou stran na druhou a snadné úpravě dostaneme ekvivalentní nerovnici $x^2 + (a + 2c)x - c^2 > 0$. Odtud po dosazení $a = c^2 - 2c - 1$ vychází nerovnice $(x - 1)(x + c^2) > 0$, která platí pro právě ta (kladná) čísla $x \in (-\frac{1}{2}a, \infty)$, která jsou větší než 1 (zopakujme, že $-\frac{1}{2}a \leq 1$). Tím je dokázáno, že množinou řešení nerovnice (1) je skutečně množina K ze zadání úlohy.

Odpověď: Hledané trojice jsou $(a, b, c) = (c^2 - 2c - 1, 0, c)$, kde c je libovolné číslo z intervalu $(0, 1)$.

A – III – 4

V žádném slově zřejmě nemohou být čtyři stejná písmena. Maximální možná délka slova uvažovaného jazyka je tedy $3n$ (skupina n trojic stejných písmen za sebou je zřejmě slovo). Zároveň je jasné, že pro $n = 1$ existuje jediné slovo délky 3.

Nechť $n \geq 2$.

1. Každé slovo začíná dvěma stejnými písmeny. Kdyby tomu tak nebylo, měli bychom slovo $AB \dots A \dots A \dots$ začínající dvojicí různých písmen A, B . Další písmeno B se však nemůže vyskytovat mezi prvním a druhým písmenem A (jedno už tam je), ani za třetím písmenem A (dvě A by byla mezi dvěma B). Obě zbývající písmena B by tedy musela být mezi druhým a třetím písmenem A , což také není možné.

2. Vypustíme-li ze slova maximální délky $3n$ tři stejná písmena, dostaneme v jazyce s $n - 1$ písmeny opět slovo maximální délky $3(n - 1)$.

Počet slov maximální délky v jazyce s n písmeny označme p_n . Zjistíme, kolik je slov maximální délky začínajících zvoleným písmenem A . Každé takové slovo začíná dvěma písmeny A , takže třetí písmeno je buď opět A (takových slov je zřejmě tolik, kolik je slov maximální délky obsahujících $n - 1$ písmen, tj. p_{n-1}), nebo písmeno $B \neq A$. Protože po vypuštění všech písmen A dostaneme opět slovo (a to musí začínat, jak už víme, dvěma stejnými písmeny), musí původní slovo začínat skupinou $AABAB$ (možnost $AABB \dots A$ zřejmě nepřichází v úvahu). Takových slov je opět p_{n-1} . Celkem je tedy $2p_{n-1}$ slov maximální délky začínajících zvoleným písmenem A . To znamená, že $p_n = 2np_{n-1}$, odkud snadno plyne, že

$$p_n = 2^{n-1}n!p_1 = 2^{n-1}n!.$$

Nalezený vzorec vyhovuje i pro $n = 1$.

A – III – 5

Z rovnosti úseček, jež v popsané síti odpovídají týmž hranám výsledného čtyřstěnu $ABCD$, dostáváme, že $|AB_1| = |AB_2| = |AB| = b$, $|B_1C_1| = |B_2C_2| = |BC| = c$. Označme S střed úsečky AB_1 a B_3 patu kolmice z bodu D na přímkou B_2C_2 (obr. 25). Trojúhelníky B_1C_1D a B_3C_2D jsou středově souměrné podle bodu D , proto $|B_3C_2| = |B_1C_1| = c$. Protože lichoběžník $C_1AB_2C_2$ je rovnoramenný, je rovnoramenný i trojúhelník B_1AB_2 (vzhledem k předchozím rovnostem je dokonce rovnostranný), a z obdélníku $B_1SB_2B_3$ tak plyne $\frac{1}{2}b = |B_1S| = |B_2B_3| = 2c$, takže $b = 4c$.

Nyní si už jen uvědomíme, že sestavený čtyřstěn $ABCD$ bude mít dvě pravouhlé stěny CDB a ADB s pravými úhly při vrcholu B (obr. 26), což znamená, že hrana BD bude kolmá ke stěně ABC . Přitom výška v trojúhelníku ABC (neboli trojúhelníku AB_2C_2) na stranu BC je zároveň výškou B_2S rovnostranného trojúhelníku B_1AB_2 , takže $v = \frac{1}{2}\sqrt{3}b$ a zároveň $|BD| = |B_1D| = \frac{1}{2}v$. Objem V čtyřstěnu $ABCD$ tedy spočteme

Podmínku (2) lze vyjádřit jako rovnost dvou uspořádaných A -tic

$$[f(p-1), f(p-2), \dots, f(p-A)] = [f(q-1), f(q-2), \dots, f(q-A)],$$

které jsou, jak již víme, sestaveny výhradně z čísel 0 a 1. Ze dvou různých prvků lze ale sestavit pouze 2^A různých A -tic, takže například v následující skupině A -tic

$$\{[f(x-1), f(x-2), \dots, f(x-A)]: x = 0, 1, \dots, 2^A\}$$

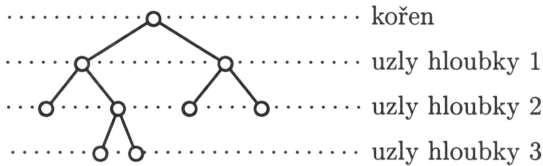
jsou některé dvě A -tice stejné. Tím je důkaz tvrzení úlohy hotov.

Kategorie P

Texty úloh

P – I – 1

Binární strom je struktura tvořená jednotlivými uzly. Jeden z uzlů je význačný — říkáme mu kořen. Každý z uzlů buď nemá žádného následníka (pak se nazývá list), nebo má právě dva následníky (další uzly stromu). Hloubkou uzlu rozumíme jeho vzdálenost od kořene stromu. Uvědomte si, že kořen může být i listem — pak je binární strom tvořen jediným vrcholem hloubky 0. Příklad binárního stromu si můžete prohlédnout na následujícím obrázku:



Abychom mohli binární stromy jednoduše popisovat, zavedeme si následující kódování: k -tý řádek kódování (pro $k = 0, 1, 2 \dots$) popisuje právě uzly hloubky k v pořadí zleva doprava. Uzel binárního stromu, který není listem, budeme v našem kódování zobrazovat znakem U, listy budeme označovat znakem L. Binární strom z předchozího obrázku tedy bude zakódován jako:

U
UU
LULL
LL

Soutěžní úloha. Je dán počet listů N ($N \leq 10\,000$) a jejich hloubky v binárním stromě (nějakých N přirozených čísel). Napište program, který sestaví strom se zadanými hloubkami listů a ten vypíše v našem kódování. Jestliže vstupním datům vyhovuje více různých binárních stromů,

program vypíše libovolný jeden z nich. Pokud vyhovující strom neexistuje, program o tom vypíše zprávu.

Formát vstupu: První řádek vstupního souboru `stromy.in` obsahuje jediné číslo N (počet listů). Druhý řádek obsahuje N čísel — hloubky listů hledaného stromu.

Formát výstupu: Výstupní soubor `stromy.out` bude obsahovat kódování nalezeného stromu ve výše uvedeném formátu, případně zprávu „Odpovídající strom neexistuje.“.

<i>Příklad 1:</i>	<code>stromy.in</code>	<code>stromy.out</code>
	4	U
	2 3 1 3	UL
		LU
		LL

<i>Příklad 2:</i>	<code>stromy.in</code>	<code>stromy.out</code>
	3	Odpovídající strom neexistuje.
	1 1 2	

P – I – 2

Na království krále Mírumila III. zaútočila nepřátelská vojska a podařilo se jim obsadit několik měst. Král nyní potřebuje dát svému generálovi příkaz k protiútoku (bez příkazu přeci generál nemůže bojovat). Generál však momentálně provádí inspekci vojsk v jiném městě. Je proto třeba vyslat posla, který příkaz co nejrychleji doručí. Příkaz ovšem v žádném případě nesmí padnout do rukou nepřítele! Proto se posel musí neustále držet co nejdále od nepřítelem obsazených měst. Vaším úkolem je navrhnout pro posla co nejlepší trasu.

Soutěžní úloha. Program dostane na vstupu zadaný počet měst N ($1 \leq N \leq 100$). Jednotlivá města budeme označovat čísly $1 \dots N$. Dále je na vstupu uveden počet cest M ($1 \leq M \leq 10\,000$) a seznam těchto cest vedoucích mezi městy. Každá cesta je určena dvojicí čísel měst, která spojuje. Cesty se kříží pouze ve městech a je možno se po nich dostat z libovolného města do libovolného (případně přes města jiná). Další údaj K zadaný na vstupu určuje počet měst obsazených nepřítelem, následuje seznam obsazených měst. Nakonec program dostane číslo města, odkud vyráží posel, a číslo města, kde se zdržuje generál. Váš program má nalézt trasu, jejíž vzdálenost od měst obsazených nepřítelem je maximální. Pokud existuje takových tras více, program určí libovolnou nejkratší z nich. Vzdálenost měst A a B počítáme jako minimální

počet cest, po kterých musíme projít, abychom se dostali z města A do města B . Vzdálenost trasy od města A je pak nejmenší ze vzdáleností města A od jednotlivých měst ležících na uvažované trase. Vzdáleností trasy od obsazených měst rozumíme nejmenší ze vzdáleností mezi trasou a některým z obsazených měst nebo nulu, pokud některé město na trase samé je obsazeno.

Formát vstupu: První řádek vstupního souboru `posel.in` obsahuje čísla N (počet měst) a M (počet cest). Po něm následuje M řádků, z nichž každý obsahuje popis jedné cesty. Cesta je popsána dvojicí čísel koncových měst. Následuje řádek s číslem K (počet obsazených měst) a za ním K řádků s čísly obsazených měst. Poslední řádek vstupního souboru obsahuje číslo města, odkud vyjíždí posel, a číslo města, kde dlí generál.

Formát výstupu: Výstupem programu v souboru `posel.out` jsou čísla měst na nejlepší nalezené trase uvedené v pořadí, v jakém jimi má posel projíždět. Všechna čísla měst jsou zapsána na jediném řádku výstupního souboru a jsou oddělena mezerami.

Příklad: `posel.in`

10 12

1 2

2 3

3 4

4 5

2 5

1 6

6 7

7 8

8 5

1 9

9 10

10 5

1

3

1 5

`posel.out`

1 9 10 5

P – I – 3

Skupina přátel se rozhodla, že v létě podniknou společný výlet na kolech. Většina zvolené trasy však vede přírodní rezervací, a proto mohou

nocovat pouze v kempech. Kempy, ve kterých na svém výletě přespí, ještě nevybrali.

Celková délka naplánované trasy je L ($1 \leq L \leq 1\,000\,000\,000$). Maximální vzdálenost, kterou naši přátelé mohou urazit za jeden den, je K , tj. ve dvou po sobě následujících dnech musí přespát v kempech vzdálených nejvýše o K . Na naplánované trase se nachází celkem N kempů ($0 \leq N \leq 10\,000$); i -tý kemp je ve vzdálenosti l_i od začátku jejich výletu a cena za přespání v něm je c_i ($1 \leq c_i \leq 20\,000$). Čísla L , K , l_i a c_i jsou celá kladná; všechna l_i jsou navzájem různá a platí $0 < l_1 < l_2 < \dots < l_N < L$.

Vášim úkolem je rozhodnout, zda skupina může naplánovanou trasu projet. Pokud lze trasu takto projet, pak určete, ve kterých kempech mají přespát tak, aby:

- a) jejich výlet trval co nejmenší počet dní.
- b) celková cena za přespání v kempech byla co nejmenší.

Úlohy a) a b) řešte zvlášť; v případě, že jednu z těchto úloh neumíte vyřešit, řešte pouze druhou z nich.

Formát vstupu: Vstupní soubor se jmenuje `vylet.in`. Na prvním řádku jsou čísla L , K a N oddělená mezerou. Na dalších N řádcích následují dvojice čísel l_i a c_i oddělených mezerou, postupně pro $i = 1$ až $i = N$.

Formát výstupu: Výstupní soubor se jmenuje `vylet-a.out` pro úlohu a) a `vylet-b.out` pro úlohu b). Na prvním řádku je věta `Trasu nelze projet.`, pokud výlet nelze uskutečnit tak, aby naši přátelé nikdy neujeli za den vzdálenost větší než K . V opačném případě první řádek obsahuje dvě čísla — M a C . První z nich, M ($0 \leq M$), je počet kempů, ve kterých skupina přespí, druhé z nich, C , je cena, kterou za přespání v těchto kempech zaplatí. Druhý řádek souboru obsahuje M mezerou oddělených čísel kempů, v nichž naši přátelé budou nocovat. Kempy jsou číslovány od jedné. Pokud je $M = 0$, nemusí být druhý řádek vůbec uveden. V případě, že existuje více řešení splňujících podmínku a) nebo b), program může vypsát libovolné jedno z nich.

Příklad 1: `vylet.in`

15 10 2

2 11

4 12

`vylet-a.out, vylet-b.out`

`Trasu nelze projet.`

Příklad 2: vylet.in

8 10 1

7 11

vylet-a.out, vylet-b.out

0 0

Příklad 3: vylet.in

25 5 9

4 2

5 8

8 2

10 8

12 2

15 8

16 2

20 8

24 2

vylet-a.out

4 32

2 4 6 8

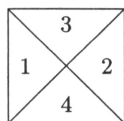
vylet-b.out

5 16

1 3 5 7 8

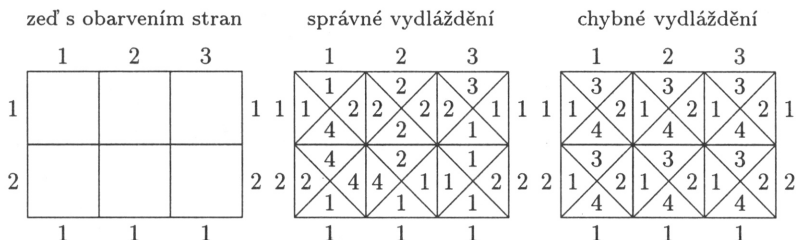
P – I – 4

Nejprve několik definic: *Dlaždice* jsou stejně velké čtverce s obarvenými hranami. Konkrétnímu přiřazení barev hranám dlaždice budeme říkat *typ* dlaždice a budeme jej zapisovat jako uspořádanou čtveřici (l, p, h, d) udávající barvu v pořadí levé, pravé, horní a dolní hrany. Abychom si usnadnili práci, budeme barvy označovat různými symboly — písmeny, čísla apod. Dlaždice typu $(1, 2, 3, 4)$ bude tedy vypadat následovně:



Prostor, který budeme dláždit (budeme mu říkat *zed'*), má tvar obdélníku o velikosti $m \times n$ (m i n jsou přirozená čísla; jednotkou délky budiž délka hrany dlaždice). Strany obdélníku jsou rozděleny na úseky jednotkové délky a každému úseku je opět přiřazena barva. Naším cílem je pokrýt *zed'* dlaždicemi tak, aby v každém z $m \cdot n$ jednotkových čtverců zdi byla umístěna právě jedna dlaždice, sousední dlaždice se dotýkaly vždy hranami téže barvy a rovněž krajní dlaždice přiléhaly k okraji zdi vždy hranou takové barvy, jakou má i příslušný úsek okraje zdi. Dlaždice není povoleno otáčet.

Příklad:



Pomocí dláždění můžeme snadno řešit úlohy, jejichž výsledkem je buďto odpověď ANO, nebo NE: sestavíme vhodnou množinu typů dlaždic (ta je pro daný problém pevná — nezávisí na vstupu), vezmeme vhodně velkou zeď, její horní okraj obarvíme podle vstupu našeho problému, ostatní okraje ponecháme jednobarevné a budeme se ptát, zda je tuto zeď možno vydláždit či nikoliv. Přitom chceme, aby tento výsledek byl shodný s řešením naší úlohy.

Abychom se nemuseli zabývat tím, jak přesně velkou zeď máme zvolit pro ten či onen vstup úlohy, budeme šířku zdi volit vždy stejnou, jako je délka vstupu (horní okraj tedy bude celý zaplněn vstupem), zatímco výšku zdi použijeme nejmenší, pro níž existuje vydláždění s použitím naší sady dlaždic.

Když tento způsob počítání srovnáme s klasickým programováním, zjistíme, že zvolená sada dlaždic tvoří v našem modelu něco podobného programu a potřebná výška zdi vzdáleně odpovídá době běhu výpočtu — budeme se proto snažit, aby u našich řešení byla co nejmenší.

Formálně řečeno, *dlaždicový program* je uspořádaná čtveřice

$$D = (T, l_0, p_0, d_0),$$

kde T je konečná množina typů dlaždic $\{(l_1, p_1, h_1, d_1), \dots, (l_k, p_k, h_k, d_k)\}$ a l_0, p_0 a d_0 jsou okrajové barvy. *Rozhodovací úlohou* $P(x)$ rozumíme úlohu zjistit, zda vstup x (konečná posloupnost symbolů, resp. barev z předem určené konečné množiny) má požadovanou vlastnost P . Říkáme, že dlaždicový program řeší rozhodovací úlohu $P(x)$, jestliže platí, že $P(x) = \text{ANO}$ právě tehdy, když existuje $v > 0$ takové, že je možno vydláždit dlaždicemi typů obsažených v množině T zeď velikosti $|x| \times v$ s horní hranou obarvenou vstupem x a levou, pravou a dolní hranou obarvenou po řadě barvami l_0, p_0 a d_0 . Od každého typu je možno použít libovolně mnoho dlaždic. *Složitostí* programu D pro daný vstup

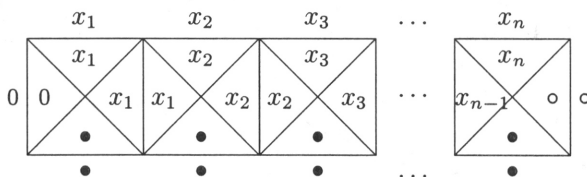
x nazveme nejmenší v , pro něž to je možné; pokud takové neexistuje, a tedy $P(x) = \text{NE}$, definujeme složitost jako nulovou. Složitost programu je funkce délky vstupu n , jejíž hodnota udává maximum ze složitostí programu pro jednotlivé vstupy této délky.

Příklad: Zkusme nyní zkonstruovat dlaždicový program, který bude ověřovat, zda je daná posloupnost tvořená přirozenými čísly $x_1, \dots, x_n$ ($0 \leq x_i \leq 9$) neklesající. Použijeme dlaždice následujících typů:

$$T = \left\{ \begin{array}{|c|c|} \hline x & x \\ \hline i & \bullet \\ \hline \end{array}, \begin{array}{|c|c|} \hline x & \circ \\ \hline i & \bullet \\ \hline \end{array}; 0 \leq i \leq x \leq 9 \right\},$$

levý okraj obarvíme barvou 0, pravý $\circ$, dolní barvou $\bullet$ a tvrdíme, že tento program řeší danou úlohu se složitostí $O(1)$. To je ovšem třeba dokázat.

Především si ověříme, že každá zeď, kterou je možno vydláždit dlaždicemi typů z množiny T , má jednotkovou výšku. To jasně plyne z toho, že spodní hrana každé dlaždice má barvu $\bullet$, která se nevyskytuje na žádné horní hraně. Z téhož důvodu se dlaždice mající na své pravé hraně barvu $\circ$ musí vyskytovat těsně u pravého okraje zdi a nikde jinde. Každé korektní dláždění proto musí vypadat takto:



což je ovšem možné právě tehdy, když $0 \leq x_1 \leq x_2 \leq \dots \leq x_{n-1} \leq x_n$, tedy když posloupnost na vstupu je neklesající.

Soutěžní úlohy.

a) Sestrojte dlaždicový program, který o dané posloupnosti nul a jedniček zjistí, zda je dvojkovým zápisem nějakého přirozeného čísla dělitelného pěti.

b) Sestrojte dlaždicový program, který o dané posloupnosti přirozených čísel $x_1, \dots, x_n$ ($0 \leq x_i \leq 9$) rozhodne, je-li nekonstantní (to jest vydláždění existuje právě tehdy, když existují indexy i, j takové, že $x_i \neq x_j$).

P – II – 1

Pepík našel na půdě u babičky krabici s dřevěnými tyčkami. Začal si s nimi hrát a sestavovat z nich trojúhelníky různých tvarů. Uviděl ho jeho tatínek a začalo ho zajímat, kolik různých trojúhelníků lze z těchto tyček sestavit. Vaším úkolem je pomoci mu s nalezením odpovědi na tuto otázku.

Váš program na vstupu obdrží celé číslo N (počet tyček) a dále N navzájem různých kladných čísel d_1 až d_N (délky tyček). Úkolem vašeho programu je určit počet trojic $i < j < k$ takových, že čísla d_i , d_j a d_k splňují trojúhelníkovou nerovnost, tj. $d_i < d_j + d_k$, $d_j < d_i + d_k$ a $d_k < d_i + d_j$.

Příklad. Pro $N = 5$ a $d_1 = 5.5$, $d_2 = 1.5$, $d_3 = 2.0$, $d_4 = 2.5$, $d_5 = 7.5$ váš program odpoví číslem 2, neboť trojúhelník lze sestavit pouze z trojice tyček o délkách d_1 , d_4 a d_5 a dále z trojice o délkách d_2 , d_3 a d_4 . Všimněte si, že trojice tyček o délkách d_1 , d_3 a d_5 netvoří trojúhelník.

P – II – 2

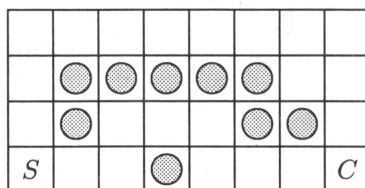
Společnost pro Rovnoprávnost robotů a lidí se snaží vyvinout robota, který by se mohl sám pohybovat v místnosti s překážkami. Bohužel této společnosti chybí softwarový expert a proto se rozhodla, že vás požádá o pomoc.

Robot se má pohybovat v obdélníkové místnosti, na jejíž podlaze je nakreslena čtvercová síť. Na některých políčkách v místnosti jsou postaveny překážky — na tato políčka nesmí robot během svého pohybu vstoupit. Robot se bude po místnosti pohybovat pouze rovnoběžně s některou ze stěn. Společnost však trpí i nedostatkem schopných techniků, a tak jsou možnosti pohybu robota po místnosti značně limitovány. Robot rozpoznává celkem tři příkazy: Krok, Doleva a Doprava. Při obdržení příkazu Krok se robot přesune na sousední políčko v tom směru, ve kterém je právě natočen. Při obdržení příkazu Doleva se otočí o 90 stupňů doleva a při obdržení Doprava se otočí o 90 stupňů doprava.

Vaším úkolem je vytvořit program, který jako vstup dostane rozměry čtvercové sítě na podlaze místnosti (M a N), souřadnice políčka, na kterém se robot právě nachází, a souřadnice políčka, na které se má robot přesunout. Souřadnice políček číslujeme od 1, první souřadnice udává řádek, druhá sloupec. Levý horní roh místnosti má souřadnice [1, 1] (viz příklad níže). Každý z následujících M řádků obsahuje N čísel 0 nebo

1. Je-li j -té číslo na i -tém řádku 1, pak na políčku se souřadnicemi $[i, j]$ je překážka, pokud je toto číslo 0, pak se na políčku se souřadnicemi $[i, j]$ překážka nenachází. Úkolem je nalézt a vypsát posloupnost příkazů, podle nichž robot dojde z počátečního políčka na cílové. Věc má však ještě jeden háček: Provedení příkazů Doleva a Doprava je časově velmi náročné a vámi vytvořená posloupnost instrukcí pro pohyb robota v místnosti by měla obsahovat co nejmenší počet těchto dvou příkazů. Počet příkazů Krok může být libovolný. Počáteční natočení robota si můžete zvolit. V případě, že robot nemůže přejít z počátečního na cílové políčko, vypište vhodnou zprávu.

Příklad. Představme si místnost se čtvercovou sítí 4×8 z následujícího obrázku:



Úkolem je přesunout robota z políčka označeného S (o souřadnicích $[4, 1]$) na políčko označené C (o souřadnicích $[4, 8]$). Vstup vašeho programu by tedy vypadal následovně:

```

4 8
4 1
4 8
0 0 0 0 0 0 0 0
0 1 1 1 1 1 0 0
0 1 0 0 0 1 1 0
0 0 0 1 0 0 0 0

```

Optimální program pro přesun robota je následující (počáteční natočení robota je nahoru):

```

Krok Krok Krok Doprava Krok Krok Krok Krok Krok
Krok Krok Doprava Krok Krok Krok

```

Při této cestě robot udělá 13 kroků a dvakrát se otočí; všimněte si též existence cesty s 9 kroky a 4 otočeními — tato cesta je sice kratší, ale

podle zadání úlohy není optimální, neboť se během ní robot musí otočit vícekrát.

P – II – 3

Malému Pepíčkovi se jednoho dne dostaly do ruky nůžky. A protože byl Pepíček tvořivý po tatínkovi, jenž byl moderním malířem, rozhodl se vylepšit jeden jeho obraz ve tvaru konvexního n -úhelníku. Vybral si dva vrcholy tohoto n -úhelníku a obraz přestříhl po spojnici těchto dvou vrcholů. Pak si na jedné ze vzniklých částí opět vybral dva vrcholy a část opět přestříhl. Když si takto Pepíček chvilku hrál, přistihl ho tatínek, nůžek ho nekompromisně zbavil a začal zachraňovat, co se dá. Po chvilce zjistil, že obraz dohromady už nesloží. Rozhodl se tedy, že alespoň nalezne zbylou část s největším počtem vrcholů a tu vystaví na své nadcházející výstavě jako miniaturu. A právě s hledáním mu máte pomoci vy.

Navrhněte co nejefektivnější algoritmus, který dostane na vstupu počet vrcholů původního obrazu n , počet Pepíčkových stříhů k a popis jednotlivých stříhů a na základě těchto údajů určí počet vrcholů té zbylé části, která jich má nejvíce. Každý stříh je popsán dvojicí čísel (a_i, b_i) , což jsou čísla vrcholů v původním n -úhelníku, mezi kterými Pepíček stříhl vedl. Vrcholy n -úhelníku jsou číslovány po obvodu po řadě čísly od 1 do n . Snažte se, aby časová ani paměťová složitost vašeho řešení nezávisela na počtu vrcholů obrazu.

Příklad. Pro $n = 10$, $k = 3$ a stříhy $(1, 8)$, $(7, 5)$ a $(4, 2)$ má největší část 6 vrcholů.

P – II – 4

(Definice dlaždicových programů je stejná jako v úloze P–I–4, pouze příklad jejich použití je složitější a ukazuje, jak je možno využívat více řádků dlaždic.)

Příklad. Zkonstruuje dlaždicový program, který bude ověřovat, zda je daná posloupnost tvořená přirozenými čísly $x_1, \dots, x_n$ ($0 \leq x_i \leq 9$) *vyvážená*, tzn. zda obsahuje stejný počet sudých a lichých čísel.

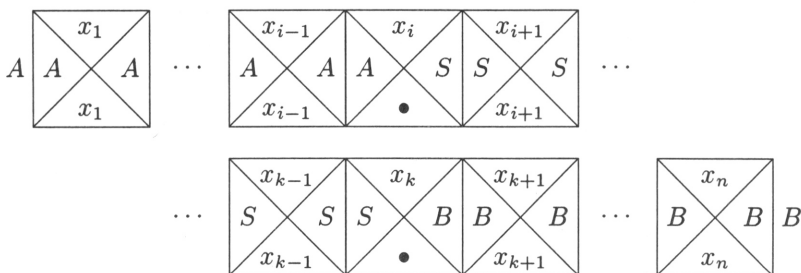
Myšlenka našeho řešení je velice jednoduchá: sestrojíme sadu dlaždic, která bude umožňovat právě taková vydláždění, v nichž v každém řádku přepíšeme právě jedno sudé a jedno liché číslo na $\bullet$. Dolní okraj zdi obarvíme též barvou $\bullet$. Jelikož obarvení spodního okraje je vyvážené a vydláždění každého řádku vyváženost zachovává, pak jakákoliv

vstupní posloupnost, pro kterou vydláždění existuje, je opravdu vyvážená. A naopak: pokud máme vyváženou posloupnost, pak snadno ověříme, že vydláždění existuje: vybereme si libovolné sudé a libovolné liché číslo (z vyváženosti víme, že v posloupnosti taková dvojice je), ta jedním řádkem dlaždic přepíšeme na $\bullet$ a toto opakujeme tak dlouho ($n/2$ -krát), dokud nebudou všechna čísla přepsána. Pokud se nám tedy podaří takový dlaždicový program sestavit, bude zadanou úlohu řešit se složitostí $O(n)$.

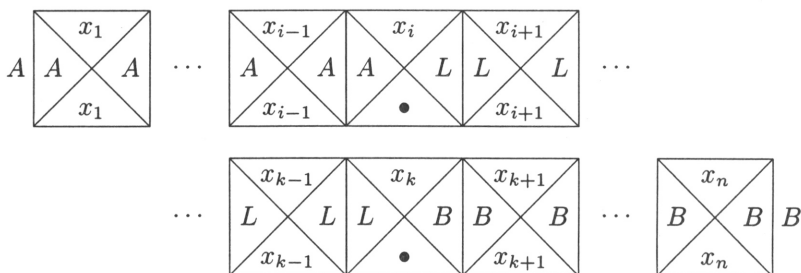
Hledaný program může vypadat například následovně:

$$T = \left\{ \begin{array}{|c|c|} \hline x & A \\ \hline A & x \\ \hline \end{array}, \begin{array}{|c|c|} \hline S & S \\ \hline S & x \\ \hline \end{array}, \begin{array}{|c|c|} \hline L & L \\ \hline L & x \\ \hline \end{array}, \begin{array}{|c|c|} \hline B & B \\ \hline B & x \\ \hline \end{array}, \begin{array}{|c|c|} \hline s & S \\ \hline A & \bullet \\ \hline \end{array}, \begin{array}{|c|c|} \hline l & L \\ \hline A & \bullet \\ \hline \end{array}, \begin{array}{|c|c|} \hline s & B \\ \hline L & \bullet \\ \hline \end{array}, \begin{array}{|c|c|} \hline l & B \\ \hline S & \bullet \\ \hline \end{array}; \right. \\ \left. x \in \{0, \dots, 9, \bullet\}, s \in \{0, 2, 4, 6, 8\}, l \in \{1, 3, 5, 7, 9\} \right\},$$

levý okraj obarvíme barvou A , pravý barvou B a dolní barvou $\bullet$. Z těchto dlaždic je možno konstruovat výhradně řádky typu



kde x_i je sudé a x_k liché, případně



pro x_i liché a x_k sudé. To jsou přesně řádky, jaké jsme potřebovali.

Soutěžní úloha. Sestrojte dlaždicový program, který o dané posloupnosti přirozených čísel $x_1, x_2, \dots, x_n$ ($0 \leq x_i \leq 9$) rozhodne, zda je symetrická, tj. zda $x_1 = x_n, x_2 = x_{n-1}, \dots, x_i = x_{n-i+1}, \dots, x_n = x_1$.

P – III – 1

Pan Kašparov byl náruživý hráč šachů. Protože ale často postrádal vhodného protihráče a hrát sám proti sobě ho už nebavilo (vždy si odhalil všechny léčky), vymyslel si následující hru: Na šachovnici o rozměrech $N \times N$ je třeba rozmístit N věží tak, aby se vzájemně neohrožovaly. Aby hra nebyla příliš jednoduchá, pro každou věž je určen obdélník, do kterého se věž musí umístit.

Vášim úkolem je navrhnout algoritmus, který bude tuto hru hrát. Na vstupu dostane rozměr šachovnice a počet věží N a dále popis N obdélníků. Jeden obdélník je popsán čtveřicí čísel A_x, A_y, B_x, B_y , $1 \leq A_x \leq B_x \leq N$, $1 \leq A_y \leq B_y \leq N$, kde A_x, A_y jsou souřadnice levého horního rohu obdélníku a B_x, B_y jsou souřadnice pravého dolního rohu obdélníku. Řádky číslujeme od 1 do N shora dolů a sloupce od 1 do N zleva doprava. Na výstup pak algoritmus vypíše souřadnice jednotlivých rozmístěných věží, nebo zprávu, že rozmístění dle pravidel hry neexistuje. Pokud existuje více různých řešení, stačí najít jedno libovolné z nich.

Příklad 1: $N = 4$

```
1 1 1 1
4 4 4 4
1 1 3 3
3 2 4 4
```

Rozmístění věží může být:

$(1, 1), (4, 4), (2, 2), (3, 3)$.

Příklad 2: $N = 3$

```
1 1 3 1
2 1 3 1
2 2 3 3
```

Rozmístění věží neexistuje.

P – III – 2

Napište program, který na vstupu obdrží přirozené číslo N a nalezne nejmenší takové přirozené číslo x , že x je dělitelné číslem N a zároveň dekadický zápis čísla x je tvořen pouze ciframi nula a jedna. V případě, že takové číslo x neexistuje, vypíše váš program vhodnou zprávu. Například pro $N = 6$ je hledaným číslem x číslo 1110.

(Definice dlaždicových programů je stejná jako v úloze P–I–4, pouze navíc vysvětluje, jak lze dlaždicové programy používat k výpočtu funkcí a uvádí příklad tohoto použití.)

Dlaždicové programy je možno používat nejen k řešení rozhodovacích problémů, ale také k výpočtu hodnot funkcí. Výpočet funkce $f(x)$ totiž můžeme snadno převést na rozhodovací problém $P(x, y) =$ „je $y = f(x)$?“, o kterém budeme vědět, že pro každé x bude $P(x, y)$ splněno pro právě jednu hodnotu y . Navíc můžeme dlaždicovému programu x i y zadat jako jeden vstup tak, že barvy dlaždic nebudou odpovídat hodnotám, nýbrž jejich uspořádaným dvojicím.

Příklad. Zkonstruujeme dlaždicový program, který pro každé číslo zapsané ve dvojkové soustavě spočte dvojkový zápis tohoto čísla vyděleného třemi (předpokládejme, že je dělitelné beze zbytku). Jinými slovy máme o dané posloupnosti dvojic $(x_1, y_1), \dots, (x_n, y_n)$ zjistit, zda $\langle y_1, \dots, y_n \rangle = \frac{1}{3} \langle x_1, \dots, x_n \rangle$.

Řešení založíme na tradičním algoritmu na písemné dělení (ten je na použité číselné soustavě nezávislý): zvolíme $z_0 = 0$ a spočteme postupně pro všechna k hodnoty $z_k = (2z_{k-1} + x_k) \bmod 3$ a $y_k = \lfloor \frac{1}{3}(2z_{k-1} + x_k) \rfloor$. Nyní dokážeme indukci, že pro každé k je

$$\langle x_1, \dots, x_k \rangle = 3 \cdot \langle y_1, \dots, y_k \rangle + z_k.$$

Pro $k = 0$ rovnost platí. Platí-li pro $k - 1$, pak pro k dostaneme:

$$\begin{aligned} \langle x_1, \dots, x_k \rangle &= 2 \cdot \langle x_1, \dots, x_{k-1} \rangle + x_k = \\ &= 2 \cdot (3 \cdot \langle y_1, \dots, y_{k-1} \rangle + z_{k-1}) + x_k = \\ &= 3 \cdot 2 \cdot \langle y_1, \dots, y_{k-1} \rangle + 2z_{k-1} + x_k = \\ &= 3 \cdot 2 \cdot \langle y_1, \dots, y_{k-1} \rangle + 3y_k + z_k = \\ &= 3 \cdot \langle y_1, \dots, y_k \rangle + z_k. \end{aligned}$$

Nyní si stačí zvolit následující sadu dlaždic:

$$T = \left\{ \begin{array}{|c|} \hline \begin{array}{c} xy \\ a \quad b \\ \bullet \end{array} \\ \hline \end{array} ; x, y \in \{0, 1\}, a \in \{0, 1, 2\}, b = (2a + x) \bmod 3, \right. \\ \left. y = \left\lfloor \frac{1}{3}(2a + x) \right\rfloor \right\},$$

levý a pravý okraj budou mít barvu 0, spodní barvu $\bullet$.

Z těchto dlaždic je možno konstruovat výhradně jednořádková vydláždění (• není barvou horního okraje žádné dlaždice), v nichž má k -tá dlaždice na svém okraji z_k a pro (x_k, y_k) na jejím horním okraji platí $y_k = \lfloor \frac{1}{3}(2z_{k-1} + x_k) \rfloor$. Jinými slovy tato vydláždění odpovídají přesně hodnotám spočteným naším algoritmem, tedy i požadovanému výsledku. Tím je problém vyřešen.

Soutěžní úloha. Sestrojte dlaždicový program, který bude uspořádávat posloupnosti nul a jedniček vzestupně, to znamená, že na posloupnost dvojic nul a jedniček $(x_1, y_1), \dots, (x_n, y_n)$ odpoví ANO právě tehdy, pokud $y_1, \dots, y_n$ je posloupnost vzniklá vzestupným uspořádáním posloupnosti $x_1, \dots, x_n$, tzn. $y_1 \leq \dots \leq y_n$ a posloupnosti x a y obsahují tytéž prvky, nanejvýš v jiném pořadí.

Příklad. Na posloupnost $(1, 0), (0, 0), (0, 0), (1, 1), (0, 1), (1, 1)$ program odpoví ANO, na $(1, 1), (0, 0)$ NE, na $(1, 0), (1, 1)$ taktéž NE.

P – III – 4

Program: FORMAT.PAS / FORMAT.C / FORMAT.CPP

Vstup: FORMAT.IN

Výstup: FORMAT.OUT

Pro textový editor potřebujeme napsat program sloužící k formátování textu. Editor pracuje ve znakovém režimu s neproporcionálním písmem. Všechny znaky tedy mají stejnou šířku a také mezera má pevnou šířku stejnou jako každý jiný znak. Rovněž pomocné symboly obsažené v textu (jako jsou interpunkční znaménka, závorky či uvozovky) se zpracovávají stejně jako jakékoliv jiné znaky. Editor je poměrně jednoduchý, takže dělení slov nepřipouští. Program budeme používat vždy k formátování jednoho odstavce textu.

Pro potřeby formátování rozumíme slovem každou souvislou posloupnost nemezerových znaků, která je na obou koncích ukončena mezerou nebo začátkem či koncem řádku. Pomocné symboly obsažené v textu jsou tedy součástí těch slov, od nichž nejsou odděleny mezerou.

Cílem formátování textu je vhodné rozložení slov na jednotlivé řádky tak, aby byl celý text zarovnán „do bloku“ (tzn. k levému i pravému okraji) při zadané šířce řádku. Přitom mezery mezi slovy musí být co možná nejmenší a v textu co nejrovnoměrněji rozloženy. Tyto obecné požadavky si nyní upřesníme: Velikosti mezer mezi slovy na témže řádku (kromě posledního řádku odstavce) se mohou lišit maximálně o 1, před

prvním a za posledním slovem na řádce nesmí být mezera. Pokud je na řádce pouze jedno slovo, je rozložení mezer libovolné. Na posledním řádce odstavce musí být slova oddělena právě jednou mezerou a před prvním slovem nesmí být mezera.

Splňuje-li text tyto závazné požadavky, potom kvalitu zformátování odstavce hodnotíme trestnými body. Ohodnocení odstavce je součtem ohodnocení jednotlivých řádků. Ohodnocení jednoho řádku je dáno výsledkem funkce $F(\langle Width \rangle, \langle Chars \rangle, \langle Words \rangle, \langle Last \rangle)$, kde $\langle Width \rangle$ je šířka stránky (tj. počet znaků na řádce zformátovaného textu včetně všech mezer), $\langle Chars \rangle$ je počet nemezerových znaků na řádce, $\langle Words \rangle$ je počet slov na řádce a $\langle Last \rangle$ značí, zda se ohodnocuje poslední řádek odstavce či nikoliv.

Ohodnocovací funkce F zapsaná v programovací jazyce C vypadá následovně:

```
int F(int Width, int Chars, int Words, int Last)
{
    int Spaces = Width - Chars - Words + 1; /* Počet zbytečných mezer */
    int BasePen = LINEPENALTY;           /* Základní trestné body za řádek */

    if (Spaces < 0)                       /* Nevejde se text na řádek? */
        return INFTYPEN;
    if (Last)                             /* Poslední řádek? */
    {
        if (4*(Chars + Words - 1) <= Width) /* Je poslední řádek moc krátký? */
            BasePen += SMALLLINEPEN;
        return BasePen;
    }
    if (Words == 1)                       /* Pouze jedno slovo na řádce? */
        BasePen += SINGLEWORDPEN;
    return Spaces * Spaces + BasePen;     /* Ohodnocení celého řádku */
}
```

V Pascalu je zápis funkce F obdobný:

```
function F(Width, Chars, Words: Integer; Last: Boolean): Integer;
var
    Spaces : Integer;           { Počet zbytečných mezer na řádce }
    BasePen : Integer;          { Základní trestné body za řádek }
begin
    BasePen := LINEPENALTY;
    Spaces := Width - Chars - Words + 1;
    if Spaces < 0 then           { Nevejdou se slova na řádek? }
        F := INFTYPEN
    else if Last then begin      { Ohodnocujeme poslední řádek? }
        if 4*(Chars + Words - 1) <= Width then { Je řádek moc krátký? }
            Inc(BasePen, SMALLLINEPEN);
        F := BasePen;
    end;
```

```

end
else begin
  if Words = 1 then          { Je jen jedno slovo na řádku? }
    Inc(BasePen, SINGLEWORDPEN);
    F := Spaces * Spaces + BasePen; { Spočteme výsledné ohodnocení }
  end;
end;
end;

```

Hodnoty konstant jsou:

LINEPENALTY = 10

SMALLLINEPEN = 5

SINGLEWORDPEN = 20

INFTEYPEN = 30 000

Vaším úkolem je zformátovat odstavec textu při zadané šířce řádku co nejkvalitněji, tzn. splnit všechny závazné požadavky kladené na formátování a přitom dosáhnout co nejnižšího ohodnocení odstavce trestnými body podle funkce F .

Vstup: Ve vstupním souboru `FORMAT.IN` je na prvním řádku zadána požadovaná šířka stránky po zformátování. Na dalších řádcích se nachází text odstavce určený ke zformátování. Můžete předpokládat, že žádný z těchto řádků není delší než 100 znaků, na začátku ani na konci žádného řádku není mezera a mezi jednotlivými slovy na řádku je vždy právě jedna mezera. Vstupní soubor nebude delší než 10 000 znaků (počítáno včetně mezer mezi slovy).

Výstup: Do výstupního souboru `FORMAT.OUT` zapište zadaný text odstavce zformátovaný co nejkvalitněji podle výše uvedených zásad (tj. s nejnižší možnou hodnotou ohodnocovací funkce).

Příklad.

`FORMAT.IN`

40

Each section in this document will have the string "<section>" at the right-hand side of the section title. Each subsection will have "<subsection>" at the right-hand side. These strings are meant to make it easier to search through the document.

`FORMAT.OUT` (jedno z možných řešení; symbol '␣' označuje mezera)

Each␣section␣in␣this␣document␣will
have␣the␣string␣"<section>"␣at␣the
right-hand␣side␣of␣the␣section␣title.
Each␣subsection␣will␣have␣"<subsection>"
at␣the␣right-hand␣side␣.␣These␣strings
are␣meant␣to␣make␣it␣easier␣to␣search
through␣the␣document.

Program: OKRUZNI.PAS / OKRUZNI.C / OKRUZNI.CPP
Vstup: OKRUZNI.IN
Výstup: OKRUZNI.OUT

Ve městě Turiststadt začal vzkvétat turistický ruch. Aby podpořili jeho další rozkvět, rozhodli se moudří radní založit společnost City-tour. Posláním této společnosti je provozovat ve městě několik vyhlídkových okružních autobusových linek. Vaším úkolem je vytvořit program, který navrhne trasy vyhlídkových autobusových linek městem podle požadavků radních, popřípadě zjistí, že autobusové linky nelze dle jejich požadavků vytvořit.

Město je tvořeno křižovatkami, které jsou navzájem spojeny ulicemi. Každá ulice spojuje právě dvě křižovatky. Dvě stejné křižovatky mohou být spojeny více různými ulicemi. Křižovatkou rozumíme i místo, do kterého vede jen jedna nebo dvě ulice. Radní kladou na plánované trasy autobusových linek následující požadavky: Aby si turisté mohli pohodlně prohlédnout každou ulici ve městě a přitom se zbytečně neplýtvalo náklady na provoz autobusových linek, musí každou ulicí projíždět právě jedna autobusová linka. Žádná z linek nesmí projet některou z křižovatek více než jednou. Trasy linek musí být navrženy tak, aby první a poslední křižovatka na trase byla stejná — jinak by se linky provozované touto společností daly jen stěží nazývat okružní.

Vstup: První řádek vstupního souboru OKRUZNI.IN obsahuje dvě čísla oddělená mezerou — počet křižovatek (N , $1 \leq N \leq 120$) a počet ulic (M). Křižovatky jsou očíslovány čísly od 1 do N . Následujících M řádků vstupního souboru obsahuje popis jednotlivých ulic ve městě: Každý z těchto řádků obsahuje dvě čísla představující čísla křižovatek, které příslušná ulice spojuje. První číslo na každém z těchto řádků je menší než druhé z nich. Tyto řádky jsou v souboru seříděny podle prvního čísla; v případě, že se shoduje více ulic v prvním čísle, jsou seříděny podle druhého čísla. Můžete předpokládat, že počet různých ulic spojujících dvě stejné křižovatky je nejvýše 200.

Výstup: Výstupní soubor OKRUZNI.OUT obsahuje tolik řádků, kolik má společnost provozovat autobusových linek. Každý řádek obsahuje popis právě jedné autobusové linky. Trasa autobusové linky je popsána jako posloupnost čísel křižovatek, kterými linka projíždí. První a poslední číslo uvedené na řádku je tedy stejné (linka začíná a končí na stejné křižovatce). Jednotlivá čísla jsou na každém řádku oddělena právě jednou

mezerou. Pokud nelze trasy linek navrhnout tak, aby vyhovovaly podmínkám ze zadání úlohy, potom výstupní soubor obsahuje jediný řádek se slovem „Nelze“.

Příklad 1.

Vstupní soubor OKRUZNI.IN:

5 12

1 2

1 2

1 2

1 2

1 3

1 3

2 3

2 5

3 4

3 5

3 5

4 5

Výstupní soubor OKRUZNI.OUT:

1 2 3 1

2 1 2

3 4 5 3

2 5 3 1 2

Příklad 2.

Vstupní soubor OKRUZNI.IN:

3 4

1 2

1 2

1 3

2 3

Výstupní soubor OKRUZNI.OUT:

Nelze

P – I – 1

Je jasné, že za každé dva uzly hloubky K , $K > 0$ musí existovat jeden uzel hloubky $K - 1$, který není listem — každé dva uzly musíme pod nějaký uzel „zavěsit“. Protože navíc pod každý uzel můžeme zavěsit pouze žádný nebo dva uzly, musí být počet uzlů hloubky K sudý. Výše uvedená pozorování nám již dávají návod na sestrojení algoritmu. Pro každou hloubku si budeme pamatovat počet listů a počet ostatních uzlů. Budeme postupovat od uzlů s největší hloubkou. Pro každou hloubku K , $K > 0$ zkontrolujeme, zda je počet uzlů v ní sudý. Pokud není, strom neexistuje. Pokud je počet uzlů sudý, za každé dva uzly umístěné ve stromu na dané hloubce přidáme do předchozí hloubky jeden uzel. Když se takto dostaneme až k uzlům hloubky 0, stačí ověřit, jestli v této hloubce leží právě jeden uzel, jak je vyžadováno v definici binárního stromu. Pokud není, strom neexistuje. To plyne z toho, že pokud neexistuje uzel hloubky 0, nebyl dán žádný list, a tedy strom nemůže mít žádné uzly. To je ale ve sporu s požadavkem, že každý strom musí mít alespoň kořen. Pokud je uzlů hloubky 0 naopak více, strom neexistuje, protože všechny uzly, které jsme přidávali, byly vynucené a každý strom s danými počty listů tedy musí mít na jednotlivých hloubkách alespoň tolik uzlů jako náš strom. Pokud existuje právě jeden uzel hloubky 0, snadno již ze spočítaných počtů listů a ostatních uzlů v jednotlivých hloubkách vytvoříme požadovaný zápis stromu. Jednoduše vypíšeme tolik L, kolik je počet listů dané hloubky, a tolik U, kolik je počet ostatních uzlů dané hloubky. Algoritmus má časovou i paměťovou složitost $O(N)$. Program je přímou implementací algoritmu.

P – I – 2

Algoritmus řešící tuto úlohu se dá rozdělit do tří fází. V první fázi se pro každé město spočítá, jaká je jeho vzdálenost od nepřítelem obsazených měst (ve smyslu definice uvedené v zadání). Ve druhé fázi se zjistí, jakou maximální vzdálenost od nepřátelských měst dokážeme udržet při cestě z počátečního do cílového města. Ve třetí fázi pak nalezneme nejkratší z tras vedoucích z počátečního do cílového města, které udržují spočtenou vzdálenost.

První fáze: Vzdálenost od obsazených měst budeme hledat pomocí prohledávání do šířky. U každého města si budeme udržovat informaci, zda jsme v něm již byli (na počátku bude nastaveno právě u všech obsazených měst) a jeho vzdálenost od nepřítele. Pro města obsazená nepřítelem bude tato vzdálenost rovna 0. Dále si budeme udržovat frontu měst ke zpracování, do které na začátku uložíme všechna nepřátelská města. V každém kroku výpočtu vždy vezmeme jedno město z fronty a u všech jeho sousedů, ve kterých jsme dosud nebyli, nastavíme vzdálenost o jedna větší, než je vzdálenost vybraného města. U všech těchto sousedů také označíme, že jsme v nich už byli, a přidáme je na konec fronty. První fáze výpočtu končí, když se vyprázdní fronta. Tehdy jsme prošli všechna města a určili jsme vzdálenost každého z nich od nepřítele.

Druhá fáze: V této fázi si budeme udržovat front hned několik, pro každou vzdálenost od nepřátelských měst jednu. Dále si pro každé město budeme zaznamenávat, zda jsme v něm už byli. Také si budeme pamatovat dosud největší nalezenou vzdálenost, kterou dokážeme udržet od nepřítele. Na začátku nastavíme udržitelnou vzdálenost od nepřítele na hodnotu vzdálenosti královského města od nepřítele a toto město vložíme do fronty pro příslušnou vzdálenost. U tohoto města také nastavíme, že jsme v něm už byli. Výpočet probíhá tak, že postupně vyzvedáváme města z fronty pro aktuální udržitelnou vzdálenost, dokud se tato fronta nevyprázdní. Když se fronta vyprázdní, snížíme udržitelnou vzdálenost o jedna a opět začneme vybírat města z příslušné fronty. Vždy, když vezmeme nějaké město z fronty, projdeme všechny jeho sousedy, u dosud nenavštívených z nich nastavíme příznak, že už jsme je nenavštívili, a přidáme je do fronty — jestliže je vzdálenost takového města od nepřátelských měst větší, než je aktuální udržitelná vzdálenost, přidáme vrchol do fronty odpovídající aktuální udržitelné vzdálenosti, jinak město přidáme do fronty odpovídající jeho vzdálenosti od nepřátelských měst. Druhá fáze končí, jakmile vybereme z fronty cílové město. Aktuální udržitelná vzdálenost je pak výslednou udržitelnou vzdáleností.

Třetí fáze: Tato fáze představuje opět prosté prohledávání do šířky. Pro každé město si pamatujeme, zda jsme v něm již byli, a pokud ano, zaznamenáme si také město, ze kterého jsme do něj přišli. Opět používáme frontu na dosud nezpracovaná města. Na začátku vložíme do fronty cílové město. U něj nastavíme, že jsme v něm již byli, a jako jeho předchůdce nastavíme je samé. V každém kroku výpočtu pak vezmeme jedno město z fronty a projdeme všechny jeho sousedy. Každého souseda, kterého jsme dosud nenavštívili a jehož vzdálenost od nepřátelských

měst je větší nebo rovna výsledné udržitelné vzdálenosti, označíme jako navštíveného a přidáme ho na konec fronty. Také u něj jako město, ze kterého jsme přišli, nastavíme právě vybrané město. Prohledávání končí ve chvíli, když je z fronty vyzvednuto počáteční (královské) město. Poté už jenom projdeme cestu z počátečního do cílového města (to je velmi snadné díky odkazům na města, odkud jsme do nich při prohledávání přišli) a cestu vypíšeme.

Algoritmus má časovou složitost $O(M + N)$, kde M je počet cest a N je počet měst.

Správnost algoritmu budeme ukazovat opět po fázích. To, že algoritmus spočte správně vzdálenosti od nepřátelských měst v první fázi, plyne z následujícího: Na počátku mají všechny vrcholy se vzdáleností nula tuto vzdálenost přiřazenu. V okamžiku, kdy jsou zpracovány všechny vrcholy vzdálenosti nula, prošli jsme všechny jejich sousedy, přiřadili jsme jim vzdálenost jedna a zařadili je do fronty. Protože jiné vrcholy vzdálenost jedna mít nemohou, je vzdálenost jedna přiřazena právě všem správným vrcholům. Tuto úvahu lze snadno zobecnit pro libovolnou vzdálenost D . Prohledávání tedy skutečně určí vzdálenosti od nepřátelských měst správně.

Ve druhé fázi se správně spočítá maximální udržitelná vzdálenost od obsazených měst. Sledujeme v ní totiž souběžně všechny možné trasy vedoucí z počátečního města tak dlouho, dokud dokážeme udržet vzdálenost počátečního města (výsledná vzdálenost od nepřítele zřejmě nemůže být větší než vzdálenost počátečního města). Když už neexistuje město s dostatečně velkou vzdáleností, do kterého bychom mohli jít, snížíme udržitelnou vzdálenost o jedna. Všechny vrcholy se vzdáleností o jedna nižší, do kterých se dokážeme dostat přes vrcholy s dosavadní udržitelnou vzdáleností, máme již připraveny v příslušné frontě a začneme tedy prohledávat z nich. Protože udržitelnou vzdálenost snižujeme až když jsme se již dostali všude, kam to bylo možné, její výsledná hodnota bude zřejmě nejvyšší možná.

To, že ve třetí fázi nalezneme nejkratší trasu s danou vzdáleností, je zřejmé. Provádíme totiž jednoduché prohledávání do šířky s tím, že ignorujeme města s příliš malou vzdáleností od nepřítele. Nalezneme tedy určité trasu s dostatečnou vzdáleností od nepřítele. Skutečnost, že to bude trasa nejkratší možná, plyne z vlastností prohledávání do šířky uvedených v první části důkazu. Program je přímou implementací uvedeného algoritmu.

Řešení úlohy rozdělíme na několik částí — nejprve zformulujeme nutné a postačující podmínky pro to, aby skupina mohla projet trasu dle podmínek v zadání úlohy, poté vyřešíme úlohu a) a nakonec nalezneme řešení úlohy b).

Plánovanou trasu lze projet právě tehdy, když tato trasa není delší než vzdálenost, kterou skupina urazí za den, tj. $L \leq K$, anebo když jsou splněny zároveň všechny čtyři následující podmínky:

1. Na trase je alespoň jeden kemp.
2. Vzdálenost prvního kempu od začátku trasy je nejvýše K , tj. $l_1 \leq K$.
3. Vzdálenost libovolných dvou po sobě následujících kempů není větší než K , tj. $l_{i+1} - l_i \leq K$ pro $1 \leq i \leq N - 1$.
4. Vzdálenost posledního kempu od konce trasy je nejvýše K , tj. $L - l_N \leq K$.

Nutnost všech uvedených podmínek je zřejmá; pokud jsou tyto podmínky splněny, pak plán cesty, ve kterém skupina bude cestovat $N + 1$ dní a i -tý den přespí v i -tém kempu, splňuje podmínky ze zadání úlohy.

Nyní vyřešíme úlohu a). Na chvíli si představme, že jsme každému kempu přiřadili číslo d_i , které udává, kolikátý den nejdříve můžeme do tohoto kempu dorazit. Číslo d_i přiřazené kempu i zřejmě splňuje jednu z následujících dvou podmínek:

1. Je $d_i = 1$ a $l_i \leq K$ — do kempu lze dorazit hned první den.
2. Existuje $j < i$ takové, že $l_i - l_j \leq K$ a $d_j = d_i - 1$ (do i -tého kempu dorazíme tak, že den před tím přespíme v j -tém kempu), ale neexistuje $j < i$ takové, že $l_i - l_j \leq K$ a $d_j < d_i - 1$ (jinak by bylo možné do j -tého kempu dorazit již dříve).

Podle těchto podmínek by bylo možné spočítat všechna d_i v čase $O(N)$, náš program však d_i počítat nebude. Plán cesty splňující podmínku a) by mohl vypadat například tak, že h -tý den skupina přespí v i -tém kempu, pokud $d_i = h$ a $d_{i+1} = h + 1$; skupina navíc přespí v N -tém kempu, pokud plán cesty bez tohoto kempu nesplňuje podmínku omezující maximální vzdálenost, kterou lze urazit za jeden den. Budeme přímo vytvářet takovýto plán cesty — pokud dosud vytvořený plán cesty končí kempem ve vzdálenosti l od začátku výletu a $L - l > K$ (nelze bez přespání dorazit na konec trasy), pak další den skupina přespí v i -tém kempu, pokud $l_i - l \leq K$ a i je maximální s touto vlastností. Vytvořit program pracující podle právě popsaného postupu je triviální; časová složitost algoritmu je $O(N)$.

Dále vyřešíme úlohu b). Každému kempu přiřadíme číslo e_i , které udává, kolik by cyklisté museli zaplatit za přespání na cestě z i -tého kempu na konec výletu (počítáno včetně poplatku za přespání v i -tém kempu). Čísla e_i náš algoritmus spočítá postupně pro $i = N$ až $i = 1$; kromě těchto čísel, si pro každý z kempů uložíme informaci, do kterého následujícího kempu se z něj máme vydat, abychom za noclehy zaplatili optimální cenu e_i . Čísla e_i lze spočítat podle následujícího předpisu:

1. Pokud $L - l_i \leq K$, potom $e_i = c_i$; z i -tého kempu lze dorazit na konec trasy během jednoho dne.
2. Pokud $L - l_i > K$, potom $e_i = \min_j (c_i + e_j) = c_i + \min_j e_j$, kde se minimum počítá přes všechna $j > i$ taková, že $l_j - l_i \leq K$; to j , pro které se nabývá minima, určuje pořadí kempu, ve kterém přespíme ten den, kdy vyjedeme z i -tého kempu.

První den, pokud $L > K$, přespíme v kempu s číslem i s minimálním e_i , pro který platí $l_i \leq K$. Jak bude algoritmus pracovat je nyní již jasné. Zbývá ještě určit, jak rychle lze najít j , které minimalizuje vztah v druhém bodě.

K rychlému nalezení indexu j , použijeme datovou strukturu, která se nazývá halda. Halda je datová struktura, která umožňuje v konstantním čase určit nejmenší z prvků v haldě; prvek do haldy přidat nebo vyjmout nejmenší prvek umí v čase logaritmickém v počtu prvků obsažených v haldě. V haldě si budeme udržovat čísla kempů seřazená podle hodnot e_i ; na začátku budeme mít v haldě navíc konec trasy, jehož hodnotu budeme považovat za rovnu nule (bude nejmenším prvkem haldy). Nalezení vhodného j bude probíhat následovně: Zjistíme, zda nejmenší prvek haldy je od i -tého kempu vzdálený nejvýše o K — pokud ano, našli jsme příslušné j , v opačném případě z haldy odstraníme tento prvek a celý postup zopakujeme. Poté do haldy přiřadíme i -tý kemp. Za předpokladu logaritmického času přidání prvku do haldy a vyjmutí nejmenšího prvku z haldy je celková doba běhu algoritmu $O(N \log N)$.

Haldu budeme reprezentovat v poli. Bude-li v haldě n prvků, pak její prvky budou uloženy v poli na pozicích s čísly 0 až $n - 1$. Pro prvek x s indexem k budeme prvky na pozicích $2k + 1$ a $2k + 2$ nazývat syny prvku x a prvek x budeme nazývat otcem těchto prvků. Všimněte si, že každý prvek, mimo prvek na pozici 0, má právě jednoho otce. Při práci s haldou budeme dodržovat následující invariant: Každý prvek je větší než jeho otec. Nejmenším prvkem v haldě je proto prvek na nulté pozici; zjištění nejmenšího prvku haldy lze tedy provést v konstantním čase. Přidání prvku do haldy bude probíhat následovně: Je-li v haldě n

prvků, pak nový prvek umístíme na pozici n ; pokud je jeho otec větší, vyměníme přidávaný prvek s jeho otcem a celý postup opakujeme tak dlouho, dokud nový prvek není nultým prvkem nebo jeho otec není menší než on. V každém kroku se index nového prvku v poli zmenší alespoň na polovinu a tedy se po logaritmicky mnoha krocích zastavíme. Odebrání prvku z haldy bude probíhat podobně: Je-li v haldě n prvků, pak nejmenší prvek haldy nahradíme prvkem z $(n-1)$ -té pozice; tento prvek porovnáme s oběma jeho syny a popřípadě ho zaměníme s menším z obou jeho synů. Skončíme, pokud je tento prvek menší než oba jeho synové. Protože se v každém kroku posuneme na prvek s alespoň dvojnásobným indexem, odebrání nejmenšího prvku z haldy bude trvat čas logaritmický v počtu prvků v haldě.

P – I – 4

a) Mějme zadáno nějaké dvojkové číslo $\langle x_1, \dots, x_n \rangle$, o němž máme rozhodnout, zda je dělitelné pěti. Sestrojíme sadu dlaždic, jíž bude možno vydláždit pouze zeď o jednom řádku, a to tak, aby barva pravé hrany i -té dlaždice (tu budeme značit p_i) odpovídala zbytku po dělení dvojkového čísla $\langle x_1, \dots, x_i \rangle$ pěti. Když navíc zvolíme barvu pravého okraje zdi tak, aby odpovídala zbytku 0, půjde zeď vydláždit právě tehdy, je-li zadané číslo dělitelné pěti, a to je přesně to, co potřebujeme.

Použijeme dlaždice následujících typů:

$$T = \left\{ \begin{array}{|c|} \hline \begin{array}{c} y \\ \diagup \quad \diagdown \\ x \quad z \\ \bullet \end{array} \\ \hline \end{array} ; 0 \leq x \leq 4, 0 \leq y \leq 1, z = (2x + y) \bmod 5 \right\},$$

levý i pravý okraj budou mít barvu 0 a dolní okraj barvu $\bullet$. Jelikož barva $\bullet$ se nevyskytuje na horní hraně žádné dlaždice, musí být každé korektní vydláždění tvořeno jediným řádkem. Zbývá dokázat, že barvy pravých hran dlaždic odpovídají zbytkům, což učiníme indukcí:

- ▷ $p_1 = \langle x_1 \rangle = \langle x_1 \rangle \bmod 5$ (existuje právě jedna dlaždice, která může být na prvním políčku — ta, která má na levém okraji nulu a na horním okraji x_1).
- ▷ Je-li $p_i = \langle x_1, \dots, x_i \rangle \bmod 5$, může být na $i + 1$ -ním políčku pouze jediná dlaždice (mající na levém okraji p_i a na horním okraji x_{i+1}) a její pravý okraj má barvu $p_{i+1} = (2p_i + x_{i+1}) \bmod 5 = (2(\langle x_1, \dots, x_i \rangle \bmod 5) + x_{i+1}) \bmod 5 = (2\langle x_1, \dots, x_i \rangle + x_{i+1}) \bmod 5 = \langle x_1, \dots, x_{i+1} \rangle \bmod 5$.

Z toho plyne, že popsany dlaždicový program řeší zadanou úlohu se složitostí $O(1)$.

b) Použijeme následující typy dlaždic:

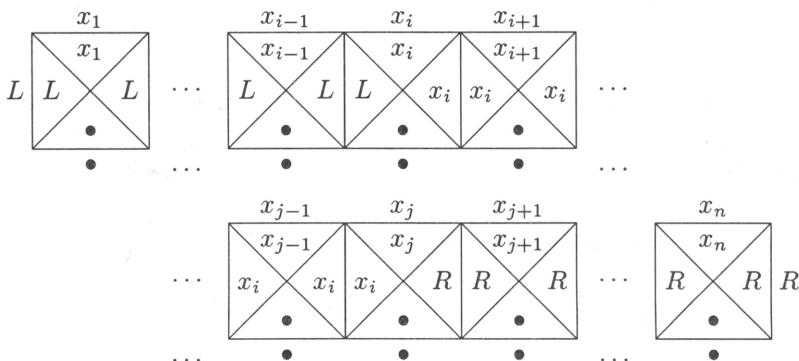
$$\triangleright \text{„levé“ dlaždice } \left\{ \begin{array}{c} x \\ L \quad x \\ \bullet \end{array} ; 0 \leq x \leq 9 \right\},$$

$$\triangleright \text{„pravé“ dlaždice } \left\{ \begin{array}{c} y \\ x \quad R \\ \bullet \end{array} ; 0 \leq x, y \leq 9, x \neq y \right\},$$

$$\triangleright \text{„opakovací“ dlaždice } \left\{ \begin{array}{c} x \\ L \quad L \\ \bullet \end{array}, \begin{array}{c} x \\ R \quad R \\ \bullet \end{array}, \begin{array}{c} x \\ y \quad y \\ \bullet \end{array} ; 0 \leq x, y \leq 9 \right\}.$$

Levý okraj zdi obarvíme barvou L , pravý barvou R a spodní barvou $\bullet$. Jelikož barva $\bullet$ se nevyskytuje na horní hraně žádné dlaždice, musí být každé korektní vydláždění tvořeno jediným řádkem.

Z toho, jak jsme si typy dlaždic nadefinovali, ihned plyne, že každé vydláždění zdi musí vypadat takto:



(zleva doprava: nejprve [možno i prázdná] posloupnost dlaždic opakuji-cích L , pak jedna levá dlaždice, následuje opět několik opakovacích dlaž-dic, jedna pravá dlaždice a případně opakování R). Takové vydláždění je ovšem korektní právě tehdy, bylo-li možné nalézt indexy i a j takové, že $x_i \neq x_j$ (díky definici barev hran pravé dlaždice), takže náš dlaždicový program řeší zadanou úlohu se složitostí $O(1)$.

Poznámka. Naše řešení využívá toho, že dlaždicové programy jsou nedeterministické (to znamená, že nemají pevně definovaný průběh vý-počtu a místo toho připouštějí více různých výpočtů s tím, že odpovědí

programu je ANO, pokud existuje alespoň jeden korektní výpočet) a že nám nedeterminismus „uhodne“ polohu nějakých dvou různých prvků vstupní posloupnosti.

P – II – 1

Předpokládejme, že máme délky dřevěných tyček seříděny podle velikosti, tj. $d_1 < d_2 < \dots < d_N$. Potom trojice indexů $i < j < k$ určuje tyčky tvořící trojúhelník právě tehdy, pokud $d_k < d_i + d_j$. Zbývající dvě trojúhelníkové nerovnosti jsou totiž splněny triviálně, neboť $d_i, d_j < d_k$. Pro libovolnou dvojici indexů $i < j$ označme symbolem $l(i, j)$ největší číslo k takové, že $d_k < d_i + d_j$; speciálně tedy platí $l(i, j) \geq j$. Zvolenou dvojici indexů $i < j$ lze doplnit na trojici $i < j < k$ určující trojúhelník právě těmi k , pro která platí $j < k \leq l(i, j)$. Pro pevnou dvojici indexů i a j tedy existuje právě $l(i, j) - j$ takových k , že tyčky s indexy $i < j < k$ tvoří trojúhelník. K určení počtu trojúhelníků proto stačí spočítat hodnoty $l(i, j)$ pro všechna $i < j$ a sečíst výrazy $l(i, j) - j$.

Z definice $l(i, j)$ plyne, že $N = l(i, N) \geq l(i, N-1) \geq l(i, N-2) \geq \dots \geq l(i, i+1)$. Náš program bude pracovat následovně: Pro každé i spočteme hodnoty $l(i, N-1), l(i, N-2), \dots, l(i, i+1)$ a současně budeme počítat součet $(l(i, N-1) - (N-1)) + \dots + (l(i, i+1) - (i+1))$, který představuje počet trojúhelníků, jejichž nejkratší strana má délku d_i . Hodnotu $l(i, j)$ spočítáme tak, že hodnotu $l(i, j+1)$ budeme zmenšovat o jedničku tak dlouho, dokud $d_{l(i,j)} \geq d_i + d_j$. Protože celkový počet zmenšení o jedničku během výpočtu hodnot $l(i, N-1), l(i, N-2), \dots, l(i, i+1)$ je nejvýše $N-2$, je doba výpočtu pro jedno pevné i lineární v N . Celková doba výpočtu pro všech $N-2$ možných hodnot i je tedy $O(N^2)$. V tomtéž čase můžeme provést i úvodní seřídění zadaných délek tyček, a tedy časová složitost našeho algoritmu je $O(N^2)$ a paměťová pak $O(N)$.

```

program trojuhelnyky;                { P-II-1 }
const MAXN=1000;
var N:word; { počet tyček }
    T:longint; { počet trojúhelníků }
    i,j,k:word;
    d:array[1..MAXN] of real; { délky tyček }
    e:real;
begin
  read(N);
  for i:=1 to N do read(d[i]);
  for i:=1 to N-1 do { seřídíme délky tyček }
    for j:=i+1 to N do
      if d[i]>d[j] then

```

```

begin
  e:=d[i]; d[i]:=d[j]; d[j]:=e
end;
T:=0;
for i:=1 to N-2 do { i - nejkratší tyčka z trojice }
begin
  j:=N-1; k:=N; { j - druhá nejkratší tyčka z trojice }
  repeat
    while (j<k) and (d[k]>=d[i]+d[j]) do { podmínku (j<k) lze vypustit }
      dec(k); { hledáme nejdelší tyčku do trojice }
    T:=T+k-j;
    dec(j);
  until j=i;
end;
writeln(T);
end.

```

P – II – 2

Nejprve si rozmysleme, jak bychom zadanou úlohu řešili, kdybychom chtěli najít nejkratší cestu robota mezi dvěma zadanými políčky. Algoritmus pro řešení této úlohy je znám pod názvem prohledávání do šířky nebo též algoritmus vlny. Každému políčku v průběhu výpočtu přiřadíme číslo, jež udává minimální počet kroků, které robot potřebuje k přemístění z počátečního políčka na uvažované políčko. Algoritmus pracuje ve fázích. Nejprve počátečnímu políčku přiřadí nulu. V i -té fázi přiřadí číslo i všem políčkům, která sousedí s nějakým políčkem, kterému bylo přiřazeno číslo $i - 1$ a kterým jsme dosud žádné číslo nepřidali. Je zřejmé, že takto přiřazená čísla určují minimální počet kroků potřebný k přemístění robota z počátečního políčka.

Nyní si rozmyslíme, jak lze tento algoritmus modifikovat tak, aby řešil úlohu ze zadání. V i -té fázi nebudeme číslovat políčka ve vzdálenosti i kroků, ale políčka, na které se lze přesunout cestou s i změnami směru. Přesněji v i -té fázi budeme číslovat ta políčka, která leží ve stejném řádku nebo sloupci jako některé políčko s číslem $i - 1$ a nejsou od něj v tomto řádku či sloupci oddělena překážkou. Správnost tohoto algoritmu je zřejmá. Zbývá domyslet detaily jeho implementace. Políčka si budeme uchovávat v poli tak, že políčka se stejným číslem budou tvořit souvislé úseky a políčka s nižším číslem budou předcházet políčkům s vyšším číslem. Pokud přijdeme v průběhu fáze na dosud neočíslované políčko, zařadíme ho na konec pole. V průběhu algoritmu vyjmeme vždy první nezpracované políčko z pole a prohledáme jeho řádek a sloupec. Pole, se kterým se pracuje právě popsáním způsobem, se obvykle nazývá

fronta — políčka se stavějí na konec fronty a čekají, až na ně přijde řada (budou zpracovávána). Nechť M a N jsou rozměry čtvercové sítě, potom zpracování každého z $M \times N$ políček vyžaduje čas $O(M + N)$. Celková časová složitost našeho algoritmu by tedy byla $O(M^2N + MN^2)$.

Čas potřebný k výpočtu však lze ještě zlepšit. Jeden a tentýž souvislý úsek v řádku bez překážek je totiž prohledáván několikrát — pro každé políčko z tohoto souvislého úseku jednou. Přitom by ale stačilo prohledat ho jenom z toho políčka, na které přijdeme nejdříve. Proto si budeme u každého políčka pamatovat, zda jsme prohledali souvislý úsek řádku (sloupce) bez překážek, ve kterém se toto políčko nachází. Před prohledáváním řádku (sloupce) otestujeme tento příznak a pokud jsme již příslušný souvislý úsek prohledali, tak vyjmeme ke zpracování další políčko z fronty. Všimněte si, že pro každé políčko je třeba udržovat dva příznaky — jeden pro souvislý úsek bez překážek v řádku a jeden pro úsek ve sloupci. Celkový čas strávený načítáním vstupních dat, prací s frontou a výpisem řešení je zřejmě $O(MN)$. Zbývá stanovit čas potřebný k prohledávání řádků a sloupců. Každý souvislý úsek bez překážek je prohledán právě jednou a součet délek všech souvislých úseků bez překážek je určitě nejvýše MN (tolik je totiž políček ve čtvercové síti). Prohledání jednoho souvislého úseku lze snadno provést v čase lineárním v jeho délce a tedy i celková časová složitost našeho algoritmu je $O(MN)$; paměťová složitost je též $O(MN)$.

```

program robot;                { P-II-2 }
const MAX=20;                 { maximální rozměry čtvercové sítě v místnosti }
type policko = record
    x,y : word;
    end;
var sirka, vyska: word; { rozměry místnosti }
    prekazky: array[1..MAX,1..MAX] of boolean;
    { rozložení překážek v místnosti }
    navstiveno, svisle, vodorovne: array[1..MAX,1..MAX] of boolean;
    { indikátory navštívení políček v místnosti }
    predchozi: array[1..MAX,1..MAX] of policko;
    { předchozí políčko na optimální cestě }
    start, cil: policko;
    { počáteční a cílové políčko }
    fronta: array[1..MAX*MAX] of policko;
    zpracovano, vefronte: word;
    { fronta prohledávání do šířky }
    x, y: word; { pomocné proměnné }
    i: integer;
procedure vypis( x, y: word);
    var x0, y0: word;
    k: integer;

```



```

begin
x0:=predchozi[x,y].x; { předchozí políčko na optimální cestě }
y0:=predchozi[x,y].y;
if ( x0 = predchozi[x0,y0].x ) and ( y0 = predchozi[x0,y0].y ) then
begin
{ Jsme na počátečním políčku ... }
if x0 < x then
write('Program pro robota (počáteční natočení DOLŮ):');
if x0 > x then
write('Program pro robota (počáteční natočení NAHORU):');
if y0 < y then
write('Program pro robota (počáteční natočení DOPRAVA):');
if y0 > y then
write('Program pro robota (počáteční natočení DOLEVA):');
end
else
begin
{ Nejprve vypíšeme předchozí políčko a potom směr našeho otočení }
vypis(x0,y0);
k:=(x-x0)*(y0-predchozi[x0,y0].y)-(x0-predchozi[x0,y0].x)*(y-y0);
if k > 0 then write(' <DOPRAVA>');
if k < 0 then write(' <DOLEVA>');
end;
k:=x+y-x0-y0; { Spočítáme počet kroků, které je třeba udělat }
if k < 0 then k:=-k;
while k > 0 do
begin
write(' <KROK>');
dec(k)
end
end;
begin
{ Načteme rozměry sítě, překážky, počáteční a cílové políčko }
readln(vyska,sirka);
readln(start.x,start.y);
readln(cil.x,cil.y);
for x:= 1 to vyska do
for y:= 1 to sirka do
begin
read(i);
prekazky[x,y]:= (i=1);
navstiveno[x,y]:=false;
svisle[x,y]:=false;
vodorovne[x,y]:=false;
end;
{ Test na shodu počátečního a cílového políčka }
if ( start.x = cil.x ) and ( start.y = cil.y ) then
begin
writeln('Počáteční a cílové políčko jsou stejné.');
```

```

fronta[1]:=start;
navstiveno[start.x,start.y]:=true;
predchozi[start.x,start.y]:=start;
{ Cyklus prohledávání do šířky }
while ( zpracovano < vefronte ) do
begin
    inc(zpracovano);
    x:=fronta[zpracovano].x;
    y:=fronta[zpracovano].y;
    if not vodorovne[x,y] then
        begin
{ Projedeme síť vodorovně ( řádek ) }
            vodorovne[x,y]:=true;
            i:=1;
while ( y+i <= sirka ) and not ( prekazky[x,y+i] ) do
                begin
                    vodorovne[x,y+i]:=true;
                    if not navstiveno[x,y+i] then
                        begin
                            navstiveno[x,y+i]:=true;
                            predchozi[x,y+i]:=fronta[zpracovano];
                            inc(vefronte);
                            fronta[vefronte].x:=x;
                            fronta[vefronte].y:=y+i;
                        end;
                    end;
                inc(i);
            end;
            i:=-1;
while ( y+i >= 1 ) and not ( prekazky[x,y+i] ) do
                begin
                    vodorovne[x,y+i]:=true;
                    if not navstiveno[x,y+i] then
                        begin
                            navstiveno[x,y+i]:=true;
                            predchozi[x,y+i]:=fronta[zpracovano];
                            inc(vefronte);
                            fronta[vefronte].x:=x;
                            fronta[vefronte].y:=y+i;
                        end;
                    end;
                dec(i);
            end;
        end;
    if not svisle[x,y] then
        begin
{ Projedeme síť svisle ( sloupeček ) }
            svisle[x,y]:=true;
            i:=1;
while ( x+i <= vyska ) and not ( prekazky[x+i,y] ) do
                begin
                    svisle[x+i,y]:=true;
                    if not navstiveno[x+i,y] then
                        begin
                            navstiveno[x+i,y]:=true;
                            predchozi[x+i,y]:=fronta[zpracovano];

```

```

        inc(vefronte);
        fronta[vefronte].x:=x+i;
        fronta[vefronte].y:=y;
    end;
    inc(i);
end;
    i:=-1;
while ( x+i >= 1 ) and not ( prekazky[x+i,y] ) do
    begin
        svisle[x+i,y]:=true;
        if not navstiveno[x+i,y] then
            begin
                navstiveno[x+i,y]:=true;
                predchozi[x+i,y]:=fronta[zpracovano];
                inc(vefronte);
                fronta[vefronte].x:=x+i;
                fronta[vefronte].y:=y;
            end;
        dec(i);
    end;
end;
    end;
    if not navstiveno[cil.x,cil.y] then
        begin
            { Na cílové políčko se nelze dostat ... }
            writeln('Cesta z počátečního na cílové políčko neexistuje.');
```

```

            halt
        end;
        { A vypíšeme nalezenou cestu ... }
        vypis(cil.x,cil.y);
        writeln;
    end.

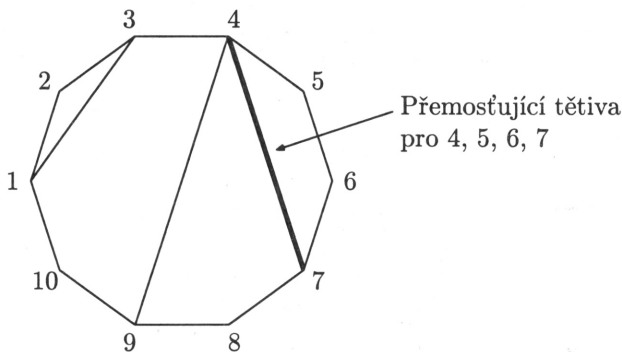
```

P – II – 3

Úlohu si nejdříve lehce přeformulujeme. Obraz je vlastně nějaký konvexní n -úhelník a stříhy jsou neprotínající se tětivy tohoto mnohoúhelníku. Úkolem je nalézt mnohoúhelník s největším počtem vrcholů, ve kterém neleží žádná tětiva (nadále budeme tento mnohoúhelník označovat jako „největší mnohoúhelník“).

Algoritmus nejdříve upraví popis každé tětivy tak, aby počáteční vrchol tětivy měl menší číslo než vrchol koncový. Pak všechny tětivy setřídí. Při třídění se porovnává nejdříve číslo počátečního vrcholu. Pokud je u více tětív stejné, bere se obrácené pořadí čísel jejich koncových vrcholů (tedy $(1,2) < (2,3)$, ale $(1,2) > (1,3)$). Po setřídění začne algoritmus hledat největší mnohoúhelník. Algoritmus při hledání využívá toho, že v každém mnohoúhelníku (až na jeden) existuje právě jedna tětiva „přemošťující“ stranu $(1,n)$. Problematickým mnohoúhelníkem je ten, který

stranu $(1, n)$ obsahuje. Problémy s tímto mnohoúhelníkem řešíme tak, že uvažujeme pomocnou tětivu $(1, n)$. „Přemosťující“ tětiva bude mít ze všech tětív a stran mnohoúhelníku nejmenší počáteční číslo vrcholu a největší koncové číslo vrcholu. (Kdyby měla nějaká tětiva větší koncové i počáteční číslo vrcholu, musela by „přemosťující“ tětivu někde protnout. Stejně tak pokud by byla obě čísla vrcholů menší. Pokud by bylo počáteční číslo menší a koncové větší, nebyla by zase naše tětiva „přemosťující“.) Také platí, že každá tětiva je „přemosťující“ pro právě jeden mnohoúhelník (mногоúhelník ohraničený tětivou (i, j) bude obsahovat vrcholy i, j a ještě nějaké vrcholy z $i \dots j$, obr. 27).



Obr. 27

A nyní již k hledání největšího mnohoúhelníku. To má následující ideu: Algoritmus postupně prochází vrcholy n -úhelníku od 1 do n . Udrží si přitom zásobník, v němž jsou uloženy tětivy, u kterých prošel jejich počátečním vrcholem, ale dosud ne jejich koncovým vrcholem. Jsou to tedy „přemosťující“ tětivy pro dosud neuzavřené mnohoúhelníky. U každé tětivy na zásobníku si také udržuje dosud napočítaný počet vrcholů pro mnohoúhelník omezený danou tětivou. Protože aktuální vrchol vždy patří k mnohoúhelníku omezenému nejpozději začínající „přemosťující“ tětivou, stačí vždy upravovat jen počet vrcholů u tětivy na vrcholu zásobníku.

Konkrétní implementace hledání: Vždy když se v algoritmu posuneme do dalšího vrcholu, odebereme ze zásobníku tětivy, které v tomto vrcholu končí. Prošli jsme totiž všechny vrcholy, které mohly ležet v mnohoúhelnících ohraničených těmito tětivami. K těmto tětivám už tedy byly spočítány počty vrcholů v jimi ohraničených mnohoúhelnících, a tak stačí podle těchto hodnot upravit dosud nalezené maximum. Při každém odebrání tětivy ze zásobníku algoritmus přičte jedna k počtu vrcholů u tětivy

na vrcholu zásobníku, protože do příslušného mnohoúhelníku se dostal i aktuální vrchol. Pak přidá všechny tětivy začínající v daném vrcholu do zásobníku. Počty vrcholů u nových tětív nastavuje na jedna, protože se musí započítat aktuální vrchol. Díky setřídění tětív stačí pouze tětivy po řadě odebírat z pole, dokud je jejich počáteční vrchol shodný s aktuálním. Setřídění také zajistí, že z tětív začínajících v aktuálním vrcholu budou ty, které skončí později, vloženy do zásobníku dříve. Když jsou přidány všechny tětivy začínající v aktuálním vrcholu, přičte se k tětivě na vrcholu zásobníku jedna za vrchol, do kterého se algoritmus přesouvá.

Právě uvedený algoritmus můžeme ještě zrychlit. Stačí si uvědomit, že je zbytečné posouvat se po obvodu pouze po jednom vrcholu. Stačí nám vlastně jen obejít vrcholy, ve kterých nějaká tětíva začíná nebo končí. To, o kolik se máme posunout, snadno zjistíme jako minimum z konce tětivy na vrcholu zásobníku a počátku první dosud nezařazené tětivy. Získáme tak algoritmus s časovou složitostí $O(k \log k)$. Čas $O(k \log k)$ totiž strávíme tříděním. Samotný průchod n -úhelníkem nám zabere pouze $O(k)$, protože každou tětívu pouze jednou přidáme na zásobník a jednou ji z něj odebereme. Počty vrcholů upravujeme dohromady pouze $2k$ -krát. Správnost algoritmu byla ukázána v popisu.

```

program Pepik; { P-II-3 }
const
  MAXK = 100;
type
  Cut = record
    a, b : Integer;
  end;
  CutA = Array[1..MAXK] of cut;
var
  n, k : Integer;      {Počet vrcholů; Počet stříhů}
  c : CutA;            {Popis jednotlivých stříhů}

{Načte vstup}
procedure ReadInp;
var
  i, tmp : Integer;
begin
  Write('Zadejte pocet vrcholu a pocet strihu: ');
  Read(n, k);
  Write('Zadejte jednotlivé strihy: ');
  {Načte popis stříhu}
  for i := 1 to k do begin
    Read(c[i].a, c[i].b);
    {První číslo bude menší}
    if c[i].a > c[i].b then begin
      tmp := c[i].a;
      c[i].a := c[i].b;
    end;
  end;
end;

```

```

    c[i].b := tmp;
end;
end;
{Přidáme pomocnou třetivou}
Inc(k);
c[k].a := 1;
c[k].b := n;
end;

{Porovná dva stříhy}
function CmpCut(a, b : cut) : ShortInt;
begin
    if (a.a < b.a) or ((a.a = b.a) and (a.b > b.b)) then
        CmpCut := -1
    else if (a.a = b.a) and (a.b = b.b) then
        CmpCut := 0
    else
        CmpCut := 1;
    end;
end;

{Setřídí pole se stříhy QuickSortem}
procedure SortCut(d, u : Integer);
var
    m, tmp : cut;           {Pivot}
    i, j : Integer;
begin
    m := c[(d+u) div 2]; {Vybereme pivota}
    i := d; j := u;
    while i <= j do begin
        {Nalezneme prvky ve špatných částech}
        while CmpCut(c[i], m) = -1 do
            Inc(i);
        while CmpCut(c[j], m) = 1 do
            Dec(j);
        if i <= j then begin
            {Zaměníme prvky ve špatných částech}
            tmp := c[i];
            c[i] := c[j];
            c[j] := tmp;
            Inc(i);
            Dec(j);
        end;
    end;
    if i < u then {Je co třídít v pravé části?}
        SortCut(i, u);
    if d < j then {Je co třídít v levé části?}
        SortCut(d, j);
end;

{Nalezne největší mnohoúhelník}
function FindMax(n, d, u : Integer) : Integer;
var
    StackC : Array[1..MAXK] of Integer;
    StackN : Array[1..MAXK] of Integer;

```

```

AV, SP, AChord, Max : Integer;
begin
  SP := 0;
  AV := c[1].a;
  AChord := 1;
  Max := 0;
  while True do begin
    {Končí tu nějaký mnohoúhelník?}
    while (SP > 0) and (c[StackC[SP]].b = AV) do begin
      if StackN[SP] > Max then {Je největší?}
        Max := StackN[SP];
      Dec(SP);
      if SP > 0 then
        Inc(StackN[SP]); {Přidáme vrchol za právě ukončenou tětivu}
      end;
      if AV = n then      {Už jsme prošli celý mnohoúhelník?}
        break;
      {Začíná zde nějaký mnohoúhelník?}
      while (AChord <= k) and (AV = c[AChord].a) do begin
        Inc(SP);
        StackC[SP] := AChord;
        StackN[SP] := 1;
        Inc(AChord);
      end;
      {Začíná dříve nějaká tětiva, než končí jiná?}
      if (AChord <= k) and (c[AChord].a < c[StackC[SP]].b) then begin
        Inc(StackN[SP], c[AChord].a - AV);
        AV := c[AChord].a;
      end
      else begin {Nějaká tětiva nejdříve končí}
        Inc(StackN[SP], c[StackC[SP]].b - AV);
        AV := c[StackC[SP]].b;
      end;
    end;
    FindMax := Max;
  end;

begin
  ReadInp;      {Načte vstup}
  SortCut(1, k); {Setřídíme pole se stříhy}
  {Nalezne největší část a vypíše ji}
  WriteLn('Nejvetsi cast ma ', FindMax(n, 1, k), ' vrcholu.');
```

```

end.
```

P – II – 4

Využijeme jednoduchého pozorování:

- ▷ Posloupnost $x_1, \dots, x_n$ je symetrická právě tehdy, když $x_1 = x_n$ a posloupnost $x_2, \dots, x_{n-1}$ je symetrická.
- ▷ Jednoprvková i prázdná posloupnost jsou vždy symetrické.

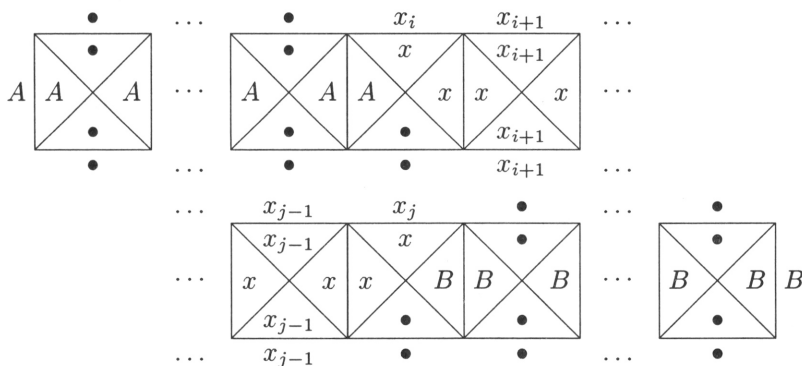
Sestrojíme dlaždicový program, který bude připouštět právě taková vydláždění, v nichž každý řádek ověří rovnost krajních prvků posloupnosti a odstraní je (přepíše na barvu $\bullet$), přičemž poslední řádek akceptuje buďto prázdnou, nebo jednoprvkovou posloupnost. Takový program odpovídá ANO právě na symetrické vstupní posloupnosti: pokud je posloupnost $x_1, \dots, x_n$ symetrická, první řádek z ní odstraní x_1 a x_n , druhý x_2 a x_{n-1} atd. až poslední řádek akceptuje buďto prostřední prvek původní posloupnosti (měla-li lichou délku), nebo prázdnou posloupnost (byla-li její délka sudá). A opačně: Pokud program posloupnost akceptuje, pak podle prvního řádku je $x_1 = x_n$, podle druhého $x_2 = x_{n-1}$ atd., tudíž je zadaná posloupnost symetrická. Proto náš program řeší zadanou úlohu se složitostí $O(n)$.

Abychom dosáhli tohoto cíle, použijeme následující sadu dlaždic:

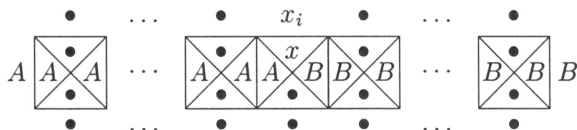
$$T = \left\{ \begin{array}{|c|c|} \hline \bullet & \\ \hline A & x \\ \hline \bullet & \\ \hline \end{array}, \begin{array}{|c|c|} \hline A & x \\ \hline & \bullet \\ \hline & \\ \hline \end{array}, \begin{array}{|c|c|} \hline x & y \\ \hline & x \\ \hline & y \\ \hline \end{array}, \begin{array}{|c|c|} \hline x & x \\ \hline & \bullet \\ \hline & \\ \hline \end{array}, \begin{array}{|c|c|} \hline \bullet & \\ \hline B & B \\ \hline \bullet & \\ \hline \end{array}, \begin{array}{|c|c|} \hline A & x \\ \hline & \bullet \\ \hline & \\ \hline \end{array} ; 0 \leq x, y \leq 9 \right\},$$

levý okraj zdi barvy A , pravý barvy B a dolní barvy $\bullet$.

Každý korektně vydlážděný řádek musí vypadat buďto takto:



(to odpovídá kontrole a odstranění krajních prvků posloupnosti), anebo takto:



(takový řádek akceptuje libovolnou jednoprvkovou posloupnost; posloupnost prázdná — taková, jejíž všechny prvky již byly přepsány na $\bullet$ — je akceptována přímo dolním okrajem zdi).

Poznámka. Časové složitosti lepší než lineární již není možno dosáhnout. Idea důkazu: Z definice vydláždění víme, že zeď sudé šířky je možno vydláždit právě tehdy, je-li možno vydláždit její levou polovinu i její pravou polovinu tak, aby v každém řádku měly dlaždice, jimiž se tyto poloviny dotýkají, stejnou barvu společné hrany (těmto hranám budeme říkat prostřední sloupec). Pro každý začátek posloupnosti $x_1, \dots, x_{n/2}$ ovšem existuje právě jedno doplnění prvky $x_{n/2+1}, \dots, x_n$ takové, že $x_1, \dots, x_n$ je symetrická posloupnost. Kdyby nějakým dvěma různým začátky $x_1, \dots, x_{n/2}$ a $y_1, \dots, y_{n/2}$ odpovídalo stejné obarvení středního sloupce, pak by po doplnění prvky $x_{n/2}, \dots, x_1$ vydláždění pravé poloviny existovalo buď pro oba začátky, nebo pro žádný (závisí totiž pouze na pravé polovině vstupu a středním sloupci), přestože pro první začátek má existovat a pro druhý nikoliv. Proto různých obarvení středního sloupce (těch je $\leq b^h$, kde b je počet barev vyskytujících se v programu a h maximální výška zdi, tedy složitost programu) musí být minimálně tolik, kolik je možných začátků posloupnosti ($10^{n/2}$), a tak musí být

$$h \geq \log_b 10^{n/2} = n \cdot \frac{\log_b 10}{2}.$$

To znamená, že složitost libovolného dlaždicového programu řešícího naši úlohu musí být alespoň lineární.

P – III – 1

Nejdříve je třeba si uvědomit, že jednotlivé souřadnice pozic věží je možno volit nezávisle na sobě. To platí proto, že volba sloupce nijak neomezí rozsah řádek, ve kterém může být věž umístěna. Můžeme tedy nejdříve pro každou věž zvolit sloupec, ve kterém se bude nacházet, a pak pro každou věž zvolit řádek. Převvedli jsme tak původní úlohu do jednoho rozměru. Jednotlivé obdélníky se promítnou na intervaly a v každém intervalu chceme nalézt číslo i (umístění věže) takové, aby se čísla žádných dvou intervalů neshodovala.

Čísla budeme hledat následujícím způsobem: Budeme postupně procházet čísla od 1 do N a budeme si udržovat informaci, které intervaly obsahují dané číslo a ještě nemají žádné číslo přiděleno. Z těchto intervalů vybereme ten, který končí nejdříve, a přidělíme mu aktuální číslo. Pokud vybraný interval již neobsahuje přidělované číslo (skončil dříve), úloha nemá řešení. Přírná implementace této myšlenky je samozřejmě možná, ale pomalá ($O(N^2)$). My implementaci zrychlíme následujícím způsobem:

Abychom mohli rychle upravovat informace o tom, které intervaly obsahují dané číslo, seřídíme si je nejdříve vzestupně podle jejich počátku. Abychom byli dále schopni rychle nalézt interval, který končí nejdříve, budeme si intervaly obsahující dané číslo uchovávat v haldě srovnané podle konců intervalů. Přidělování čísel tedy ve vylepšené implementaci probíhá následovně: Procházíme všechna čísla od 1 do N . Pro každé číslo přidáme do haldy všechny intervaly začínající na daném čísle. Pak z haldy (pokud je neprázdná) odebereme minimum (tj. nejdříve končící interval) a přidělíme mu aktuální číslo. Pokud interval již neobsahuje aktuální číslo, řekneme, že požadované rozestavení věží neexistuje. S těmito vylepšeními má algoritmus časovou složitost $O(N \cdot \log N)$ a paměťovou složitost $O(N)$.

Nechť máme nějaké korektní rozestavení věží R . Indukcí dokážeme, že toto rozestavení lze upravit na takové, jaké navrhne náš algoritmus, a tím tedy ukážeme, že náš algoritmus nalezne korektní rozestavení věží.

- ▷ Počátek indukce: Nechť C je nejmenší číslo, které náš algoritmus chce přidělit nějakému (vlastně prvnímu) intervalu I . Z chování našeho algoritmu je zřejmé, že v žádném rozestavení se nižší číslo vyskytovat nemůže. Pokud se v R nevyskytuje ani číslo C , můžeme R upravit tak, že I dáme číslo C . Tím jistě získáme korektní rozestavení, které navíc má pro první interval I přiděleno stejné číslo, jaké by navrhl náš algoritmus. Pokud je v R číslo C již přiděleno nějakému intervalu J , můžeme snadno intervalu J přidělit číslo přidělené intervalu I a intervalu I přiřadit C . Protože interval I končil dříve než interval J , jistě máme opět korektní rozestavení.
- ▷ Indukční krok: Z indukčního předpokladu víme, že existuje rozestavení, které se shoduje s rozestavením navrhovaným naším algoritmem v prvních k číslech. Chceme ukázat, že existuje rozestavení, které se shoduje v prvních $k + 1$ číslech. Protože myšlenka důkazu je naprosto stejná jako v počátku indukce, necháváme tuto část důkazu na čtenáři.

Program je přímou implementací algoritmu. Halda je implementována v poli, kde prvek na pozici i má syny na pozicích $2i$ a $2i + 1$.

```

program Veze;                                {P-III-1}
const
  MAXN = 100;
type
  {Popis jednoho intervalu}
  Int = record
    s, e : Integer;    {Počátek a konec intervalu}

```

```

        n : Integer;           {Číslo věže, které interval patří}
    end;
    {Popis jednoho bodu}
    Point = array [1..2] of Integer;
    {Popis jednoho obdélníku}
    Rectangle = record
        a, b : Point;   {Levý horní a pravý dolní roh}
    end;
    {Intervaly v jedné souřadnici}
    IntList = array[1..MAXN] of Int;
    CmpProc = function(A, B : Int) : ShortInt;

var
    N : Integer;   {Pocet vezi}
    Rec : Array[1..MAXN] of Rectangle;   {Obdélníky}
    T : Array[1..MAXN] of Point;         {Umístění věží}

{Načte vstup}
procedure ReadInp;
var
    i : Integer;
begin
    Write('Pocet vezi: ');
    Read(N);
    WriteLn('Souradnice obdelniku:');
    for i := 1 to N do
        Read(Rec[i].a[1], Rec[i].a[2], Rec[i].b[1], Rec[i].b[2]);
    end;

{Přidá interval do haldy}
procedure AddHeap(var N: Integer; var H: IntList; I: Int; Cmp: CmpProc);
var
    A : Integer;
    Tmp : Int;
begin
    Inc(N);
    H[N] := I;
    A := N;
    while A <> 1 do begin {Nejsme na vrcholu}
        if Cmp(H[A], H[A div 2]) <> -1 then {Je splněna podmínka haldy?}
            break;
        {Zaměníme prvky, aby byla podmínka splněna}
        Tmp := H[A];
        H[A] := H[A div 2];
        H[A div 2] := Tmp;
        A := A div 2; {Posun o úroveň výše}
    end;
end;

{Odebere z haldy minimum}
procedure
    GetHeapMin(var N: Integer; var H: IntList; Cmp: CmpProc; var Res: Int);
var
    A, M : Integer;
    Tmp : Int;

```

```

begin
  Res := H[1];
  H[1] := H[N];
  Dec(N);
  A := 1;
  while A * 2 < N do begin      {Nejsme na dně?}
    {Nalezneme menšího ze synů}
    if Cmp(H[A*2], H[A*2+1]) = -1 then
      M := A*2
    else
      M := A*2+1;
    if Cmp(H[M], H[A]) <> -1 then {Podmínka haldy splněna?}
      break;
    {Zaměníme prvky, aby byla podmínka splněna}
    Tmp := H[M];
    H[M] := H[A];
    H[A] := Tmp;
    A := M; {Posun o úroveň níže}
  end;
end;

{Porovná dva intervaly podle počátku}
function CmpInt(a, b : Int) : ShortInt; far;
begin
  if (a.s < b.s) or ((a.s = b.s) and (a.e < b.e)) then
    CmpInt := -1
  else if (a.s = b.s) and (a.e = b.e) then
    CmpInt := 0
  else
    CmpInt := 1;
end;

{Setřídí pole intervalů}
procedure SortInts(var A : IntList);
var
  C, i : Integer;
  H : IntList;
begin
  C := 0;
  for i := 1 to N do
    AddHeap(C, H, A[i], CmpInt);
  for i := 1 to N do
    GetHeapMin(C, H, CmpInt, A[i]);
end;

{Srovná intervaly podle konce}
function CmpBack(a, b : Int) : ShortInt; far;
begin
  if a.e < b.e then
    CmpBack := -1
  else if a.e = b.e then
    CmpBack := 0
  else
    CmpBack := 1;
end;

```

```

end;

{Spočte jednu souřadnici pro každou věž}
function CountCoord(c : Integer) : Boolean;
var
  Ints : IntList; {Intervaly}
  i, a : Integer;
  ActInts : IntList; {Halda intervalů k uplatnění}
  ActIntsN : Integer; {Počet intervalů v haldě}
  Tmp : Int;
begin
  {Vytvoří pole intervalů z obdélníku}
  for i := 1 to N do begin
    Ints[i].s := Rec[i].a[c];
    Ints[i].e := Rec[i].b[c];
    Ints[i].n := i;
  end;
  SortInts(Ints); {Setřídí intervaly}
  {Určí souřadnici věží}
  ActIntsN := 0;
  a := 1;
  for i := 1 to N do begin
    {Přidá do haldy všechny intervaly začínající na aktuální pozici}
    while (a <= N) and (Ints[a].s = i) do begin
      AddHeap(ActIntsN, ActInts, Ints[a], CmpBack);
      Inc(a);
    end;
    if ActIntsN > 0 then begin
      {Vybereme nejdříve končící interval}
      GetHeapMin(ActIntsN, ActInts, CmpBack, Tmp);
      if Tmp.e < i then begin {Už skončil?}
        CountCoord := False;
        Exit;
      end;
      T[Tmp.n][c] := i;
    end;
  end;
  CountCoord := True;
end;

{Vytiskne souřadnice věží}
procedure Print;
var
  i : Integer;
begin
  WriteLn('Souřadnice vezi jsou:');
  for i := 1 to N do
    WriteLn(T[i][1], ' ', T[i][2]);
  end;
end;

begin
  ReadInp; {Načte vstup}
  if CountCoord(1) and CountCoord(2) then
    {Určí souřadnice věží - vešly se?}

```

```

Print
else
  WriteLn('Rozmístění vezi neexistuje.');
```

end.

P – III – 2

Nejprve ukážeme, že pro každé přirozené číslo N existuje přirozené číslo x , jehož ciframi jsou pouze číslice 0 a 1 a které je dělitelné číslem N . Označme $x_1 = 1$, $x_2 = 11$, $x_3 = 111$ atd. Dále označme jako m_i číslo $x_i \bmod N$. Čísla m_i mohou nabývat pouze hodnot od 0 do $N - 1$ a proto alespoň dvě z čísel m_1 až m_{N+1} musí být stejná — necht' např. $m_i = m_j$ ($i < j$). Potom ale číslo $x_j - x_i$ je dělitelné číslem N , neboť $(x_j - x_i) \bmod N = m_j - m_i = 0$. Ciframi čísla $x_j - x_i$ jsou zřejmě pouze číslice 0 a 1 a tedy číslo $x_j - x_i$ splňuje podmínky ze zadání úlohy.

Nadále budeme uvažovat pouze ta čísla, jejichž dekadický zápis je tvořen pouze číslicemi 0 a 1. Předchůdcem čísla x nazveme číslo $x \div 10$, tedy číslo x bez své poslední cifry; naopak následníky čísla x nazveme čísla $10x$ a $10x+1$, tedy ta čísla, která lze vytvořit přidáním jedné cifry na konec čísla x . Číslo x budeme nazývat minimálním číslem pro zbytek z , jestliže $x \bmod N = z$, v dekadickém zápisu čísla x se vyskytují pouze číslice 0 a 1 a x je nejmenší číslo s těmito vlastnostmi. Nyní si dokážeme jednoduché lemma:

Lemma. *Necht' x je minimální číslo pro zbytek z , necht' x' je předchůdcem x ; označme $z' = x' \bmod N$. Potom x' je minimální číslo pro zbytek z' .*

DŮKAZ. Postupujme sporem, tedy předpokládejme, že x' není minimální číslo pro zbytek z' , tj. že existuje číslo $x'' < x'$ takové, že $x'' \bmod N = x' \bmod N$. Necht' c je poslední cifra čísla x . Protože $(10x' + c) \bmod N = z$ a $x' \bmod N = x'' \bmod N$, musí nutně platit i $(10x'' + c) \bmod N = z$. Ale potom by číslo $x = 10x' + c$ nemohlo být minimální číslo pro zbytek z , neboť číslo $10x'' + c$ má požadované vlastnosti a je menší — tedy nemůže existovat x'' a tedy x' je minimální číslo pro zbytek z' .

Toto lemma nám dává návod, jak počítat minimální čísla pro různé zbytky. Pro daný zbytek z lze spočítat minimální číslo x tak, že budeme postupně testovat v rostoucím pořadí následníky minimálních čísel pro ostatní zbytky a první následník y nějakého minimálního čísla, pro kterého platí $y \bmod N = z$ je zřejmě hledané minimální číslo pro zbytek z .

Tímto postupem lze vygenerovat minimální čísla pro všechny zbytky, pro které existují: Položme $l_1 = 1$ a pokud už známe $l_1, \dots, l_k$, položme

l_{k+1} rovno nejmenšímu následníkovi některého z čísel $l_1, \dots, l_k$ takovému, že $l_{k+1} \bmod N$ je různý od všech $l_i \bmod N$ pro $1 \leq i \leq k$. Z předchozích úvah ale vyplývá, že jednotlivá čísla l_i jsou minimální čísla pro zbytky $z_i = l_i \bmod N$. Navíc podle prvního odstavce je jedno z čísel z_i rovno nule.

Náš algoritmus bude pracovat přesně podle výše uvedeného popisu. V poli **fronta** si budeme pamatovat hodnoty z_i a na m -té pozici v poli **predchozi** si budeme pamatovat zbytek předchůdce od minimálního čísla pro zbytek m — z hodnot v tomto poli jsme schopni jednoduše zkonstruovat minimální číslo pro zadaný zbytek. Postupně budeme brát hodnoty z_i z pole **fronta**, spočítáme $(10 * z_i) \bmod N$ a $(10 * z_i + 1) \bmod N$ a pokud jsme dosud nenašli číslo, jehož zbytek by byla jedna z těchto hodnot, tak tento zbytek přidáme na konec pole **fronta** a příslušně zmodifikujeme pole **predchozi**. Časová a paměťová složitost algoritmu je zřejmě $O(N)$.

```

program nulajed; { P-III-2 }
const MAXN=1000;
var N:word; { zadané číslo N }
    predchozi:array[0..MAXN-1] of integer;
    { zbytek předchůdce minimálního čísla s daným zbytkem }
    { Hodnoty se zvláštním významem:
    -2 ... dosud nenalezeno minimální číslo pro tento zbytek
    -1 ... nemá předchůdce (číslo 1)
    }
    fronta:array[0..MAXN-1] of word;
{ fronta použitá pro generování minimálních čísel }
    ukazatel:word;
{ právě zpracovávaný prvek v poli fronta }
    vefronte:word;
{ počet prvků v poli fronta }
procedure pridej(puvodni,novy:word);
begin { přidá minimální prvek pro zbytek novy do fronty;
      jeho předchůdce je minimální pro zbytek původní }
    if predchozi[novy]<>-2 then exit;
    fronta[vefronte]:=novy;
    inc(vefronte);
    predchozi[novy]:=puvodni;
end;
procedure vypis(p:word);
begin { vypíše číslo se zadaným zbytkem }
    if predchozi[p]>=0 then
        begin
            vypis(predchozi[p]);
            if (10*predchozi[p]) mod N=p then write(0) else write(1)
        end
    else
        write(1)
    end;
begin

```

```

readln(N); { načteme číslo N a inicializujeme pole predchozi }
for ukazatel:=0 to N-1 do predchozi[ukazatel]:=-2;
fronta[0]:=1 mod N;
predchozi[fronta[0]]:=-1;
ukazatel:=0; vefronte:=1;
while (predchozi[0]=-2) do
begin { generujeme čísla s různými zbytky... }
  pridej(fronta[ukazatel], (10*fronta[ukazatel]) mod N);
  pridej(fronta[ukazatel], (10*fronta[ukazatel]+1) mod N);
  inc(ukazatel);
end;
vypis(0); { vypíšeme výsledek a odřádkujeme }
writeln
end.

```

P – III – 3

Abychom nemuseli vše popisovat zbytečně složitě, zavedme si jednoduché značení: *Posloupnosti* budou vždy složeny jen z nul a jedniček a všechny budou mít n prvků. Budeme je značit tučnými písmeny a jejich prvky písmeny s indexy, tedy například $\mathbf{x}$ je posloupnost obsahující prvky $x_1, \dots, x_n$. Počtu jedniček v posloupnosti $\mathbf{x}$ budeme říkat její *váha* a značit $\#\mathbf{x}$.

Posloupnost $\mathbf{y}$ je setříděním posloupnosti $\mathbf{x}$ právě tehdy, když platí současně následující dvě podmínky:

1. $y_1 \leq \dots \leq y_n$ (je to neklesající posloupnost)
2. $\#\mathbf{y} = \#\mathbf{x}$ ($\mathbf{x}$ a $\mathbf{y}$ obsahují stejný počet jedniček a jelikož mají stejnou délku, tak i stejný počet nul, čili se navzájem liší pouze pořadím prvků)

Zabývejme se nejprve tím, jak ověřit druhou podmínku. Mohli bychom postupovat podobně, jako u příkladu v zadání oblastního kola: v každém řádku vydláždění odstranit po jedné jedničce z obou posloupností, a to opakovat tak dlouho, až obě posloupnosti budou obsahovat pouze nuly, což snadno ověříme barvou spodního okraje zdi. Tím úlohu vyřešíme s lineární složitostí, což ovšem, jak si ukážeme, není optimální. U myšlenky postupného „zjednodušování“ testovaných posloupností však zůstaneme:

Tvrzení. $\#\mathbf{x} = \#\mathbf{y} \iff \#\mathbf{x} \bmod 2 = \#\mathbf{y} \bmod 2 \wedge \#\hat{\mathbf{x}} = \#\hat{\mathbf{y}}$, kde $\hat{\mathbf{x}}$ je posloupnost, která vznikne z posloupnosti $\mathbf{x}$ přepsáním každé druhé jedničky na nulu (to jest přepsáním první jedničky, ponecháním druhé, přepsáním třetí, ponecháním čtvrté atd.).

DŮKAZ. Pokud $\#\mathbf{x} = \#\mathbf{y}$, pak jistě $\#\mathbf{x} \bmod 2 = \#\mathbf{y} \bmod 2$, ale jelikož $\#\hat{\mathbf{x}} = \lfloor \#\mathbf{x}/2 \rfloor$ (v $\hat{\mathbf{x}}$ zbyla právě polovina jedniček z $\mathbf{x}$, případná lichá jed-

nička na konci byla odstraněna), musí platit i $\# \hat{x} = \# \hat{y}$. A naopak: pokud $\# \hat{x} = \# \hat{y}$, mohou se posloupnosti x a y lišit pouze případnou lichou poslední jedničkou, což ovšem není možné, protože váhy $\#x$ a $\#y$ jsou buďto obě liché, nebo obě sudé.

Vytvořme nejprve dlaždicový podprogram, který pro danou posloupnost x zadanou barvami horního okraje spočte $\hat{x}$ na okraji dolním a $\#x \bmod 2$ na okraji pravém (v tom smyslu, že vydláždění bude existovat právě tehdy, když barvy těchto okrajů splňují dané podmínky, a toto vydláždění bude mít jediný řádek), předpokládaje, že levý okraj má barvu 0. K tomu nám poslouží následující množina typů dlaždic:

$$T_0 = \left\{ \begin{array}{|c|c|} \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array}, \begin{array}{|c|c|} \hline 1 & 1 \\ \hline 0 & 0 \\ \hline \end{array}, \begin{array}{|c|c|} \hline 0 & 1 \\ \hline 1 & 0 \\ \hline \end{array}, \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & 1 \\ \hline \end{array} \right\}$$

Označíme-li si z_i barvu pravé hrany a y_i barvu dolní hrany i -té dlaždice vydlážděného řádku (z_0 budiž barva levé hrany první dlaždice, tedy barva levého okraje zdi, což je 0), snadno ukážeme, že:

$$\triangleright z_i = \left(\sum_{k=1}^i x_k \right) \bmod 2 \dots \text{Indukcí: pro } i = 0 \text{ platí, pro } i > 0 \text{ je}$$

$$\begin{aligned} z_i &= (z_{i-1} + x_i) \bmod 2 = \\ &= \left(\left(\sum_{k=1}^{i-1} x_k \right) \bmod 2 + x_i \right) \bmod 2 = \left(\sum_{k=1}^i x_k \right) \bmod 2. \end{aligned}$$

$\triangleright y_i = 1 \iff x_i = 1 \wedge z_{i-1} = 1$, jinými slovy právě tehdy, je-li x_i jednička, před níž byl lichý počet jedniček, čili jednička sudá — právě ta, která se má objevit v $\hat{x}$.

Zkombinujme nyní dva takové programy tak, aby pracovaly současně: jeden s posloupností x a druhý s y (posloupnosti jsou kódovány dvojicemi (x_i, y_i)), počítaly stejně kódované posloupnosti $\hat{x}$ a $\hat{y}$ a rovněž zbytky $\#x \bmod 2$ a $\#y \bmod 2$. K tomu nám stačí sestrojit množinu T_1 obsahující „součiny“ dvojic dlaždic z množiny T_0 , to znamená pro každé dvě dlaždice

$$A = \begin{array}{|c|c|} \hline c & b \\ \hline a & d \\ \hline \end{array} \quad \text{a} \quad B = \begin{array}{|c|c|} \hline g & f \\ \hline e & h \\ \hline \end{array}$$

z T_0 do T_1 přidat dlaždici

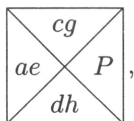
$$\begin{array}{|c|c|} \hline cg & bf \\ \hline ae & dh \\ \hline \end{array},$$

kde ae , bf , cg a dh jsou uspořádané dvojice barev. Existuje-li vydláždění řádku, který má na horní hraně dvojice $(x_1, y_1), \dots, (x_n, y_n)$, na dolní hraně $(\hat{x}_1, \hat{y}_1), \dots, (\hat{x}_n, \hat{y}_n)$, na levé $(0, 0)$ a na pravé (z_x, z_y) , dlaždicemi typů z množiny T_1 , musí díky tomu, jak jsme si tyto typy na-definovali, existovat i vydláždění řádku s horní hranou $x_1, \dots, x_n$, dolní $\hat{x}_1, \dots, \hat{x}_n$, levou 0 a pravou z_x , jakož i řádku s horní hranou $y_1, \dots, y_n$, dolní $\hat{y}_1, \dots, \hat{y}_n$, levou 0 a pravou z_y dlaždicemi typů T_0 . A opačně — existují-li tato dvě vydláždění, existuje i jejich složení sestavené z typů z množiny T_1 . Proto $z_x = \#x \bmod 2$, $z_y = \#y \bmod 2$ a spodní hrana opravdu obsahuje dvojice prvků posloupností $\hat{x}$ a $\hat{y}$.

My ovšem potřebujeme akceptovat právě ta vydláždění, u nichž $z_x = z_y$, tedy s pravým okrajem jak $(0, 0)$, tak $(1, 1)$, ale máme k dispozici pouze jedinou barvu pravého okraje zdi. Proto rozšíříme množinu T_1 na T_2 tak, že ke každému typu dlaždice z T_1 tvaru



přidáme do T_2 typ



kde P je barva pravého okraje, která se neshoduje s žádnou jinou barvou. Jelikož se tato dlaždice může vyskytnout výhradně těsně u pravého okraje (žádná dlaždice totiž nemá levou hranu barvy P , na kterou bychom mohli navázat), odpovídají korektní vydláždění pomocí této rozšířené sady dlaždic právě těm vydlážděním pomocí původní sady, u nichž je $z_x = z_y$.

Zbývá nám ještě přidat test vzestupnosti posloupnosti y , což vyřešíme přidáním dlaždic typů

$$T_z = \left\{ \begin{array}{|c|c|} \hline \mathbf{x0} & \\ \hline 0 & 0 \\ \hline \end{array}, \begin{array}{|c|c|} \hline \mathbf{x1} & \\ \hline 0 & 1 \\ \hline \end{array}, \begin{array}{|c|c|} \hline \mathbf{x1} & \\ \hline 1 & 1 \\ \hline \end{array}, \begin{array}{|c|c|} \hline \mathbf{x0} & \\ \hline 0 & P \\ \hline \end{array}, \begin{array}{|c|c|} \hline \mathbf{x1} & \\ \hline 0 & P \\ \hline \end{array}, \begin{array}{|c|c|} \hline \mathbf{x1} & \\ \hline 1 & P \\ \hline \end{array} \right\};$$

$$x \in \{0, 1\}, \mathbf{x} = \text{„tučná verze“ } x,$$

kde 00 , 01 , 10 a 11 jsou barvy dvojic kódujících zadané posloupnosti x a y a 00 , 01 , 10 a 11 analogické kódy používané všemi ostatními dosud

nadefinovanými dlaždicemi. Tím z T_2 vznikne množina typů T , o které tvrdíme, že použita s barvou levého okraje 0, barvou pravého P a barvou dolního okraje 00, řeší zadanou úlohu se složitostí $O(\log n)$, což také ihned dokážeme:

- ▷ První řádek vydláždění musí obsahovat výhradně dlaždice typů T_z , protože žádné jiné neobsahují na svých horních hranách barvy, jimiž je kódován vstup. Jak jsme již ukázali v příkladu v zadání domácího kola, typy T_z zajišťují neklesání posloupnosti y . Navíc spodní okraj tohoto řádku předává níže obě vstupní posloupnosti, pouze jinak kódované.
- ▷ Všechny ostatní řádky obsahují pouze dlaždice typů T_2 , přičemž každý řádek přepíše posloupnosti x_i a y_i zadané na svém horním okraji na $x_{i+1} = \hat{x}_i$ a $y_{i+1} = \hat{y}_i$ a ověří, zda $\#x_i \bmod 2 = \#y_i \bmod 2$. Pokud $\#x = \#y$, pak po maximálně $\lceil \log_2 n \rceil$ takových řádcích budou obě posloupnosti zredukovány na samé nuly, což odpovídá barvě dolního řádku, čili vydláždění celé zdi existuje a má hloubku $\leq 1 + \lceil \log_2 n \rceil$; pokud $\#x \neq \#y$, vydláždění nemůže existovat, protože alespoň v jednom kroku by $\#x_i \bmod 2$ nebylo rovno $\#y_i \bmod 2$ (viz Tvzení výše).

Poznámka. Lepší hloubky než $\Omega(\log n)$ již není možno dosáhnout. To můžeme zdůvodnit podobně, jako jsme v předchozím kole dokazovali, že pro ověření symetrie posloupnosti potřebujeme hloubku minimálně lineární. Opět budeme počítat možná obarvení středního sloupce — tentokrát si uvědomíme, že tato obarvení musí být rozdílná pro každé dva různé počty jedniček v $x_1, \dots, x_{n/2}$ — pokud jsou jedničky pouze mezi těmito prvky a $x_{n/2+1}, \dots, x_n$ jsou všechny nulové, musí setříděná posloupnost y obsahovat naopak ve své levé polovině samé nuly a v polovině pravé stejný počet jedniček, jako má x v polovině levé. Možných počtů jedniček mezi $x_1, \dots, x_{n/2}$ je $n/2 + 1$, možných obarvení středního sloupce $\leq b^h$, kde b je počet barev použitých v našem dlaždicovém programu (to je nějaká konstanta) a h je jeho složitost = výška středního sloupce. Z toho dostaneme:

$$b^h \geq n/2 + 1 \implies b^h > n/2 \implies h > \log_b n - \log_b 2 \implies h = \Omega(n).$$

P – III – 4

Řešení této úlohy je založeno na dynamickém programování. Pro každé slovo si budeme pamatovat, kde nejlépe zalomit předchozí řádek, když

toto slovo bude na konci řádku. Také si budeme pamatovat celkový počet trestných bodů textu při tomto zalomení. Když spočítáme nejlepší zalomení pro poslední slovo, můžeme ze zapamatovaných informací snadno zrekonstruovat, jak celý text zalámat. Z informací u posledního slova zjistíme, za kterým slovem měl být zalomen předposlední řádek. Z informací u posledního slova na předposledním řádku zjistíme, kde měl být zalomen předpředposlední řádek atd. Když víme, kde měly být jednotlivé řádky zalámány, stačí již jen správně doplnit mezi vypisovaná slova mezery. To uděláme tak, že pokud máme umístit M mezer mezi $S + 1$ slov, tak vytvoříme $M \bmod S$ oddělení slov s $(M \div S) + 1$ mezerami a $S - (M \bmod S)$ oddělení s $M \div S$ mezerami. Tím se zjevně žádná dvě oddělení slov neliší o více jak jednu mezeru a počet mezer na řádce je také správný. Formátování posledního řádku a řádku s jedním slovem jsou triviální.

A nyní již k zajímavé části algoritmu. Jak rychle spočítat, kde nejlépe zalomit předchozí řádek, když bude aktuální slovo na konci řádku? Tuto informaci budeme postupně počítat od prvního slova. U prvního slova nastavíme počet trestných bodů i místo zalomení na nula. Když máme spočteny hodnoty pro prvních N slov, začneme je počítat pro slovo $N + 1$ -ní. Jdeme od N -tého slova směrem k začátku textu. Pro každé slovo si spočteme počet trestných bodů za řádek, který začíná za ním a končí $N + 1$ -vým slovem (pokud je $N + 1$ -ní slovo posledním v textu, bude jím ukončovaná řádka řádkou poslední, a tak hodnotící funkci informujeme, že hodnotí poslední řádku). K tomuto počtu trestných bodů ještě přičteme trestné body za předchozí text (ty máme již spočteny a uloženy u slova, za kterým jsme se rozhodli řádek zlomit) a zjistíme, zda jsme získali pro $N + 1$ -ní slovo text s menším počtem trestných bodů. Pokud ano, zapamatujeme si počet bodů pro tento text a místo zalomení. Když už je poslední řádek moc dlouhý (hodnotící funkce nám vrátila ∞), tak víme, že už lepší zalomení nenajdeme. Prošli jsme už totiž všechna možná zalomení předposledního řádku a vybrali z nich to nejlepší. Můžeme tedy počítat hodnoty pro další slovo. Algoritmus má časovou složitost $O(T + W \cdot L)$, kde T je délka textu, W je počet slov a L je délka řádku. Paměťová složitost algoritmu je $O(T)$. Správnost algoritmu plyne z popisu.

Program je přímou implementací algoritmu:

```
#include <stdio.h>
#include <stdlib.h>

#define INPUT "format.in"
```

```

#define OUTPUT "format.out"

#define TEXTLEN 10000 /* Maximalni delka textu */
#define MAXWORDS 5000 /* Maximalni pocet slov v textu */
#define MAXLINELEN 100 /* Maximalni delka vstupni radky */
#define MAXLINES 5000 /* Maximalni pocet radek ve vystupu */

#define LINEPENALTY 10 /* Cena radky */
#define LASTLINESMALLPEN 3 /* Cena za maly posledni radek */
#define SINGLEWORDPEN 20 /* Penalta za jedno slovo na radku */
#define INFYPEN 30000 /* Nekonecna penalta */

typedef unsigned long price_t;

int Width; /* Sirka radku */
char Text[TEXTLEN]; /* Text nacteny ze souboru */
int Words[MAXWORDS]; /* Text rozsekany na slova */
int WCnt; /* Pocet slov */
price_t WrapPrice[MAXWORDS];
/* Ohodnoceni textu, pokud zalomime za timto slovem */
int WrapPos[MAXWORDS];
/* Pozice, kde jsme zalomili, kdyz jsme pridavali toto slovo */

/* Ohodnoti radek */
int LinePrice(int WordLen, int Words, int Last)
{
    int Pts = Width - WordLen - Words + 1;
    int Base = LINEPENALTY;

    if (Pts < 0)
        return INFYPEN;
    if (Last)
        return LINEPENALTY + (((WordLen+Words-1)*4 < Width) ?
            LASTLINESMALLPEN : 0);
    if (Words == 1)
        Base += SINGLEWORDPEN;
    return Pts * Pts + Base;
}

/* Nacte vstup a rozdeli ho na slova */
void ProcessInput(void)
{
    FILE *In;
    char Buf[MAXLINELEN]; /* Buffer na radku */
    int i, TPos = 0, WPos = 0; /* ; Pozice v bufferu na text ;
                                Cislo aktualniho slova */
    int SWPos = 0; /* Pozice pocatku slova */

    if (!(In = fopen(INPUT, "r")))
    {
        puts("Can't open input file.");
        exit(1);
    }
    /* Nacteme delku radku */

```

```

fgets(Buf, MAXLINELEN, In);
sscanf(Buf, "%d", &Width);

while (fgets(Buf, MAXLINELEN, In))
{
    for (i = 0; Buf[i] != '\0'; i++, TPos++)
    {
        if (Buf[i] == '\n')
            Buf[i] = ' ';
        Text[TPos] = Buf[i];
        if (Buf[i] == ' ')
        {
            Words[WPos++] = TPos - SWPos;
            SWPos = TPos + 1;
        }
    }
}
fclose(In);
WCnt = WPos;
}

void FindBestSep(void)
{
    int i, j;
    price_t Min, Price;
    /* Minimalni dosazene ohodnoceni; Cena aktualniho zlomu */
    int WordsLen, MinPos;
    /* Celkova delka slov na radce; Pozice minimalniho zalomeni */

    WrapPrice[0] = 0;
    WrapPos[0] = 0;

    for (i = 0; i < WCnt; i++) /* Postupne pridavame jednotlivá slova */
    {
        Min = INFTYPEN;
        MinPos = 0;
        WordsLen = 0;
        for (j = i; j >= 0; j--) /* Vyzkousime vsechna možná zalomení */
        {
            WordsLen += Words[j];
            Price = LinePrice(WordsLen, i - j + 1, i == WCnt - 1);
            if (Price == INFTYPEN) /* Uz jsem prekrocili velikost radku? */
                break;
            Price += WrapPrice[j];
            if (Price < Min)
            {
                Min = Price;
                MinPos = j;
            }
        }
        WrapPrice[i+1] = Min;
        WrapPos[i+1] = MinPos;
    }
}

```

```

/* Vytiskne dalsi slovo */
void PrintWord(FILE *Out)
{
    static int WPos = 0; /* Pozice slova k vypsani */
    int WStart;

    for (WStart = WPos; Text[WPos] != ' '; WPos++);
    fwrite(Text + WStart, 1, WPos - WStart, Out);
    WPos++;
}

/* Vytiskne radku */
void PrintLine(FILE *Out, int First, int Cnt)
{
    int i, j;
    int Len = 0, TotSpc; /* Pocet znaku na radce; Celkovy pocet mezer */
    int Spc; /* Pocet mezer v aktualni mezere */

    /* Spocteme pocet znaku ve slovech */
    for (i = 0; i < Cnt; i++)
        Len += Words[First+i];
    TotSpc = Width - Len;
    if (Cnt == 1) /* Jedno slovo? */
    {
        for (j = 0; j < TotSpc; j++)
            fputc(' ', Out);
    }
    else
        for (i = 0; i < Cnt-1; i++)
        {
            PrintWord(Out); /* Vytiskne dalsi slovo */
            /* Spocteme a vytiskneme potrebny pocet mezer */
            Spc = TotSpc / (Cnt - 1) + (i < TotSpc % (Cnt-1));
            for (j = 0; j < Spc; j++)
                fputc(' ', Out);
        }
    PrintWord(Out); /* Jeste vytiskneme posledni slovo */
    fputc('\n', Out);
}

/* Vypiseme nejlepsi vysledek */
void PrintBestText(void)
{
    FILE *Out;
    int Lines = 0, ActWord;
    /* Pocet radek vysledneho textu; Aktualni slovo */
    int LB[MAXLINES], i; /* Pozice jednotlivych zalomeni */

    /* Zjistime pozice jednotlivych zalomeni */
    for (ActWord = WCnt; ActWord > 0; Lines++, ActWord = WrapPos[ActWord])
        LB[Lines] = ActWord;
    LB[Lines] = 0;
    if (!(Out = fopen(OUTPUT, "w")))

```

```

{
    puts("Can't open output file.");
    exit(1);
}
/* Nechame vytisknout radku */
for (i = Lines - 1; i > 0; i--)
    PrintLine(Out, LB[i+1], LB[i] - LB[i+1]);

/* Ted jeste vytiskneme posledni radku */
for (i = LB[1]; i < LB[0] - 1; i++)
{
    PrintWord(Out);
    fputc(' ', Out);
}
PrintWord(Out);
fputc('\n', Out);

fclose(Out);
}

int main(void)
{
    ProcessInput(); /* Nacte vstup a rozdeli ho na slova */
    FindBestSep(); /* Malezneme nejlepsi rozdeleni na radky */
    PrintBestText(); /* Vypise text podle spoctenych zalomeni */
    return 0;
}

```

P – III – 5

Nejprve si rozmysleme, že platí následující tvrzení: Trasy linek městem lze navrhnout právě tehdy, pokud ze všech křižovatek vychází sudý počet ulic. Kdybychom si trasy jednotlivých linek vyznačili různými barvami v plánu města, potom by každá z nich tvořila cyklus. Každá ulice by měla jednoznačně určenu svou barvu. Pokud si trasy linek zakreslíme do mapy, bude z každé křižovatky vycházet buď žádná nebo právě dvě ulice od jedné určité barvy — žádná, pokud trasa příslušné barvy křižovatkou neprochází, a dvě, pokud ano; více ulic stejné barvy nemůže z křižovatky vycházet, neboť každá linka projíždí křižovatkou nejvýše jedenkrát. Obarvení ulic v mapě tedy „páruje“ ulice vycházející z křižovatky a tudíž počet ulic vycházejících z jedné křižovatky musí být sudý.

Nyní si naopak rozmysleme, že pokud z každé křižovatky vychází sudý počet ulic, potom lze navrhnout trasy linek tak, aby splňovaly požadavky zadání úlohy. Postupně obarvujeme ulice ve městě tak, aby ulice stejné barvy tvořily cyklus (tj. odpovídaly nějaké autobusové lince). Uli-

ce, které jsme již obarvili, nebudeme nadále považovat za součást města; tím zmenšíme počet ulic vycházejících z jedné křižovatky o sudé číslo (o nulu nebo o dvě), takže počet ulic vycházejících z každé křižovatky bude stále sudý. Vyberme si nějakou křižovatku ve městě a označme si ji na mapě (položme do ní kamínek); vydejme se z této křižovatky po libovolné (dosud neobarvené) cestě a položme do křižovatky, do které jsme dorazili, kamínek. Z každé křižovatky lze vždy pokračovat alespoň jednou ulicí — do křižovatky jsme po jedné ulici přišli, a protože počet neobarvených ulic, které z ní vedou, je sudý, musí z ní vést tedy alespoň dvě neobarvené ulice — tou druhou můžeme pokračovat. Skončíme, pokud bychom na nějakou křižovatku měli položit druhý kamínek — tehdy jsme našli cyklus z ulic a tento cyklus obarvíme nějakou dosud nepoužitou barvou (a prohlásíme ho za novou autobusovou linku). Kamínky odstraníme z mapy a celý proces opakujeme tak dlouho, dokud mapa města obsahuje nějaké neobarvené ulice.

Předchozí důkaz nám dává návod k vytvoření algoritmu, který řeší zadanou úlohu. Nejprve ověříme, zda z některé křižovatky vychází lichý počet ulic; je-li tomu tak, potom rovnou vypíšeme „Nelze“. V opačném případě začneme aplikovat postup z minulého odstavce; kamínky samozřejmě nahradíme nastavováním vhodného příznaku v programu. Pokud nalezneme cyklus, příslušné ulice z mapy rovnou vymažeme a cyklus vypíšeme na výstup. Abychom ušetřili čas, ponecháme v mapě „kamínky“ na křižovatkách, které jsou mezi výchozí křižovatkou a křižovatkou, kam jsme měli položit dva kamínky; na tyto křižovatky bychom mohli položit kamínky i ve městě, ve kterém by byl vynechán právě nalezený cyklus. Pokládání kamínků budeme realizovat jednoduchou rekurzivní funkcí. Zbývá vyřešit, jak právě obarvené ulice rychle odstraňovat z mapy města uložené v paměti počítače. Ulice spojující dvě stejné křižovatky si budeme pamatovat jako jednu ulici s uvedením počtu ulic, které vedou paralelně s touto ulicí (tj. spojují dvě stejné křižovatky). Ulice si uložíme do dvojrozměrného pole; jeden jeho index bude představovat číslo křižovatky a jeho druhý index bude představovat pořadové číslo ulice vycházející z dané křižovatky. Rozměry tohoto pole tedy budou počet křižovatek $\times$ maximální počet ulic vycházejících z jedné křižovatky. U každé křižovatky jsou uvedeny všechny ulice, které z ní vycházejí. U každé ulice si pamatujeme, kam vede, kolik ulic s ní vede paralelně a index do pole určující, kde jsou informace o této ulici rovněž uloženy u křižovatky na jejím opačném konci. Příslušnou ulici vymažeme jednoduše tak, že upravíme informace o ní u obou křižovatek, které spojuje, což

lze provést v konstantním čase, neboť si pamatujeme její index u druhé křižovatky.

Paměťové i časové nároky našeho algoritmu jsou $O(N + M)$, kde N je počet křižovatek ve městě a M je počet ulic ve městě. Paměťové nároky algoritmu lze reprezentací pouze jedné z paralelních hran (viz minulý odstavec) snížit na $O(N + H)$, kde H je maximální počet ulic, z nichž žádné dvě nejsou paralelní.

Přípravná soustředění před 42. MMO

V průběhu 50. ročníku se konalo výběrové soustředění pro přípravu na mezinárodní matematickou olympiádu bezprostředně po skončeném celostátním kole kategorie A, a to od 9. do 13. dubna 2001 v Jevíčku. Na základě výsledků II. a III. kola bylo na ně pozváno 10 kandidátů na reprezentaci.

Soustředění bylo zaměřeno na řešení obtížných úloh v omezeném čase (v soutěžních podmínkách). Po odpolední relaxaci byl proveden detailní rozbor opravených řešení. Úspěšnost jednotlivých studentů ukazuje následující tabulka:

Jan Herman	4/4, G Brno, tř. Kpt. Jaroše 14	73
Martin Tancer	3/4, G Ch. Dopplera, Praha 5	72,5
Josef Křišťan	7/7, G Plzeň, Mikulášské nám.	64,5
Tomáš Protivínský	3/4, G Brno, tř. Kpt. Jaroše 14	70
Ondřej Suchý	7/7, G Plzeň, Mikulášské nám.	69
Martin Káldy	2/4, G Ch. Dopplera, Praha 5	59,5
Marek Sulovský	4/4, G Brno, tř. Kpt. Jaroše 14	59,5
Jan Kynčl	6/6, G Jilemnice	70,5
Pavel Čížek	2/4, G a OA Kralupy n. Vlt.	47
Jaroslav Hájek	3/4, GMK, Bílovec	68

Na základě uvedených výsledků, v nichž jsou započítány i výsledky oblastního a celostátního kola, bylo vybráno šest reprezentantů a jeden náhradník. Ti pak byli ještě pozváni na přípravné soustředění, které se konalo opět v Jevíčku v týdnu od 29. května do 1. června a bylo věnováno nejen řešení úloh, ale i rozšíření znalostí našich olympioniků.

Jednotlivé semináře vedli a úlohy připravili:

dr. Karel Horák (9. 4.),

dr. Jaroslav Švrček (10. 4.).

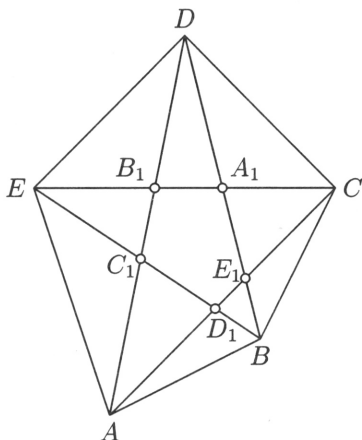
dr. Martin Panák a dr. Pavel Calábek (11. 4.)

a doc. Jaromír Šimša (12. a 13. 4.).

Úlohy zadané na přípravném soustředění

1. V rovině s kartézskou soustavou souřadnic je dán konvexní pětiúhelník $ABCDE$, jehož vrcholy mají celočíselné souřadnice. Dokažte, že uvnitř nebo na hranici pětiúhelníku $A_1B_1C_1D_1E_1$ (obr. 28) leží aspoň jeden bod s celočíselnými souřadnicemi.

2. Je dán čtyřúhelník $ABCD$ s vepsanou kružnicí ω . Jeho strany AB a CD leží na různoběžkách, které se protínají v bodě O . Označme ω_1 kružnici, jež se dotýká strany BC v bodě K a prodloužení stran AB a CD , a ω_2 kružnici, jež se dotýká strany AD v bodě L a prodloužení stran AB a CD . Jestliže body O, K, L leží v přímce, potom leží v přímce i střed kružnice ω a středy stran BC a AD .



Obr. 28

3. V rovině je dán konvexní n -úhelník $P_1P_2 \dots P_n$. Jestliže pro jeho dva libovolné vrcholy P_i, P_j existuje vrchol daného mnohoúhelníku, z něhož je úsečka P_iP_j vidět pod úhlem 60° , potom $n = 3$. Dokažte.

4. Nechť I značí střed kružnice vepsané trojúhelníku ABC , středy stran AB, AC označme po řadě C_1, B_1 . Nechť přímky IC_1 a AC se protínají v bodě B_2 a přímky IB_1 a AB se protínají v bodě C_2 . Stanovte velikost úhlu CAB , jsou-li rovny obsahy trojúhelníků ABC a AB_2C_2 .

5. Sestrojte čtyřúhelník $ABCD$, jsou-li dány délky a, b, c, d jeho stran a délka m spojnice středů stran AB a CD .

6. Nechť D, E, F jsou po řadě vnitřní body stran BC, CA, AB trojúhelníku ABC a r nechť značí poloměr kružnice jemu opsané. Dokažte, že platí nerovnost

$$\left(\frac{1}{|AD|} + \frac{1}{|BE|} + \frac{1}{|CF|} \right) (|DE| + |EF| + |FD|) \geq \frac{|AB| + |BC| + |CA|}{r}.$$

7. Kružnice vepsaná čtyřúhelníku $ABCD$ se dotýká jeho stran DA, AB, BC, CD po řadě v bodech K, L, M, N . Nechť k_1, k_2, k_3, k_4 jsou kružnice vepsané po řadě trojúhelníkům AKL, BLM, CMN, DNK . Uvažujme společné vnější tečny ke dvojicím kružnic $(k_1, k_2), (k_2, k_3), (k_3, k_4)$,

(k_4, k_1) různé od stran čtyřúhelníku $ABCD$. Dokažte, že úhlopříčky čtyřúhelníku, jehož strany leží na těchto společných vnějších tečnách, jsou navzájem kolmé.

8. Existuje rozklad množiny celých čísel na tři disjunktní podmnožiny takové, že pro libovolné celé číslo n leží čísla n , $n - 50$ a $n + 187$ vždy v různých podmnožinách?

9. Buď $\{a_n\}_{n=1}^{\infty}$ nekonečná posloupnost přirozených čísel taková, že v rozkladu libovolného jejího členu na prvočísla se objevuje nejvýše 2001 čísel (každé prvočíslu počítáme tolikrát, v jaké mocnině dělí daný člen). Dokažte, že z posloupnosti $\{a_n\}_{n=1}^{\infty}$ lze vybrat podposloupnost $\{b_n\}_{n=1}^{\infty}$ takovou, že největší společný dělitel libovolných dvou jejích členů (b_i, b_j) je vždy totéž číslo.

10. Nechť $a, a + d, a + 2d, \dots$ je nekonečná aritmetická posloupnost. Dokažte, že obsahuje-li nějaké druhé mocniny přirozených čísel, tak alespoň jedna z nich je menší než $a + 2d\sqrt{a} + d^2$.

11. Nechť $A = \{1, 2, \dots, m + n\}$, kde m, n jsou přirozená čísla, a nechť funkce $f: A \rightarrow A$ je definována následujícím způsobem:

$$\begin{aligned} f(i) &= i + 1 && \text{pro } i = 1, 2, \dots, m - 1, m + 1, \dots, m + n - 1, \\ f(m) &= 1 && \text{a} && f(m + n) = m + 1. \end{aligned}$$

a) Dokažte: Nechť m a n jsou lichá čísla, pak existuje funkce $g: A \rightarrow A$ taková, že pro všechna $a \in A$ platí $g(g(a)) = f(a)$.

b) Dokažte: Nechť m je sudé číslo, pak $m = n$, právě když existuje funkce $g: A \rightarrow A$ taková, že pro všechna $a \in A$ platí $g(g(a)) = f(a)$.

12. Ukažte, že pro libovolné celé číslo n existuje právě jeden polynom Q s koeficienty z množiny $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ takový, že $Q(-2) = Q(-5) = n$.

13. Určete všechny funkce $f: \mathbb{R} \rightarrow \mathbb{R}$ takové, že $f(x + yf(x)) = f(x) + xf(y)$ pro všechna reálná čísla x a y .

14. Dokažte, že pro libovolná čísla a, b, c z intervalu $\langle 0, 1 \rangle$ platí nerovnost

$$2(a^3 + b^3 + c^3) \leq 3 + a^2b + b^2c + c^2a.$$

15. Hráči A a B hrají pro předem zvolené přirozené číslo n takovou hru. Začíná hráč A , který napíše libovolnou číslici z množiny

$M = \{1, 2, 3, 4, 5, 6\}$. K ní hráč B připiše opět číslici z M , k ní zase připiše hráč A číslici z M atd., až vznikne $2n$ -místné číslo. Je-li toto číslo násobkem devíti, vyhrává hráč B ; v opačném případě vyhrává hráč A . Který z hráčů má (v závislosti na čísle n) zaručenu výhru?

16. Najděte všechny dvojice celých čísel x a y , pro které platí

$$x^3 + x^2y + xy^2 + y^3 = 8(x^2 + xy + y^2 + 1).$$

17. Dokažte, že existuje rostoucí nekonečná aritmetická posloupnost sestavená pouze z těch přirozených čísel, která nelze pro žádné $k < 10$ zapsat jako součet k druhých mocnin lichých celých čísel.

18. Množinu všech složených lichých přirozených čísel menších než 100 запиšte jako sjednocení tří (nikoliv nutně disjunktních) aritmetických posloupností.

19. Doplněte celá čísla a_i, b_i v rovnici

$$\frac{1}{n+1} \binom{2n}{n} = \binom{a_1n + b_1}{a_2n + b_2} - \binom{a_3n + b_3}{a_4n + b_4}$$

tak, aby vzniklá rovnost platila pro každé přirozené číslo n .

20. Napište aspoň jednoho dělitele d čísla $N = 4^{545} + 545^4$ splňujícího odhady $2^{545} < d < N$.

21. Z písmen A, B je sestaveno libovolné slovo délky n , ve kterém je právě m slabik BA ($2m \leq n$). Počet všech těchto slov запиšte kombinačním číslem $\binom{N}{K}$, kde N a K jsou mnohočleny proměnných n a m .

Mezinárodní střetnutí česko-slovensko-polské

BÍLOVEC, 14.–15. ČERVNA 2001

V rámci přípravy na 42. MMO se uskutečnilo již sedmé tradiční střetnutí českého a slovenského reprezentačního výběru. Tentokrát se poprvé soutěže zúčastnilo i polské reprezentační družstvo.

Soutěž se uskutečnila v termínu 14.–16. 6. 2001 na půdě Gymnázia Mikuláše Koperníka v Bílovci. Všechna zúčastněná družstva dorazila do místa konání soutěže již ve středu 13. června. Organizace a průběh soutěže zůstal zachován z předešlých ročníků. Soutěžící tedy (podobně jako na MMO) řešili ve dvou soutěžních dnech (14. a 15. června) po třech úlohách. Za každou z nich mohli získat nejvýše 7 bodů, celkově tedy maximálně 42 body. Na jejich řešení měli žáci každý soutěžní den rezervováno (stejně jako na MMO) 4,5 hodiny čistého času.

Pořadí	Jméno	Země	body	Součet
1.–2.	Michał Adamaszek	POL	777773	38
	Karol Cwalina	POL	775577	38
3.	Katarína Quittnerová	SVK	772777	37
4.–6.	Radovan Bauer	SVK	777724	34
	<i>Jan Kynčl</i>	CZE	727774	34
7.–9.	Jarosław Wrona	POL	776563	34
	<i>Jaroslav Hájek</i>	CZE	772771	31
	Roman Łomowski	POL	770773	31
	Paweł Walter	POL	656563	31
10.–11.	Jana Szolgayová	SVK	777620	29
	<i>Martin Tancer</i>	CZE	747713	29
12.	Tomáš Kulich	SVK	347760	27
13.–14.	Peter Bella	SVK	375144	24
	Aleksander ZabŁocki	POL	732660	24
15.	<i>Jan Herman</i>	CZE	040773	21
16.	Robert Lukotka	SVK	011773	19
17.	<i>Tomáš Protivínský</i>	CZE	122623	16
18.	<i>Ondřej Suchý</i>	CZE	002711	11

Návrh všech šesti úloh (a jejich vzorová řešení) připravili členové úlohové komise z České republiky — dr. Jaroslav Švrček a doc. Jaromír Šimša. Úlohy koordinovala mezinárodní komise ve složení Jaromír Šimša, Jaroslav Švrček, Pavel Calábek za Českou republiku, Eugen Kováč a Juraj Földes za Slovensko a Marcin Kuczma, Józef Kalinowski a Rafał Łochowski za Polsko. Jak je vidět z připojené tabulky, nejlépe si v 7. ročníku soutěže vedl polský reprezentační výběr. Celkové výsledky také naznačily, že od našich olympioniků nelze za necelý měsíc na 42. MMO bohužel očekávat výraznější úspěch. Tato prognóza se skutečně potvrdila.

V příštím roce se uskuteční 8. mezinárodní střetnutí MO (CZE – POL – SVK) na základě pozvání polského družstva počátkem června v horském příhraničním středisku Zwardoń.

Texty soutěžních úloh

1. Dokažte, že pro libovolná kladná čísla $a_1, a_2, \dots, a_n$ ($n \geq 2$) platí nerovnost

$$(a_1^3 + 1)(a_2^3 + 1) \dots (a_n^3 + 1) \geq (a_1^2 a_2 + 1)(a_2^2 a_3 + 1) \dots (a_n^2 a_1 + 1).$$

2. Trojúhelník ABC má ostré vnitřní úhly při vrcholech A a B . Nad stranami AC a BC jsou trojúhelníku vně připsány rovnoramenné trojúhelníky ACD a BCE se základnami AC a BC tak, že $|\sphericalangle ADC| = |\sphericalangle ABC|$ a současně $|\sphericalangle BEC| = |\sphericalangle BAC|$. Označme S střed kružnice opsané trojúhelníku ABC . Dokažte, že délka lomené čáry DSE je rovna obvodu trojúhelníku ABC , právě když je úhel ACB pravý.

3. Pro libovolná přirozená čísla n, k splňující podmínku $\frac{1}{2}n < k \leq \frac{2}{3}n$ najděte nejmenší počet polí, která lze obsadit na čtvercové šachovnici $n \times n$ tak, aby v žádném řádku ani v žádném sloupci šachovnice neexistovalo k volných (tj. neobsazených) sousedních polí.

4. V rovině jsou dány body A, B ($A \neq B$). V této rovině uvažujme libovolný trojúhelník ABC s vlastností: Uvnitř jeho stran BC, CA existují po řadě body D, E , pro něž platí

$$(i) \quad \frac{|BD|}{|BC|} = \frac{|CE|}{|CA|} = \frac{1}{3};$$

(ii) body A, B, D, E leží (v tomto pořadí) na téže kružnici.

Určete množinu průsečíků přímk AD a BE pro všechny trojúhelníky ABC s danou vlastností.

5. Určete všechny funkce $f: \mathbb{R} \rightarrow \mathbb{R}$ vyhovující rovnici

$$f(x^2 + y) + f(f(x) - y) = 2f(f(x)) + 2y^2$$

pro všechna $x, y \in \mathbb{R}$.

6. V prostoru je dána kartézská soustava souřadnic. Každý bod s celočíselnými souřadnicemi nazveme *mřížovým*. Obarvěme 2000 mřížových bodů modře a jiných 2000 mřížových bodů červeně tak, aby žádné dvě modročervené úsečky neměly společný vnitřní bod. (Úsečku nazýváme modročervenou, pokud je jeden její krajní bod obarven modře a druhý červeně.) Uvažujme nejmenší kvádr s hranami rovnoběžnými s osami souřadnic, který obsahuje všechny obarvené body.

a) Dokažte, že kvádr obsahuje alespoň 500 000 mřížových bodů.

b) Udejte příklad popsaneho obarvení, kdy uvažovaný kvádr obsahuje nejvýše 8 000 000 mřížových bodů.

Řešení úloh

1. Tvrzení dokážeme užitím principu matematické indukce vzhledem k přirozenému číslu n .

(i) Pro $n = 2$ dostáváme postupně

$$\begin{aligned} (a_1^3 + 1)(a_2^3 + 1) &\geq (a_1^2 a_2 + 1)(a_2^2 a_1 + 1), \\ a_1^3 a_2^3 + a_1^3 + a_2^3 + 1 &\geq a_1^3 a_2^3 + a_1^2 a_2 + a_1 a_2^2 + 1, \\ a_1^2(a_1 - a_2) + a_2^2(a_2 - a_1) &\geq 0, \\ (a_1^2 - a_2^2)(a_1 - a_2) &\geq 0, \\ (a_1 - a_2)^2(a_1 + a_2) &\geq 0. \end{aligned}$$

Poslední nerovnost zřejmě platí pro libovolná kladná čísla a_1, a_2 . Protože všechny použité úpravy byly ekvivalentní, platí i daná nerovnost pro $n = 2$.

(ii) Předpokládejme nyní, že nerovnost z textu úlohy platí pro určité $n \geq 2$. Dokážeme platnost nerovnosti i pro $n + 1$. Nerovnost

$$\begin{aligned} (a_1^3 + 1) \dots (a_n^3 + 1)(a_{n+1}^3 + 1) &\geq \\ &\geq (a_1^2 a_2 + 1) \dots (a_n^2 a_{n+1} + 1)(a_{n+1}^2 a_1 + 1) \end{aligned}$$

bude (za uvedeného předpokladu) platit, bude-li splněna podmínka

$$a_{n+1}^3 + 1 \geq \frac{(a_n^2 a_{n+1} + 1)(a_{n+1}^2 a_1 + 1)}{a_n^2 a_1 + 1}.$$

Pomocí ekvivalentních úprav zjistíme, kdy je uvedená podmínka splněna. Postupně tak obdržíme

$$\begin{aligned} (a_{n+1}^3 + 1)(a_n^2 a_1 + 1) &\geq (a_n^2 a_{n+1} + 1)(a_{n+1}^2 a_1 + 1), \\ a_{n+1}^3 a_n^2 a_1 + a_n^2 a_1 + a_{n+1}^3 + 1 &\geq a_{n+1}^3 a_n^2 a_1 + a_{n+1} a_n^2 + a_{n+1}^2 a_1 + 1, \\ a_n^2(a_1 - a_{n+1}) + a_{n+1}^2(a_{n+1} - a_1) &\geq 0, \\ (a_{n+1} - a_1)(a_{n+1} - a_n)(a_{n+1} + a_n) &\geq 0. \end{aligned}$$

Poslední nerovnost obecně neplatí. Zadaná nerovnost je však cyklická, tj. nezmění se, jestliže n -tici kladných čísel $(a_1, a_2, \dots, a_n)$ zaměníme libovolnou z n -tic

$$(a_2, a_3, \dots, a_n, a_1), (a_3, a_4, \dots, a_n, a_1, a_2), \dots, (a_n, a_1, a_2, \dots, a_{n-1}).$$

Lze tudíž danou $(n+1)$ -tici $(a_1, a_2, \dots, a_{n+1})$ předem cyklicky pozměnit tak, aby číslo a_{n+1} bylo maximální ze všech $n+1$ čísel a_i . Potom jsou obě čísla $a_{n+1} - a_1$ a $a_{n+1} - a_n$ nezáporná, a proto platí i poslední nerovnost.

Tím je důkaz nerovnosti matematickou indukcí ukončen.

2. Uvažujme obvyklé označení délek stran a velikostí vnitřních úhlů v trojúhelníku ABC , r nechť značí poloměr jeho opsané kružnice. Dále nechť G je střed strany AC a D' bod souměrně sružený s bodem D podle osy AC . Protože $|\angle AD'C| = |\angle ABC|$, leží bod D' na kružnici opsané trojúhelníku ABC , tj. $|SD'| = r$, a tudíž $|GD| = |GD'| = |SG| + r$.

Odtud dostaneme $|SD| = |SG| + |GD| = 2|SG| + r = r(2 \cos \beta + 1)$. Analogicky $|SE| = r(2 \cos \alpha + 1)$. Délka lomené čáry DSE je tedy

$$|SD| + |SE| = 2r(\cos \alpha + \cos \beta + 1),$$

zatímco obvod trojúhelníku ABC je $2r(\sin \alpha + \sin \beta + \sin \gamma)$.

Zjistíme dále, za jakých podmínek platí rovnost

$$\cos \alpha + \cos \beta + 1 = \sin \alpha + \sin \beta + \sin \gamma.$$

Tu postupně upravíme následujícím způsobem:

$$\begin{aligned} 2 \cos \frac{\alpha + \beta}{2} \cos \frac{\alpha - \beta}{2} + 1 &= 2 \sin \frac{\alpha + \beta}{2} \cos \frac{\alpha - \beta}{2} + \sin \gamma, \\ \left(\cos \frac{\gamma}{2} - \sin \frac{\gamma}{2} \right)^2 &= 2 \cos \frac{\alpha - \beta}{2} \left(\cos \frac{\gamma}{2} - \sin \frac{\gamma}{2} \right), \end{aligned}$$

tj.

$$\left(\cos \frac{\gamma}{2} - \sin \frac{\gamma}{2}\right) \left(\cos \frac{\gamma}{2} - \sin \frac{\gamma}{2} - 2 \cos \frac{\alpha - \beta}{2}\right) = 0.$$

Je-li $\gamma = 90^\circ$, je poslední rovnost splněna. Je-li však $\gamma \neq 90^\circ$, lze rovnost upravit ekvivalentně na tvar $\cos \frac{1}{2}\gamma - \sin \frac{1}{2}\gamma = 2 \cos \frac{1}{2}(\alpha - \beta)$, což je však ve sporu s odhady

$$\cos \frac{\gamma}{2} - \sin \frac{\gamma}{2} = \sqrt{2} \cos \left(\frac{\gamma}{2} + \frac{\pi}{4}\right) < \sqrt{2} = 2 \cos \frac{\pi}{4} < 2 \cos \frac{\alpha - \beta}{2}$$

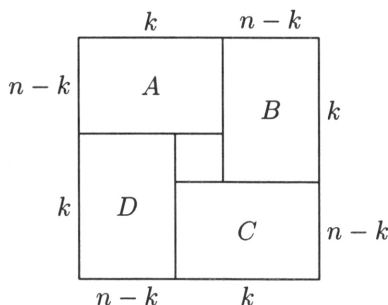
(pro ostré úhly α, β totiž zřejmě platí $|\frac{1}{2}(\alpha - \beta)| < \frac{1}{4}\pi$).

Tím je důkaz ukončen.

3. Ukážeme, že hledaný nejmenší počet polí je roven číslu $4(n - k)$. Čísla n, k splňující podmínku z textu zadání budou v celém textu pevná; každé obsazení několika polí šachovnice $n \times n$ s požadovanou vlastností nazveme „dobrým“. (Dobré je například obsazení všech n^2 polí.)

Popíšme nejdříve dobré obsazení $4(n - k)$ polí. Jednotlivé řádky šachovnice označme postupně číslы $i = 0, 1, \dots, n - 1$, stejně tak sloupce číslы $j = 0, 1, \dots, n - 1$ a obsaďme právě ta pole, jejichž souřadnice (i, j) splňují $i + j \equiv k - 1 \pmod{k}$. Toto obsazení je zřejmě dobré a je tvořeno jednak k polí, pro která $i + j = k - 1$, jednak $2n - 2k$ polí, pro která $i + j = 2k - 1$, a konečně $2n - 3k$ polí, pro která $i + j = 3k - 1$ (v případě $k = \frac{2}{3}n$ pole třetího druhu chybí, tehdy ovšem $2n - 3k = 0$), což je celkem $k + (2n - 2k) + (2n - 3k) = 4(n - k)$ polí.

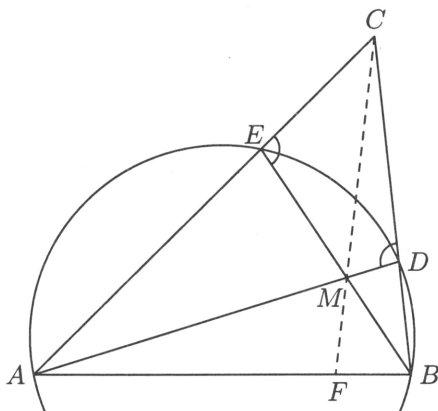
V druhé části řešení ukážeme, že každé dobré obsazení je tvořeno alespoň $4(n - k)$ polí. K tomuto účelu vyčleníme na dané šachovnici $n \times n$ čtyři nepřekrývající se obdélníky A, B, C, D o rozměrech $k \times (n - k)$, jak vidíme na obr. 29.



Obr. 29

Při libovolném dobrém obsazení celé šachovnice musí být obsazeno aspoň jedno pole v každém z $(n - k)$ řádků polí obdélníku A , takže v obdélníku A je celkem obsazeno alespoň $(n - k)$ polí. Obdobnou úvahou o sloupcích v B , řádcích v C a sloupcích v D zjistíme, že rovněž v každém z obdélníků B , C , D je obsazeno alespoň $(n - k)$ polí. Proto je na celé šachovnici obsazeno alespoň $4(n - k)$ polí, což jsme chtěli dokázat.

4. Délky stran uvažovaného trojúhelníku ABC označme obvyklým způsobem a , b , c .



Obr. 30

Vzhledem k tomu, že body A , B , D , E leží na téže kružnici, platí (obr. 30) $|\sphericalangle ADC| = |\sphericalangle BEC|$. Trojúhelníky ADC a BEC jsou tudíž podobné, z čehož vyplývá

$$\frac{|DC|}{|CA|} = \frac{|CE|}{|CB|}, \quad \text{tj.} \quad \frac{\frac{2}{3}a}{b} = \frac{\frac{1}{3}b}{a}.$$

Snadnou úpravou odtud dostáváme $b : a = \sqrt{2} : 1$. Vrchol C trojúhelníku ABC dané vlastnosti leží tedy na Apolloniově kružnici k (se středem na přímce AB).

Označme dále M průsečík přímek AD a BE , F průsečík přímky CM se stranou AB a $|AF| = x$. Z Cèvyovy věty vyplývá

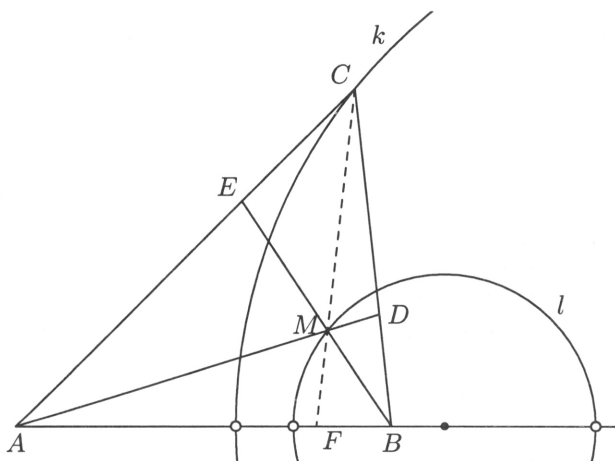
$$\frac{|AF|}{|FB|} \cdot \frac{|BD|}{|DC|} \cdot \frac{|CE|}{|EA|} = \frac{x}{c-x} \cdot \frac{1}{2} \cdot \frac{1}{2} = 1.$$

Z poslední rovnosti již bezprostředně plyne $x = \frac{4}{5}c$, tj. přímka CM protíná stranu AB v pevném vnitřním bodě F , pro který platí $|AF| : |FB| = 4 : 1$.

Užitím Van Aubelovy věty pro *ceviány* AD , BE a CF , které se protínají v bodě M , dostáváme dále

$$\frac{|CM|}{|MF|} = \frac{|CD|}{|DB|} + \frac{|CE|}{|EA|} = \frac{2}{1} + \frac{1}{2} = \frac{5}{2}, \quad \text{a tedy} \quad |FM| : |FC| = 2 : 7.$$

Uvažujme nyní stejnoolehlost $h(F, \frac{2}{7})$. Ta zobrazuje vrchol C trojúhelníku ABC dané vlastnosti na průsečík M přímek AD a BE a Apolloniovu kružnici k na kružnici l , jejíž střed leží rovněž na přímce AB (obr. 31). Hledaná množina průsečíků M přímek AD a BE leží tedy na kružnici l .



Obr. 31

Ukažme naopak, že ke každému bodu M kružnice l (z níž jsou vyňaty dva body ležící na přímce AB) existuje bod $C \in k$, který je vrcholem trojúhelníku ABC s danou vlastností. Ukažme tedy, že uvnitř jeho stran BC , CA existují po řadě body D , E vyhovující podmínkám (i), (ii). Ke každému bodu M kružnice l existuje bod $C \in k$ tak, že platí $C = h^{-1}(M)$. Uvnitř stran BC , CA trojúhelníku ABC existují po řadě body D , E vyhovující podmínce (i). Ukážeme, že tyto dva body vyhovují také podmínce (ii). Stačí ověřit, že platí $\cos |\sphericalangle ADC| = \cos |\sphericalangle BEC|$. To lze např. využitím kosinové věty pro trojúhelníky ADC a BEC . Zde

využijeme vztah $b : a = \sqrt{2} : 1$. Při výpočtu délek úseček AD , resp. BE opět využijeme kosinovou větu v trojúhelníku ABC (a to pro vyjádření $\cos |\sphericalangle BCA|$).

Závěr: Hledanou množinou M průsečíků přímek AD a BE všech trojúhelníků s vlastností danou v textu úlohy je tedy kružnice l , která je obrazem kružnice k ve stejnoolehlosti $h(F, \frac{2}{7})$ a z níž jsou vyňaty dva body (jejího průměru) ležící na přímce AB .

5. Ukážeme, že jediné řešení je funkce $f(x) = x^2$. Položme nejprve $y = f(x)$ a dále $y = -x^2$. Porovnáním výsledků, které dostaneme těmito speciálními volbami, dostaneme $|f(x)| = x^2$ a také $f(0) = 0$. Volbou $x = 0$ dále obdržíme $f(y) + f(-y) = 2y^2$. Jak už ale víme, je nutně $f(y) \leq y^2$ a současně $f(-y) \leq y^2$. Obě předchozí nerovnosti však přecházejí s ohledem na podmínku $f(y) + f(-y) = 2y^2$ v rovnosti, a to pro každé $y \in \mathbb{R}$.

Zkouškou se přesvědčíme, že řešením dané funkcionální rovnice je funkce $f(x) = x^2$ ($x \in \mathbb{R}$).

6. a) Nejprve nahlédneme, že žádné dvě modročervené úsečky nesmějí mít stejný (orientovaný) směr. V opačném případě by jejich koncové body tvořily vrcholy lichoběžníku (nebo rovnoběžníku) s modročervenými úhlopříčkami majícími společný vnitřní bod (platí to i v případě, kdy čtyřúhelník degeneruje v úsečku).

Dále si uvědomme, že všechny dvojice mřížových bodů v uvažovaném kvádru o rozměrech a, b, c určují nejvýše $8(a+1)(b+1)(c+1)$ směrů.

Je-li totiž uvažovaný kvádr v kartézské souřadné soustavě $Oxyz$ popsán nerovnostmi

$$x_0 \leq x \leq x_0 + a, \quad y_0 \leq y \leq y_0 + b, \quad z_0 \leq z \leq z_0 + c,$$

je každý zmíněný směr určen vektorem (u, v, w) , kde u, v, w jsou celá čísla splňující nerovnosti $|u| \leq a, |v| \leq b, |w| \leq c$. Všech takových vektorů je právě $(2a+1)(2b+1)(2c+1)$, což je méně než $8(a+1)(b+1)(c+1)$.

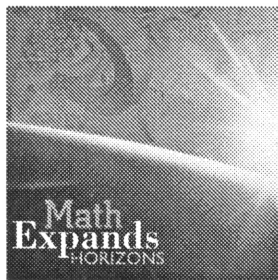
Daných 2 000 modrých a 2 000 červených bodů určuje 4 000 000 modročervených úseček, z nichž žádné dvě nemohou mít stejný směr. Je tedy $4\,000\,000 \leq 8(a+1)(b+1)(c+1)$, neboli $(a+1)(b+1)(c+1) \geq 500\,000$. Tím jsme hotovi s důkazem části a), neboť $(a+1)(b+1)(c+1)$ je počet mřížových bodů tohoto kvádru.

b) Vezměme kvádr o rozměrech $1\,999 \times 1\,999 \times 1$ s vrcholy v bodech $[0, 0, 0]$, $[0, 0, 1]$, $[1\,999, 0, 0]$, $[0, 1\,999, 0]$, $[1\,999, 0, 1]$, $[0, 1\,999, 1]$,

$[1\,999, 1\,999, 0]$, $[1\,999, 1\,999, 1]$. Modře nyní obarvíme všech 2 000 mřížových bodů na hraně o vrcholech $[0, 0, 0]$ a $[1\,999, 0, 0]$, červeně pak všech 2 000 mřížových bodů na hraně o vrcholech $[0, 0, 1]$ a $[0, 1\,999, 1]$. Protože tyto hrany (na nichž leží modré, resp. červené body) jsou mimoběžné, žádné dvě různé modročervené úsečky nemají společný vnitřní bod. Kvádr přitom obsahuje právě 8 000 000 mřížových bodů.

42. mezinárodní matematická olympiáda

Tato každoroční soutěž středoškoláků v řešení matematických úloh se prvních 21 let své existence konala vždy v některé evropské zemi. Teprve v roce 1981 opustila starý kontinent a zamířila do Washingtonu, hlavního města USA, kde její 22. ročník uspořádala vědecká společnost The Mathematical Association of America. Po dvaceti letech se soutěž vrátila téměř na stejné místo, když se hlavním dějištěm 42. ročníku MMO v červenci 2001 stal areál Univerzity George Masona, vzdálený od centra Washingtonu asi



30 km (v roce 1981 to byla Univerzita Georgetown, ke které se dostanete po 20 minutách chůze od Bílého domu). Během dvou desetiletí se sice nezměnila ani náplň soutěže (ve dvou dnech řeší jednotliví účastníci vždy tři úlohy po dobu 4,5 hodin), ani vyhodnocení dosažených výsledků (podle bodového zisku za všech šest soutěžních úloh získává nejlepší dvanáctina soutěžících zlaté medaile, další šestina stříbrné a další třetina bronzové medaile), významně však vzrostl rozsah celé akce: zatímco v roce 1981 se soutěže zúčastnilo 185 žáků z 27 zemí, v roce 2001 už to bylo 473 žáků z 83 zemí. Šestici našich soutěžících v roce 2001 tvořili *Jaroslav Hájek* (GMK Bílovec), *Jan Herman* (G Brno, tř. Kpt. Jaroše), *Jan Kynčl* (G Jilemnice), *Tomáš Protivínský* (G Brno, tř. Kpt. Jaroše), *Ondřej Suchý* (G Plzeň, Mikulášské nám.) a *Martin Tancer* (G Ch. Dopplera Praha). Vedoucím delegace České republiky byl doc. RNDr. *Jaromír Šimša*, CSc. (MÚ AV ČR Brno), družstvo doprovázel RNDr. *Jaroslav Švrček*, CSc. (PřF UP Olomouc).

Mezinárodní porota 42. MMO, tvořená vedoucími delegací jednotlivých zemí, zahájila svou práci pod předsednictvím známého amerického matematika *R. Grahama* 2. července 2001, den před příletem soutěžících. Úkolem poroty bylo vybrat šestici soutěžních úloh a připravit jejich znění v jazycích zúčastněných zemí. Jak se mnozí vedoucí před vlastní

soutěží (konané 8. a 9. července) vyjádřili, vybraná šestice úloh byla neobvykle náročná. Přesto náš vedoucí věřil, že první a čtvrtá úloha budou českým soutěžícím vyhovovat a že většině z nich přinesou bodové zisky zaručující alespoň bronzové medaile. Nestalo se tak k našemu velkému zklamání, které nerozptýlil ani obdobný „výpadok“ slovenského družstva, jež v přípravě na soutěž vykazovalo lepší výsledky. Museli jsme se spokojit se ziskem dvou bronzových medailí *Jana Kynčla* a *Martina Tancera*. Kromě toho získali *Jaroslav Hájek* a *Tomáš Protivínský* čestné uznání, které se uděluje těm soutěžícím, kteří sice nedosáhli na žádnou z medailí, ale přitom za některou z úloh dostali plný počet sedmi bodů. Výsledky našich soutěžících najdete v přiložené tabulce.

Umístění		Body za úlohu						Body	Cena
		1	2	3	4	5	6		
243.–261.	Jaroslav Hájek, 3. roč. GMK Bílovec	7	0	0	3	0	0	10	
327.–346.	Jan Herman, 4. roč. gymnázia Brno, tř. Kpt. Jaroše	2	0	0	2	1	0	5	
148.–163.	Jan Kynčl, 6. roč. gymnázia Jilemnice	7	0	0	2	7	0	16	III.
243.–261.	Tomáš Protivínský, 3. roč. gymnázia Brno, tř. Kpt. Jaroše	0	7	0	1	2	0	10	
398.–428.	Ondřej Suchý, 4. roč. gymnázia Plzeň, Mikulášské nám.	0	0	0	0	2	0	2	
181.–198.	Martin Tancer, 3. roč. G. Ch. Dopplera Praha 5	3	0	7	2	2	0	14	III.
Celkem		19	7	7	10	14	0	57	

O náročnosti soutěžních úloh svědčí i nízké hranice pro zisk medailí: na bronzovou medaili stačilo 11 bodů, na stříbrnou 20 a na zlatou 30 z možného počtu 42 bodů. Žádný bod neztratili jen čtyři soutěžící: *Liang*

Xao s Zhiquiang Zangem z Číny a Reid Barton s Gabrielem Carrollem z USA. Jak je patrné z tabulky zúčastněných států, v neoficiálním hodnocení jednotlivých zemí podle celkového bodového zisku dopadly nejlépe (jak už je tradicí) Čína, Rusko a Spojené státy.

	I	II	III	body		I	II	III	body
ČLR	6	0	0	225	Makedonie	0	0	2	59
Rusko	5	1	0	196	Nový Zéland	0	1	1	58
USA	4	2	0	196	Česká republika	0	0	2	57
Bulharsko	3	3	0	185	Itálie	0	0	2	56
Korea	3	3	0	185	Mexiko	0	0	2	56
Kazachstán	4	1	0	168	Slovensko	0	0	2	54
Indie	2	2	2	148	Venezuela	0	1	1	53
Ukrajina	1	5	0	143	Norsko	0	1	1	48
Tchaj-wan	1	5	0	141	Bosna a Hercegovina	0	1	1	47
Vietnam	1	4	0	139	Maroko	0	0	1	45
Turecko	1	3	2	136	Arménie	0	0	2	44
Bělorusko	1	2	3	135	Nizozemsko	0	0	2	42
Japonsko	1	3	2	134	Rakousko	0	0	1	41
Německo	1	3	1	131	Litva	0	0	1	39
Rumunsko	1	2	2	129	Švýcarsko	0	0	2	38
Brazílie	0	4	2	120	Španělsko	0	0	1	37
Izrael	1	2	1	113	Indonézie	0	1	0	36
Írán	0	2	4	111	Malajsie	0	0	0	36
Hongkong	0	2	4	107	Trinidad a Tobago	0	0	2	36
Polsko	0	3	1	107	Tunisko	0	0	1	36
Maďarsko	0	2	3	104	Finsko	0	0	1	32
Argentina	0	3	2	103	Irsko	0	0	1	32
Thajsko	0	2	2	103	Macao	0	0	1	32
Kanada	1	0	4	100	Turkmenistán	0	0	0	29
Austrálie	1	0	4	97	Slovinsko	0	0	0	27
Kuba	1	1	3	92	Belgie	0	0	0	25
Uzbekistán	0	1	3	91	Dánsko	0	0	0	25
Francie	0	2	3	88	Švédsko	0	0	1	25
Singapur	0	1	4	87	Srí Lanka	0	0	1	21
Řecko	0	1	3	86	Albánie	0	0	0	20
Jugoslávie	0	1	3	79	Ázerbájdžán	0	0	1	19
Mongolsko	0	2	2	79	Island	0	0	0	18
Velká Británie	0	1	3	79	Filipíny	0	0	0	16
Kypř	0	0	4	78	Guatemala	0	0	0	12
Chorvatsko	0	1	2	76	Uruguay	0	0	0	8
JAR	0	1	3	75	Portugalsko	0	0	0	6
Estonsko	0	1	3	72	Kirgizie	0	0	0	5
Gruzie	0	1	3	71	Lucembursko	0	0	0	4
Lotyšsko	0	1	2	71	Kuvajt	0	0	0	3
Moldavsko	0	2	1	70	Paraguay	0	0	0	2
Peru	0	0	4	67	Ekvádor	0	0	0	0
Kolumbie	0	0	4	64					

Velkým překvapením je nevídaný úspěch Kazachstánu, jehož soutěžící se na olympiádu pečlivě připravovali během měsíčního internátního soustředění pod vedením pozvaného petrohradského matematika *S. Ruk-*

šina. Najdeme v budoucnu prostředky a časové možnosti pro obdobnou intenzivní přípravu i u nás?

Poděkování ovšem patří přerovskému podniku PRECHEZA, a.s., a společnosti AUTO NISSAN Kobliha v Přerově za jednotné oblečení celého českého družstva.

Texty soutěžních úloh

(v závorce je uvedena země, která úlohu navrhla)

1. Nechť O je střed kružnice opsané ostroúhlému trojúhelníku ABC . Bod P strany BC je patou výšky z vrcholu A . Předpokládejme, že $|\sphericalangle BCA| \geq |\sphericalangle ABC| + 30^\circ$. Dokažte, že $|\sphericalangle CAB| + |\sphericalangle COP| < 90^\circ$.
(Korea)

2. Dokažte, že nerovnost

$$\frac{a}{\sqrt{a^2 + 8bc}} + \frac{b}{\sqrt{b^2 + 8ca}} + \frac{c}{\sqrt{c^2 + 8ab}} \geq 1$$

platí pro všechna kladná reálná čísla a, b, c .
(Korea)

3. Matematické soutěže se zúčastnilo 21 dívek a 21 chlapců. Každý soutěžící vyřešil nejvýše šest úloh. Pro každou dívku a každého chlapce existuje alespoň jedna úloha, kterou vyřešili oba současně. Dokažte, že existuje úloha, kterou vyřešili alespoň tři dívky a alespoň tři chlapci.
(Německo)

4. Nechť n je liché číslo větší než 1 a nechť $k_1, k_2, \dots, k_n$ jsou daná celá čísla. Pro každou z $n!$ permutací $a = (a_1, a_2, \dots, a_n)$ množiny $\{1, 2, \dots, n\}$ označme

$$S(a) = \sum_{i=1}^n k_i a_i.$$

Dokažte, že existují dvě permutace b a c , $b \neq c$, pro které je číslo $n!$ dělitelem $S(b) - S(c)$.
(Kanada)

5. V trojúhelníku ABC osa úhlu BAC protíná stranu BC v bodě P , osa úhlu ABC protíná stranu AC v bodě Q . Je známo, že $|\sphericalangle BAC| = 60^\circ$ a že $|AB| + |BP| = |AQ| + |QB|$. Jaké jsou možné velikosti úhlů trojúhelníku ABC ?
(Izrael)

6. Nechť pro celá čísla a, b, c, d platí $a > b > c > d > 0$. Předpokládejme, že

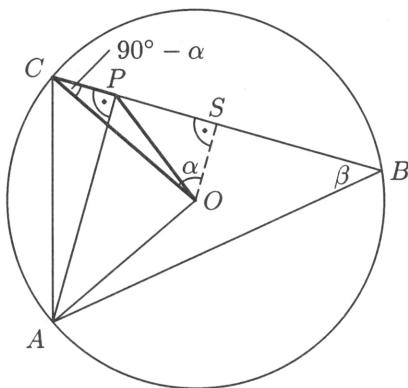
$$ac + bd = (b + d + a - c)(b + d - a + c).$$

Dokažte, že $ab + cd$ není prvočíslo.

(Bulharsko)

Řešení úloh

1. (Podle *Jana Kynčla*.) Označme vnitřní úhly v trojúhelníku ABC obvyklým způsobem α, β, γ . Podle předpokladu platí $\gamma - \beta \geq 30^\circ$, tj. $\sin(\gamma - \beta) \geq \frac{1}{2}$. Bez újmy na obecnosti předpokládejme, že kružnice opsaná uvažovanému trojúhelníku ABC má poloměr $r = 1$. Protože trojúhelník ABC je ostroúhlý, je bod P (pata výšky z vrcholu A) vnitřním bodem strany BC , jejíž střed označíme S .



Obr. 32

Z obr. 32 je patrné, že dokazovaná nerovnost $|\sphericalangle COP| < 90^\circ - \alpha$ je ekvivalentní s nerovností $|CP| < |OP|$, neboť ze vztahu mezi středovým a obvodovým úhlem plyne $|\sphericalangle SOC| = \alpha$, takže $|\sphericalangle BCO| = 90^\circ - \alpha$. Pro velikost tětivy AC s obvodovým úhlem β platí $|AC| = 2 \sin \beta$ a podobně $|CS| = \sin \alpha$. Pomocí těchto vztahů snadno vyjádříme délky úseček CP a OP . Platí totiž $|CP| = |AC| \cos \gamma = 2 \sin \beta \cos \gamma$. Aplikací Pythagorovy věty na pravoúhlý trojúhelník POS obdržíme konečně

$$|OP| = \sqrt{(\sin \alpha - 2 \sin \beta \cos \gamma)^2 + \cos^2 \alpha}.$$

Zbývá tedy dokázat, že pro libovolné vnitřní úhly α, β, γ ostroúhlého trojúhelníku ABC platí nerovnost

$$\sqrt{(\sin \alpha - 2 \sin \beta \cos \gamma)^2 + \cos^2 \alpha} > 2 \sin \beta \cos \gamma.$$

Protože pravá strana této nerovnosti je kladné číslo, přejdeme (po umocnění obou stran nerovnosti na druhou) k ekvivalentní nerovnosti

$$(\sin^2 \alpha - 4 \sin \alpha \sin \beta \cos \gamma + 4 \sin^2 \beta \cos^2 \gamma) + \cos^2 \alpha > 4 \sin^2 \beta \cos^2 \gamma.$$

Po její úpravě zjistíme, že je třeba dokázat nerovnost

$$1 - 4 \sin \alpha \sin \beta \cos \gamma > 0.$$

Užitím vztahu $2 \sin \beta \cos \gamma = \sin(\beta + \gamma) + \sin(\beta - \gamma) = \sin \alpha + \sin(\beta - \gamma)$ však vidíme, že postupně platí

$$\begin{aligned} 1 - 4 \sin \alpha \sin \beta \cos \gamma &= 1 - 2 \sin \alpha (\sin \alpha + \sin(\beta - \gamma)) = \\ &= 1 - 2 \sin^2 \alpha - 2 \sin \alpha \sin(\beta - \gamma) \geq \\ &\geq 1 - 2 \sin^2 \alpha + \sin \alpha = (1 + 2 \sin \alpha)(1 - \sin \alpha) > 0, \end{aligned}$$

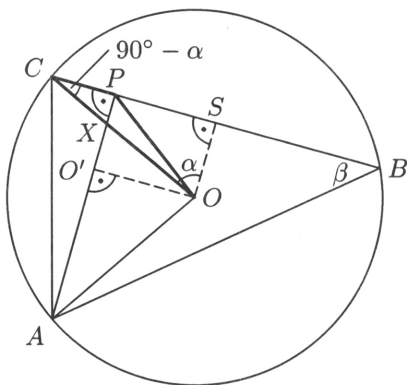
neboť pro každý ostrý úhel α platí současně $1 + 2 \sin \alpha > 0$ a $1 - \sin \alpha > 0$.

Tím je důkaz ukončen.

Jiné řešení. Stejně jako v předchozím řešení vyjdeme z toho, že dokazovaná nerovnost $|\sphericalangle COP| < 90^\circ - \alpha$ je ekvivalentní s nerovností $|CP| < |OP|$. Ukážeme, že pro bod X , v němž výška AP protíná poloměr OC kružnice opsané, platí $|OX| \geq \frac{1}{2}|OC|$, neboli $|OX| \geq |CX|$, což dává

$$|CP| < |CX| \leq |OX| < |OP|,$$

a tím jsme s důkazem hotovi.



Obr. 33

Platí totiž (obr. 33)

$$|\sphericalangle OAP| = |\sphericalangle BAP| - |\sphericalangle BAO| = (90^\circ - \beta) - (90^\circ - \gamma) = \gamma - \beta \geq 30^\circ,$$

a protože $\sin(\gamma - \beta) \geq \frac{1}{2}$, je

$$\frac{1}{2} \leq \sin(\gamma - \beta) = \frac{|OO'|}{|OA|} \leq \frac{|OX|}{|OA|} = \frac{|OX|}{|OC|},$$

neboli $|OX| \geq \frac{1}{2}|OC|$, což jsme potřebovali dokázat.

2. Má-li některé reálné číslo p tu vlastnost, že pro libovolná kladná a, b, c platí nerovnost

$$\frac{a}{\sqrt{a^2 + 8bc}} \geq \frac{a^p}{a^p + b^p + c^p}, \quad (1)$$

je tvrzení úlohy nasnadě: stačí pak totiž sečíst nerovnost (1) s nerovnostmi

$$\frac{b}{\sqrt{b^2 + 8ca}} \geq \frac{b^p}{b^p + c^p + a^p} \quad \text{a} \quad \frac{c}{\sqrt{c^2 + 8ab}} \geq \frac{c^p}{c^p + a^p + b^p}.$$

Uvažujme prozatím obecné p a všimněme si, že z nerovností mezi aritmetickými a geometrickými průměry skupin čísel $\{b^p, c^p\}$ a $\{a^p, a^p, b^p, c^p\}$ plyne

$$\begin{aligned} (a^p + b^p + c^p)^2 - (a^p)^2 &= (b^p + c^p)(a^p + a^p + b^p + c^p) \geq \\ &\geq 2(b^p c^p)^{\frac{1}{2}} \cdot 4(a^p a^p b^p c^p)^{\frac{1}{4}} = 8a^{\frac{1}{2}p} (bc)^{\frac{3}{4}p}. \end{aligned}$$

Odtud po přičtení členu $(a^p)^2$ získáme odhad

$$(a^p + b^p + c^p)^2 \geq 8a^{\frac{p}{2}} (bc)^{\frac{3}{4}p} + (a^p)^2 = a^{\frac{1}{2}p} \cdot (a^{\frac{3}{2}p} + 8(bc)^{\frac{3}{4}p}),$$

ve kterém nyní zvolíme $p = \frac{4}{3}$. Dostaneme

$$(a^{\frac{4}{3}} + b^{\frac{4}{3}} + c^{\frac{4}{3}})^2 \geq a^{\frac{2}{3}} (a^2 + 8bc),$$

odkud je již zřejmé, že nerovnost (1) pro zvolené p platí.

Jiné řešení. (Podle *Tomáše Protivínského*.) Předně si uvědomme, že daná nerovnost je homogenní. Vyhovuje-li totiž dané nerovnosti trojice (a, b, c) kladných čísel, vyhovuje jí rovněž každá trojice (ka, kb, kc) , kde k je libovolné kladné číslo. Bez újmy na obecnosti lze tedy předpokládat, že platí $abc = 1$. Na základě tohoto předpokladu ji můžeme dále

upravit a užitím nerovnosti mezi aritmetickým a geometrickým průměrem dvojice kladných čísel odhadnout např. první sčítanec na levé straně nerovnosti následujícím způsobem:

$$\begin{aligned}\frac{a}{\sqrt{a^2 + 8bc}} &= \frac{a}{\sqrt{a^2 + 8/a}} = \frac{1}{\sqrt{1 + 8/a^3}} = \\ &= \frac{1}{\sqrt{\left(1 + \frac{2}{a}\right)\left(1 - \frac{2}{a} + \frac{4}{a^2}\right)}} \geq \frac{2}{\left(1 + \frac{2}{a}\right) + \left(1 - \frac{2}{a} + \frac{4}{a^2}\right)} = \\ &= \frac{2}{2 + \frac{4}{a^2}} = \frac{1}{1 + \frac{2}{a^2}} = \frac{1}{1 + 2b^2c^2}.\end{aligned}$$

Podobně pro zbylé dva sčítance dostaneme odhady

$$\frac{b}{\sqrt{b^2 + 8ca}} \geq \frac{1}{1 + 2c^2a^2} \quad \text{a} \quad \frac{c}{\sqrt{c^2 + 8ab}} \geq \frac{1}{1 + 2a^2b^2}.$$

Jejich součtem pak dostáváme nerovnost

$$\begin{aligned}\frac{a}{\sqrt{a^2 + 8bc}} + \frac{b}{\sqrt{b^2 + 8ca}} + \frac{c}{\sqrt{c^2 + 8ab}} &\geq \\ &\geq \frac{1}{1 + 2b^2c^2} + \frac{1}{1 + 2c^2a^2} + \frac{1}{1 + 2a^2b^2}.\end{aligned}$$

Ukážeme-li nyní, že pro libovolná kladná čísla a, b, c ($abc = 1$) je splněna nerovnost

$$\frac{1}{1 + 2b^2c^2} + \frac{1}{1 + 2c^2a^2} + \frac{1}{1 + 2a^2b^2} \geq 1,$$

jsme s důkazem hotovi.

Uvedeme-li nejprve levou stranu poslední nerovnosti na společného jmenovatele, dostaneme po několika zjednodušujících úpravách (za využití podmínky $abc = 1$) ekvivalentní nerovnost

$$a^2b^2 + b^2c^2 + c^2a^2 \geq 3.$$

Užijeme-li nyní opět nerovnost mezi aritmetickým a geometrickým průměrem pro trojici kladných čísel a^2b^2, b^2c^2, c^2a^2 , dostaneme

$$a^2b^2 + b^2c^2 + c^2a^2 \geq 3\sqrt[3]{a^4b^4c^4} = 3,$$

v níž rovnost nastává, právě když $a^2b^2 = b^2c^2 = c^2a^2$, tedy pro $a = b = c$ ($= 1$).

Tím je důkaz nerovnosti ukončen.

Jiné řešení. (Podle *T. Laffeye*, vedoucího irské delegace.) I když půjde o řešení, které je ve srovnání s autorským řešením poněkud delší, oceníme na něm to, že je méně trikové. Jeho prvním krokem je úprava, kterou běžně používáme při počítání s homogenními výrazy. Protože platí

$$\begin{aligned} \frac{a}{\sqrt{a^2 + 8bc}} + \frac{b}{\sqrt{b^2 + 8ca}} + \frac{c}{\sqrt{c^2 + 8ab}} &= \\ &= \frac{1}{\sqrt{1 + 8\frac{bc}{a^2}}} + \frac{1}{\sqrt{1 + 8\frac{ca}{b^2}}} + \frac{1}{\sqrt{1 + 8\frac{ab}{c^2}}}, \end{aligned}$$

zavedeme nové proměnné

$$x = \frac{bc}{a^2}, \quad y = \frac{ca}{b^2}, \quad z = \frac{ab}{c^2}$$

splňující podmínku $xyz = 1$ a ukážeme, že pro taková kladná čísla x, y, z platí

$$\frac{1}{\sqrt{1 + 8x}} + \frac{1}{\sqrt{1 + 8y}} + \frac{1}{\sqrt{1 + 8z}} \geq 1.$$

Označíme ještě $u = \sqrt{1 + 8x}$, $v = \sqrt{1 + 8y}$, $w = \sqrt{1 + 8z}$ a dokazovanou nerovnost

$$\frac{1}{u} + \frac{1}{v} + \frac{1}{w} \geq 1$$

upravíme zřejmým způsobem na tvar $(uv + vw + wu)^2 \geq (uvw)^2$. Obě strany poslední nerovnosti nyní vyjádříme takto:

$$\begin{aligned} (uv + vw + wu)^2 &= \\ &= (uv)^2 + (vw)^2 + (wu)^2 + 2uvw(u + v + w) = \\ &= 3 + 16(x + y + z) + 64(xy + xz + yz) + 2uvw(u + v + w), \\ (uvw)^2 &= (1 + 8x)(1 + 8y)(1 + 8z) = \\ &= 1 + 8(x + y + z) + 64(xy + xz + yz) + 512. \end{aligned}$$

Náš úkol se proto zřejmě redukuje na důkaz nerovnosti

$$4(x + y + z) + uvw(u + v + w) \geq 255.$$

Protože $xyz = 1$, platí $(x + y + z) \geq 3(xyz)^{\frac{1}{3}} = 3$ a také $xy + xz + yz \geq 3((xyz)^2)^{\frac{1}{3}} = 3$. S pomocí AG-nerovnosti odhadneme i výraz $uvw(u + v + w)$:

$$\begin{aligned} uvw(u + v + w) &\geq uvw \cdot 3(uvw)^{\frac{1}{3}} = 3(u^2v^2w^2)^{\frac{2}{3}} = \\ &= 3((1 + 8x)(1 + 8y)(1 + 8z))^{\frac{2}{3}} = \\ &= 3(1 + 8(x + y + z) + 64(xy + xz + yz) + 512)^{\frac{2}{3}} \geq \\ &\geq 3(1 + 8 \cdot 3 + 8^2 \cdot 3 + 8^3)^{\frac{2}{3}} = \\ &= 3[(1 + 8)^3]^{\frac{2}{3}} = 3 \cdot 81 = 243. \end{aligned}$$

Sečtením obou odhadů dostaneme nerovnost, kterou jsme potřebovali dokázat.

3. (Podle *Martina Tancera*.) Soutěžní úlohy budeme značit písmenem p , soutěžící dívky písmenem g , soutěžící chlapce písmenem b . Trojici (g, p, b) nazveme *spojením*, pokud úlohu p vyřešili dívka g i chlapec b . Řekneme, že spojení (g, p, b) je G -vyhovující (resp. B -vyhovující), pokud byla příslušná úloha p vyřešena alespoň třemi dívkami (resp. alespoň třemi chlapci). Naším cílem je úvahou o počtu N všech spojení ukázat, že některé spojení je jak G -vyhovující, tak i B -vyhovující.

Protože pro každé g a každé b existuje alespoň jedno spojení $(g, *, b)$, je pro každé g počet všech spojení $(g, *, *)$ roven $21 + n(g)$, kde $n(g)$ je vhodné nezáporné celé číslo. Celkový počet N všech spojení je pak dán vzorcem

$$N = \sum_g (21 + n(g)) = 441 + \sum_g n(g)$$

(využili jsme toho, že počty chlapců i dívek se rovnají 21).

Připusťme nyní, že *některá* dívka g vyřešila *šest* (různých) úloh p_i a že přitom žádné z příslušných spojení $(g, p_i, *)$ není B -vyhovující, tedy že existují (při pevném i) vždy nejvýše dvě spojení $(g, p_i, *)$. Protože dívka g dle zadání žádnou další úlohu nevyřešila, je pak tato dívka zastoupena v nejvýše $2 \cdot 6 = 12$ spojeních, a to je spor, protože každá z dívek je zastoupena nejméně v 21 spojeních. Tak jsme zdůvodnili, že *každá* dívka g vyřešila *nejvýše pět* úloh p_i takových, že žádné ze spojení $(g, p_i, *)$ není B -vyhovující. Znamená to, že počet spojení $(g, *, *)$, která nejsou B -vyhovující, nepřevyšuje číslo 10. Odtud pro každé g plyne, že počet těch spojení $(g, *, *)$, která jsou naopak B -vyhovující, je alespoň

$21 + n(g) - 10$, tedy alespoň $11 + n(g)$. Sečteme-li tyto odhady přes všechna g , zjistíme, že počet všech B -vyhovujících spojení je alespoň

$$\sum_g (11 + n(g)) = 231 + \sum_g n(g) = N - 210.$$

Úplně stejně zdůvodníme, že počet všech G -vyhovujících spojení je alespoň $N - 210$. Kdyby tudíž žádné spojení nebylo současně B -vyhovující i G -vyhovující, musela by platit nerovnost

$$N \geq (N - 210) + (N - 210), \quad \text{neboli} \quad N \geq 420.$$

To ale není možné, neboť $N \geq 441$. Důkaz je ukončen.

4. Tvrzení úlohy dokážeme sporem, a to úvahou o součtu všech čísel $S(a)$.¹ Označme zmíněný součet písmenem S :

$$S = \sum_a S(a).$$

Připustíme tedy, že žádný z rozdílů $S(b) - S(c)$ není násobkem čísla $n!$, takže každé z $n!$ čísel $S(a)$ dává při dělení číslem $n!$ jiný zbytek. Všech možných zbytků je ale právě $n!$, proto je každý ze zbytků $0, 1, 2, \dots, n! - 1$ zastoupen mezi zbytky čísel $S(a)$ právě jednou. Znamená to, že součet S všech čísel $S(a)$ dává při dělení číslem $n!$ stejný zbytek jako součet

$$S^* = 0 + 1 + 2 + \dots + (n! - 1) = n! \cdot \frac{n! - 1}{2}.$$

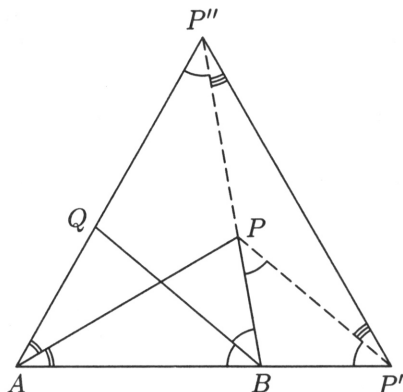
Součet S lze však vypočítat přímo: protože pro libovolná $i, j \in \{1, 2, \dots, n\}$ existuje právě $(n - 1)!$ permutací a s vlastností $a_i = j$, je koeficient čísla k_i v součtu S roven $(n - 1)! \cdot (1 + 2 + \dots + n) = n! \cdot \frac{1}{2}(n + 1)$, takže platí rovnost

$$S = n! \cdot \frac{n + 1}{2} \cdot \sum_{i=1}^n k_i.$$

Protože číslo n je dle předpokladu liché, je zlomek $\frac{n + 1}{2}$ celé číslo, takže vyjádřená hodnota S je násobkem čísla $n!$. Pro žádné $n > 1$ však zlomek $\frac{n! - 1}{2}$ z vyjádření čísla S^* nemá sudý čitatel, takže součet S^* není násobkem čísla $n!$. Čísla S a S^* tedy dávají při dělení číslem $n!$ různé zbytky, což je kýžený spor.

¹ S politováním konstatujeme, že uvažovat o tomto součtu nenapadlo žádného našeho soutěžícího. Nikdo z nich tuto úlohu, patrně nejjednodušší z celé soutěžní šestice, nevyřešil.

5. Uvažujme obvyklé značení vnitřních úhlů trojúhelníku ABC . Podle zadání je tedy $\alpha = 60^\circ$. Na polopřímce AB (obr. 34) sestrojme bod P' za bodem B tak, že platí $|BP| = |BP'|$, neboli $|AP'| = |AB| + |BP|$.



Obr. 34

Trojúhelník $BP'P$ je tedy rovnoramenný s vnitřními úhly při základně $P'P$ velikosti $\frac{1}{2}\beta$. Na polopřímce AQ uvažujme podobně bod P'' , pro který platí $|QP''| = |QB|$, neboli $|AP''| = |AQ| + |QB|$. Trojúhelník QBP'' je tedy rovněž rovnoramenný a pro jeho úhly platí $|\sphericalangle QP''B| = |\sphericalangle QBP''| = \frac{1}{2}\beta$. Protože platí

$$|AP'| = |AB| + |BP| = |AQ| + |QB| = |AP''|,$$

je i trojúhelník $AP'P''$ rovnoramenný, a dokonce rovnostranný, neboť jeden z jeho úhlů má velikost 60° . Protože bod P leží na ose AP úhlu při jeho vrcholu A , platí $|PP'| = |PP''|$ a zároveň $|\sphericalangle AP''P| = |\sphericalangle AP'P| = \frac{1}{2}\beta$. A protože $|\sphericalangle AP''P| = |\sphericalangle QP''P| = |\sphericalangle QP''B| = \frac{1}{2}\beta$, leží body B , P a P'' v přímce, proto je bod P'' totožný s vrcholem C trojúhelníku ABC .

Protože trojúhelník BCQ je rovnoramenný, platí

$$\frac{\beta}{2} = \gamma = 120^\circ - \beta,$$

tudíž $\beta = 80^\circ$ a $\gamma = 40^\circ$.

Závěr: Velikosti vnitřních úhlů trojúhelníku ABC jsou tedy $\alpha = 60^\circ$, $\beta = 80^\circ$ a $\gamma = 40^\circ$.

6. Tvzení úlohy dokážeme sporem. Pripustme tedy, že za podmínek úlohy je číslo $p = ab + cd$ prvočíslo. Rovnost ze zadání nejprve upravíme:

$$\begin{aligned} ac + bd &= (b + d)^2 - (a - c)^2, \\ ac + bd &= b^2 + 2bd + d^2 - a^2 + 2ac - c^2, \\ a^2 - ac + c^2 &= b^2 + bd + d^2. \end{aligned}$$

Všechny způsoby řešení, které známe, jsou založeny na objevení poznatku, že číslo $b^2 + bd + d^2$ je násobkem prvočísla p , a to s využitím některé rovnosti typu $r(b^2 + bd + d^2) = ps$, kde r a s jsou přirozená čísla, přičemž číslo r je buď samo menší než p , nebo je součinem čísel menších než p . Uvedme tři takové rovnosti

$$\begin{aligned} (ac + bd)(b^2 + bd + d^2) &= p(ad + bc), \\ (b - c)(b + c)(b^2 + bd + d^2) &= p(ab - cd - bc), \\ (a - d)(a + d)(b^2 + bd + d^2) &= p(ab - cd + ad) \end{aligned}$$

a dokažme například první z nich. Platí

$$\begin{aligned} (ac + bd)(b^2 + bd + d^2) &= ac(b^2 + bd + d^2) + bd(a^2 - ac + c^2) = \\ &= ab^2c + abcd + acd^2 + a^2bd - abcd + bc^2d = \\ &= ab \cdot bc + cd \cdot ad + ab \cdot ad + cd \cdot bc = \\ &= (ab + cd)(ad + bc) = p(ad + bc). \end{aligned}$$

Z dokázané rovnosti plyne, že prvočíslo p dělí alespoň jedno z přirozených čísel $ac + bd$, $b^2 + bd + d^2$. První z nich však dělit nemůže, neboť

$$p - (ac + bd) = (ab + cd) - (ac + bd) = (a - d)(b - c) > 0$$

(připomeňme, že $a > b > c > d$). Platí tedy $p \mid (b^2 + bd + d^2)$. Z nerovností $bd < b^2 < ab$ a $d^2 < cd$ ovšem plyne

$$b^2 + bd + d^2 < ab + ab + cd < 2(ab + cd) = 2p,$$

tudíž nutně $b^2 + bd + d^2 = p$. Odtud plyne, že čísla b , d jsou nesoudělná. Rovnost $b^2 + bd + d^2 = ab + cd$ lze upravit do tvaru $(b + d - a)b = d(c - d)$, z něhož vzhledem k nesoudělnosti čísel b , d plyne, že přirozené číslo $c - d$ je násobkem čísla b . To je však nemožné, neboť $c - d < c < b$. Tvzení úlohy je tím dokázáno.

Osmý ročník Středoevropské olympiády v informatice

Ve dnech 10.–17. srpna 2001 proběhla v maďarském Zalaegerszegu Středoevropská olympiáda v informatice (Central European Olympiad in Informatics, CEOI 2001).

Vedoucím české reprezentace byl jmenován RNDr. *Tomáš Pitner*, Dr., z Masarykovy univerzity v Brně, Fakulty informatiky. Úloha zástupce vedoucího týmu byla svěřena *Danielu Královi* z Matematicko-fyzikální fakulty Univerzity Karlovy v Praze.

Soutěž probíhala regulérně podle pravidel CEOI za maximální podpory především maďarského ministerstva školství, ale též složek místní a regionální veřejné správy (město Zalaegerszeg, kraj Zala) a John von Neumann Computer Society. Vlastní soutěž se konala na Zrínyi Miklós Gymnáziu Zalaegerszeg. Hmotné i personální zabezpečení akce bylo na velmi vysoké úrovni.

Novinkou letošního ročníku CEOI byla (po vzoru IOI) zásadní modernizace prostředí, v němž studenti při soutěži pracovali a v němž také byly jejich výsledky vyhodnocovány. Namísto dosud užívaných kompilátorů Turbo Pascal a Borland C/C++ a jejich integrovaného vývojového prostředí bylo poprvé při soutěži použito třicetidvoubitových překladačů GNU C++ a Free Pascal s vývojovým prostředím RHIDE. Soutěžící vyvíjeli a ladili svá díla pod systémem Windows, vyhodnocení probíhalo pod operačním systémem Linux. Vyhodnocení se provádělo plně automaticky pomocí předem připravených sad testovacích dat.

Reprezentační tým ČR tvořili (v abecedním pořadí): *Josef Cibulka* z Akademického gymnázia Štěpánská, Praha; *Jiří Danihelka* ze SPŠ Čapkova, Písek; *Martin Hamrle* z Gymnázia Pelhřimov a *Jiří Štěpánek* z Gymnázia Brno, tř. Kpt. Jaroše.

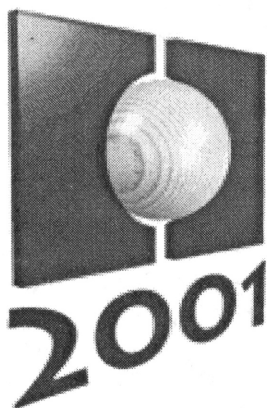
Josef Cibulka a Jiří Danihelka získali bronzové medaile, přičemž celkově se naši reprezentanti umístili (viz příložená kopie výsledkové listiny) na 18., 19., 35. a 46. místě z celkového počtu 49 hodnocených soutěžících.

Celkově lze konstatovat, že výsledky na CEOI 2001 byly pro ČR příznivější než v roce minulém.

Rozhodnutím mezinárodního výboru CEOI bylo pořadatelem CEOI v roce 2002 určeno Slovinsko s tím, že pokud se CEOI nebude moci uskutečnit tam, je náhradním místem CEOI 2002 Slovenská republika.

13. mezinárodní olympiáda v informatice

V pořadí třináctý ročník mezinárodní olympiády v informatice IOI 2001 (International Olympiad in Informatics) se konal ve dnech 14.–21. 7. 2001 ve městě Tampere ve Finsku. Soutěž IOI je pořádána pod záštitou UNESCO a je nejmladší mezinárodní předmětovou olympiádou středoškoláků. Její ohlas ve světě však v posledních letech stále narůstá a každoročně se zvyšuje počet zúčastněných zemí. Letos stoupl jejich počet na 74 a očekává se, že v příštím roce jich bude již kolem osmdesáti.



Reprezentační družstvo České Republiky pro IOI 2001 bylo vybráno na základě výsledků celostátního kola jubilejního 50. ročníku Matematické olympiády — kategorie P. Druž-

stvo mělo následující složení: *Pavel Čížek* (student gymnázia v Kralupích nad Vltavou), *Roman Krejčík* (absolvent gymnázia Ch. Dopplera v Praze 5), *Marek Sulovský* (absolvent gymnázia na tř. Kpt. Jaroše v Brně) a *Miloslav Trmač* (absolvent Biskupského gymnázia v Brně). Vedením družstva byli pověřeni doc. *Pavel Töpfer* a Mgr. *Martin Mareš*, oba z Matematicko-fyzikální fakulty Univerzity Karlovy v Praze.

Stejně jako v minulých letech byla vlastní soutěž rozdělena do dvou soutěžních dnů, v každém z nich řešili soutěžící tři úlohy. Letošní soutěžní úlohy byly algoritmicky mimořádně obtížné, v soutěži se objevily také některé netradiční formy úloh. K nim patřily interaktivní úlohy, v nichž program nemá k dispozici všechna vstupní data najednou, ale dostává je postupně a musí na ně průběžně reagovat (typickým příkladem je program hrající nějakou hru a reagující na tahy protivníka). Jinou netradiční úlohou byla tzv. úloha s otevřeným vstupem, v níž se netestoval vytvořený program, ale ověřovala se správnost výsledků, kterých program dosáhl pro předem známá vstupní data. Velkou novinkou letošního ročníku IOI byla zásadní modernizace prostředí, v němž studenti při soutěži

pracovali a v němž také byly jejich výsledky vyhodnocovány. Namísto dosud v IOI užívaných DOSovských kompilátorů Turbo Pascal a Borland C/C++ a jejich integrovaného vývojového prostředí byl poprvé při soutěži použit operační systém Linux, překladače Free Pascal a GNU C/C++, vývojové prostředí RHIDE. Alternativně bylo možné pracovat také pod operačním systémem Windows. Svá řešení úloh soutěžící odevzdávali k vyhodnocení prostřednictvím nově vyvinutého webového rozhraní. Vyhodnocení se provádělo plně automaticky pomocí předem připravených sad testovacích dat. Různá kvalita navržených algoritmů byla odlišena na základě dosti přísně nastavených časových limitů.

Mezi 272 účastníků bylo rozděleno 23 zlatých, 45 stříbrných a 68 bronzových medailí. Některou z medailí tak podle regulí IOI získala přibližně polovina soutěžících. Výkony našich studentů v soutěži byly poměrně dobré a jejich výsledky nás řadí kolem 20. místa v celkovém pořadí. Olympiáda IOI je ovšem výhradně soutěží jednotlivců a oficiální výsledky družstev se zde vůbec nevyhlašují (neexistuje ani žádná metodika, jak je určovat — zda podle počtu získaných bodů, medailí nebo třeba pořadí). Nejúspěšnějšími zeměmi v letošní IOI byly Slovensko a Singapur se dvěma zlatými a dvěma stříbrnými medailemi, naši získali jednu stříbrnou a dvě bronzové medaile.

Výsledky našich studentů:

36.	Pavel Čížek	344 bodů	stříbrná medaile
85.	Miloslav Trmač	237 bodů	bronzová medaile
100.	Marek Sulovský	197 bodů	bronzová medaile
150.	Roman Krejčík	127 bodů	

Příští 14. ročník mezinárodní olympiády v informatice IOI 2002 se bude konat ve dnech 18.–25. 8. 2002, jeho hostitelem bude Korea. Soutěž bude probíhat na univerzitě Kyung Hee ve městě Young-in, asi 40 km jižně od hlavního města Soulu. Korejští pořadatelé příštího ročníku olympiády během jednání probíhajících ve Finsku pozvali všechny zúčastněné země k účasti na IOI 2002. Mezinárodní výbor IOI také rozhodl, že další budoucí ročníky IOI budou hostit po řadě USA (2003), Řecko (2004) a Polsko (2005).

1. Mobilní telefony

Síť mobilních telefonů čtvrté generace v oblasti poblíž Tampere pracuje takto: celá oblast je rozdělena na čtverce tvořící matici $S \times S$. Řádky a sloupce matice jsou číslovány od 0 do $S - 1$. V každém čtverci je umístěna jedna základnová stanice. Počet aktivních mobilních telefonů nacházejících se v jednotlivých čtvercích se mění, jak se jednotlivé telefony pohybují mezi čtverci, případně se vypínají a zapínají. Každá základnová stanice čas od času oznamuje hlavní ústředně, jak se počet telefonů v jejím čtverci změnil oproti předchozímu ohlášenému stavu.

Napište program pro ústřednu, který bude tyto zprávy přijímat a na základě těchto informací odpovídat na dotazy, jaký je právě počet aktivních telefonů na zadaném území obdélníkového tvaru.

Vstup a výstup. Program čte vstupní data ve formě celých čísel ze svého standardního vstupu a zapisuje výsledky opět ve formě celých čísel do svého standardního výstupu. Vstup má následující tvar: každý řádek obsahuje jeden příkaz, první číslo na řádku určuje typ příkazu, zbývající čísla jsou jeho parametry.

<i>Příkaz</i>	<i>Parametry</i>	<i>Význam</i>
0	S	Inicializuje matici tak, že její velikost bude $S \times S$ a bude obsahovat samé nuly. Tento příkaz bude použit právě jednou, a to jako první v celém vstupu.
1	$X \ Y \ A$	Přičte A k počtu aktivních telefonů ve čtverci (X, Y) . A může být jak kladné, tak záporné.
2	$L \ B \ R \ T$	Dotaz na celkový počet aktivních telefonů ve všech čtvercích (X, Y) takových, že $L \leq X \leq R$, $B \leq Y \leq T$.
3		Ukončí program. Tento příkaz bude použit právě jednou, a to jako poslední v celém vstupu.

Parametry příkazů budou vždy korektní, není důvod je jakkoliv kontrolovat. Pokud bude A záporné, můžete předpokládat, že počet aktivních telefonů v daném čtverci nikdy neklesne pod nulu. Čtverce jsou indexovány od nuly, tzn. pro matici velikosti 4×4 bude $0 \leq X \leq 3$ a $0 \leq Y \leq 3$.

Program nijak neodpovídá na vstupní řádky s příkazy 0, 1, ani 3. Po každém řádku s příkazem 2 program vypíše na výstup jeden řádek obsahující jedno celé číslo, které je odpovědí na daný dotaz.

Návod k programování vstupu a výstupu. Používáte-li ke komunikaci standardní prostředky jazyka, postupujte podle následujících příkladů. V těchto příkladech integerová proměnná `last` obsahuje poslední číslo načtené ze vstupu a integerová proměnná `answer` odpověď vašeho programu.

Programujete-li v C++ a používáte `iostreams`, čtete vstup a zapisujete výstup následovně:

```
cin>>last;
cout<<answer<<endl<<flush;
```

Programujete-li v C nebo C++ a používáte `scanf` a `printf`, postupujte takto:

```
scanf("%d", &last);
printf("%d\n", answer); fflush(stdout);
```

Programujete-li v Pascalu, použijte:

```
Read(last); ... Readln;
Writeln(answer);
```

Příklad

stdin	stdout	vysvětlení
0 4		Vytvoří prázdnou matici velikosti 4×4 .
1 1 2 3		Zvýšení hodnoty na (1, 2) o 3.
2 0 0 2 2		Dotaz na celkový počet v obdélníku $0 \leq X \leq 2$, $0 \leq Y \leq 2$.
	3	Odpověď na dotaz.
1 1 1 2		Zvýšení hodnoty na (1, 1) o 2.
1 1 2 -1		Snížení hodnoty na (1, 2) o 1.
2 1 1 2 3		Dotaz na celkový počet v obdélníku $1 \leq X \leq 2$, $1 \leq Y \leq 3$.
	4	Odpověď na dotaz.
3		Ukončení programu.

Omezení

Velikost matice	$S \times S$	$1 \times 1 \leq S \times S \leq 1024 \times 1024$
Všechny hodnoty ve čtvercích po celou dobu výpočtu	V	$0 \leq V \leq 2^{15} - 1 = 32\,767$

Změny počtu aktivních telefonů	A	$-2^{15} \leq A \leq 2^{15} - 1$
Počet příkazů na vstupu	U	$3 \leq U \leq 60\,002$
Maximální počet telefonů v celé matici	M	$M = 2^{30}$

Z 20 vstupů zadaných při testování úlohy jich 16 bude používat matici velikosti nejvýše 512×512 .

2. Pěťadvacetštiny

Tajné zprávy, které si posílá Děda Mráz se svými pomocníčky, jsou obvykle psány ve zvláštním jazyce, kterému říkají pěťadvacetština, možná právě proto, že se zapisuje v 25-znakové abecedě. Tato abeceda je podobná latince s jednou jedinou výjimkou: chybí v ní písmeno Z. Abeceda tedy obsahuje 25 písmen latinky od A do Y ve stejném pořadí. Každé slovo 25-štiny je tvořeno právě 25 různými písmeny. Slovo si můžeme zapsat po řádcích do tabulky 5×5 — např. slovo ADJPTBEKQUCGLRVFINSWHMOXY se zapíše takto:

A	D	J	P	T
B	E	K	Q	U
C	G	L	R	V
F	I	N	S	W
H	M	O	X	Y

Přípustná slova 25-štiny jsou právě ta, u nichž jsou písmena v každém řádku i v každém sloupci tabulky uspořádána vzestupně. Tedy slovo ADJPTBEKQUCGLRVFINSWHMOXY je přípustné, zatímco slovo ADJPTBEGQUC-KLRVFINSWHMOXY (liší se prohozením písmen G a K) přípustné není (vzestupné pořadí písmen je porušeno ve druhém i třetím sloupci).

Děda Mráz má slovník, který obsahuje právě všechna přípustná slova 25-štiny uspořádaná vzestupně (lexikograficky). Slova ve slovníku jsou očíslována pořadovými čísly od 1. Např. slovo ABCDEFGHIJKLMNOP-QRSTUVWXY má číslo 1 a slovo ABCDEFGHIJKLMNOPQRSTUWVXY (T a U prohozena) má číslo 2.

Slovník je však veliký, převeliký. Napište proto program, který bude umět nalézt ke slovu jeho pořadové číslo a naopak pro zadané pořadové číslo vypsát, kterému slovu toto číslo náleží. Slovo ve slovníku určitě není více než 2^{31} .

Vstup. Vstupní soubor se jmenuje `twofive.in` a je tvořen vždy dvěma řádky. První řádek obsahuje jediný znak — W nebo N. Pokud je

to W, pak druhý řádek obsahuje jedno přípustné slovo 25-štiny (tj. řetězec o 25 znacích); pokud je na prvním řádku N, potom druhý řádek obsahuje pořadové číslo nějakého přípustného slova 25-štiny.

Výstup. Výstupní soubor se jmenuje `twofive.out` a je jednořádkový. Pokud druhý řádek vstupního souboru obsahoval slovo 25-štiny, obsahuje výstupní soubor pořadové číslo tohoto slova. Jestliže druhý řádek vstupního souboru obsahoval pořadové číslo, potom je obsahem výstupního souboru odpovídající slovo 25-štiny.

Příklady vstupu a výstupu

```
twofive.in
```

```
W
```

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

```
twofive.out
```

```
2
```

```
twofive.in
```

```
N
```

```
2
```

```
twofive.out
```

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

3. Gameska

Byla jednou jedna hra pro dva hráče, která se hrála na kruhové desce se sedmi jamkami rozmístěnými podél jejího okraje. Mimo to, každý z hráčů má svůj bank. Hra začíná náhodným rozmístěním dvaceti kuliček do jamek tak, že v každé jamce budou nejméně 2 a nejvýše 4 kuličky. Oba hráči se v tazích pravidelně střídají. Hráč na tahu si zvolí nějakou neprázdnou jamku, vezme z ní do ruky všechny kuličky a dokud mu nějaké kuličky v ruce zbývají, obchází jamky jednu po druhé ve směru hodinových ručiček počínaje jamkou bezprostředně následující po té, kterou si vybral. Přitom provádí následující akce:

- ▷ Má-li v ruce více než jednu kuličku: Pokud jamka, u které stojí, již obsahuje 5 kuliček, vezme si z ní jednu kuličku do svého banku. V opačném případě jednu kuličku ze své ruky do jamky vloží.
- ▷ Má-li v ruce jedinou kuličku: Pokud jamka, u které stojí, obsahuje nejméně 1 a nejvýše 4 kuličky, pak všechny kuličky z jamky i tu, kterou drží v ruce, přesune do svého banku. V opačném případě (jamka je prázdná nebo s pěti kuličkami) vloží kuličku, kterou drží v ruce, do soupeřova banku.

Hra končí, jakmile se všechny jamky vyprázdní. Zvítězí hráč, který má v banku více kuliček.

Pro hráče, který táhne jako první, vždy existuje vítězná strategie. Napište program, který hraje tuto hru jako první hráč a ve hře zvítězí. Protihráč použitý při testování hraje optimálně, tzn. jakmile může, vyhraje.

Vstup a výstup. Program čte vstupní data ze standardního vstupu a zapisuje výsledky do standardního výstupu. Váš program je hráč 1, soupeř je hráč 2. Nejprve musí váš program načíst jeden řádek se 7 celými čísly $p_1, \dots, p_7$, což jsou počáteční počty kuliček v jednotlivých jamkách. Jamky na desce jsou očíslovány od 1 do 7 ve směru hodinových ručiček.

Hra poté začíná s tímto obsahem jamek a prázdnými banky obou hráčů. Váš program hraje následovně:

- ▷ Je-li váš program na tahu, vypíše číslo jamky, kterou si zvolil pro svůj tah.
- ▷ Je-li na tahu soupeř, váš program načte číslo jamky, kterou si soupeř zvolil (tj. té, u které soupeř svůj tah začíná odebráním kuliček).

Pomůcka. Máte k dispozici program (pod Linuxem `ioiwari2`, pod Windows `ioiwari2.exe`), který z jednoho počátečního rozložení kuliček do jamek hraje hru optimálně jako hráč 2. Po spuštění vypíše na svůj standardní výstup první řádek očekávaný vaším programem; na tomto řádku je popsáno počáteční rozložení kuliček ve tvaru 4 3 2 4 2 3 2.

Poté program hru hraje, ze svého vstupu čte tahy prvního hráče a do svého výstupu na ně odpovídá tahy svými, přičemž celý dialog ukládá do souboru `ioiwari.out`. Můžete si například `ioiwari2` spustit v jednom okně a váš program v druhém a tahy mezi okny přenášet ručně.

Návod k programování vstupu a výstupu. Používáte-li ke komunikaci standardní prostředky jazyka, postupujte podle následujících příkladů. V těchto příkladech integerová proměnná `last` obsahuje poslední číslo načtené ze vstupu a integerová proměnná `mymove` obsahuje váš tah.

Programujete-li v C++ a používáte `iostreams`, čtete vstup a zapisujete výstup následovně:

```
cout<<mymove<<endl<<flush;  
cin>>last;
```

Programujete-li v C nebo C++ a používáte `scanf` a `printf`, postupujte takto:

```
printf("%d\n", mymove); fflush(stdout);
```

```
scanf("%d", &last);
```

Programujete-li v Pascalu, použijte:

```
Writeln(mymove);
```

```
Readln(last);
```

Příklad. Následující tabulka ukazuje jeden z možných průběhů hry o šesti tazích:

	<i>Obsah jamek a banků po provedení tahu</i>								
	1.	2.	3.	4.	5.	6.	7.	Bank1	Bank2
Začátek hry	4	3	2	4	2	3	2	0	0
Tah 1. hráče: 2	4	0	3	5	0	3	2	3	0
Tah 2. hráče: 3	4	0	0	4	1	4	0	3	4
Tah 1. hráče: 5	4	0	0	4	0	0	0	8	4
Tah 2. hráče: 4	0	0	0	0	1	1	1	8	9
Tah 1. hráče: 5	0	0	0	0	0	0	1	10	9
Tah 2. hráče: 7	0	0	0	0	0	0	0	11	9

Hodnocení. Při testování váš program za každou vyhranou hru získá 4 body, za každou remízu 2 body a za prohrané hry 0 bodů.

4. Dvojšifra

Nový šifrovací algoritmus AES (Advanced Encryption Standard) pracuje se třemi bloky velikosti 128 bitů. Pro daný blok zprávy p (původní text) a blok klíče k spočte šifrovací funkce E tohoto algoritmu blok c (zašifrovaný text):

$$c = E(p, k).$$

Standard AES rovněž definuje dešifrovací funkci D , která je inverzní k E :

$$D(E(p, k), k) = p, \quad E(D(c, k), k) = c.$$

Šifru AES je možné zdvojit, čímž vznikne šifra Double AES používající dva nezávislé klíče k_1 a k_2 . Double AES zprávu nejprve zašifruje pomocí k_1 a výsledek ještě pomocí k_2 :

$$c_2 = E(E(p, k_1), k_2).$$

Vášim úkolem je zjistit klíče k_1 a k_2 pro některé konkrétní zprávy zašifrované algoritmem Double AES. Vždy dostanete jak původní text p , tak odpovídající zašifrovaný text c_2 . Navíc víte, že nenulové bity se v obou

klíči (k_1 i k_2) mohou vyskytovat pouze v nejvyšších $4 \cdot s$ bitech bloku; ostatní bity obou klíčů jsou nulové. Parametr s je součástí vstupu.

Máte k dispozici knihovnu implementující funkce E a D (šifrovací a dešifrovací algoritmus AES).

Odevzdáváte nalezené klíče, nikoliv programy!

Vstup. Máte připraveny soubory `double1.in` až `double10.in`, každý z nich obsahuje jedno konkrétní zadání pro tuto úlohu. Každý vstupní soubor sestává ze tří řádků. Na prvním z nich je celé číslo s , na druhém blok p původního textu a na třetím blok c_2 získaný z p zašifrováním algoritmem Double AES. Oba bloky jsou zapsány jako znakové řetězce, každý z nich je tvořen 32 hexadecimálními ciframi (‘0’ až ‘9’, ‘A’ až ‘F’). Všechna zadání jsou řešitelná.

Součástí knihovny jsou rovněž funkce pro převod bloků do jejich hexadecimálního zápisu a zpět.

Výstup. Vaším úkolem je odevzdat (submitovat) 10 výstupních souborů odpovídajících jednotlivým vstupním souborům. Každý z těchto souborů bude obsahovat tři řádky: Na prvním řádku bude text

```
#FILE double  $i$ 
```

kde i značí číslo příslušného vstupního souboru. Na řádku druhém bude klíč k_1 a na třetím klíč k_2 takové, že $c_2 = E(E(p, k_1), k_2)$. Oba klíče budou opět zapsány jako řetězce 32 hexadecimálních číslic. Pokud má úloha více řešení, uveďte libovolné jedno z nich.

Příklad. V tomto příkladu použijeme vstupní soubor číslo 0.

```
double0.in
```

```
1
```

```
00112233445566778899AABBCCDDEEFF
```

```
6323B4A5BC16C479ED6D94F5B58FF0C2
```

Možný výstupní soubor:

```
#FILE double 0
```

```
A00000000000000000000000000000000
```

```
700000000000000000000000000000000
```

Knihovna. K dispozici je knihovna implementující algoritmy AES jak pro FreePascal (Linux: `aeslibp.p`, `aeslibp.ppu`, `aeslibp.o`; Windows: `aeslibp.p`, `aeslibp.ppw`, `aeslibp.ow`), tak pro GNU C/C++ (Linux i Windows: `aeslibc.h`, `aeslibc.o`) a příklady použití těchto knihoven

(aestoolp.pas resp. aestoolc.c). Interface obou knihoven je popsán v následujících řádcích.

type

```
HexStr = String [ 32 ]; {only '0'..'9', 'A'..'F' }  
Block = array [ 0..15 ] of Byte; {128 bits }
```

```
procedure HexStrToBlock ( const hs: HexStr; var b: Block );
```

```
procedure BlockToHexStr ( const b: Block; var hs: HexStr );
```

```
procedure Encrypt ( const p, k: Block; var c: Block );
```

```
{c = E(p,k) }
```

```
procedure Decrypt ( const c, k: Block; var p: Block );
```

```
{p = D(c,k) }
```

```
typedef char HexStr[33]; /* '0'..'9', 'A'..'F', '0'-terminated */
```

```
typedef unsigned char Block[16]; /* 128 bits */
```

```
void hexstr2block ( const HexStr hs, /* out-param */ Block b );
```

```
void block2hexstr ( const Block b, /* out-param */ HexStr hs );
```

```
void encrypt ( const Block p, const Block k, /* out-param */ Block c );
```

```
/* c = E(p,k) */
```

```
void decrypt ( const Block c, const Block k, /* out-param */ Block p );
```

```
/* p = D(c,k) */
```

Omezení. Počet s použitých hexadecimalních číslic klíčů je vždy alespoň 1 a nejvýše 5.

Nápověda. Existuje řešení, které libovolné zadání s tímto omezením na velikost s vyřeší do 10 sekund.

5. Skladiště

Jistá nejmenovaná finská technologická společnost má velké obdélníkové skladiště, jehož jedinými dvěma zaměstnanci jsou skladník a ředitel skladiště. Stranám skladiště budeme říkat (po směru hodinových ručiček) levá, horní, pravá a dolní. Celá plocha skladiště je rozdělena na stejně velké čtverce, které jsou uspořádány do řad a sloupců. Řady jsou číslovány shora dolů přirozenými čísly 1, 2, ...; sloupce zleva doprava opět 1, 2, ...

Ve skladišti jsou umístěny kontejnery označené jednoznačnými identifikačními čísly. Každý kontejner zaujímá plochu jednoho čtverce. Skladiště je tak velké, že celkový počet všech kontejnerů je menší než počet řad i než počet sloupců. Ze skladiště se žádné kontejnery neodvážejí, pouze čas od času přibude nový.

Skladník umisťuje nové kontejnery následovně, aby si usnadnil jejich pozdější vyhledávání podle identifikačních čísel: Předpokládáme, že nově umisťovaný kontejner má identifikační číslo k . Skladník prochází první řadu zleva a hledá první kontejner s identifikačním číslem větším než k .

Pokud tam žádný takový kontejner není, umístí nový kontejner na konec této řady (hned za dosud poslední kontejner). Pokud takový kontejner l najde, pak na jeho místo uloží kontejner k a kontejner l vloží do následující řady stejným způsobem. Takto postupuje tak dlouho, až uloží nějaký kontejner za poslední kontejner v některé řadě nebo na první místo v dosud prázdné řadě.

Kdyby do prázdného skladiště přivezli postupně kontejnery 3, 4, 9, 2, 5, 1, jejich rozmístění ve skladišti by bylo následující:

1 4 5
2 9
3

Jednoho dne přišel ředitel skladiště za svým jediným podřízeným. Zde je část jejich rozhovoru:

Ředitel: Přivezli kontejner 5 dříve než kontejner 4?

Skladník: Ne, pane řediteli, to se nemohlo stát!

Ředitel: Ó, vy dokážete podle rozmístění kontejnerů ve skladišti určit, v jakém pořadí je přivezli?

Skladník: No, ... částečně. Tak například kontejnery, které tu máme teď, mohli přivést v pořadí 3, 2, 1, 4, 9, 5 nebo také 3, 2, 1, 9, 4, 5 nebo čtrnácti dalšími způsoby.

(Ředitel, poněkud zmaten, opouští scénu.)

Vášim úkolem je napsat program, který na základě rozmístění kontejnerů ve skladišti určí všechna možná pořadí, v jakých mohly být do skladiště přidávány.

Vstup. Vstupní soubor se jmenuje `depot.in`. První řádek obsahuje jediné celé číslo R : číslo poslední řady skladiště, ve které je uložen alespoň jeden kontejner. Následujících R řádků souboru popisuje, které kontejnery jsou uloženy v prvních R řadách skladiště. Každý z těchto řádků začíná celým číslem M (počet kontejnerů v příslušné řadě), za ním následuje M dalších celých čísel, což jsou identifikační čísla jednotlivých kontejnerů postupně zleva doprava. Všechna identifikační čísla kontejnerů jsou z rozmezí od 1 do 50 včetně. Celkem není ve skladišti více než 13 kontejnerů.

Výstup. Výstupní soubor se jmenuje `depot.out` a každý jeho řádek popisuje jedno z možných pořadí přivážení kontejnerů do skladiště. Každé pořadí je popsáno N -ticí identifikačních čísel, kde N je celkový počet

kontejnerů přivezených do skladiště. Každé z možných pořadí musí být uvedeno právě jednou.

Příklad 1

depot.in

3

3 1 4 5

2 2 9

1 3

depot.out

3 2 1 4 9 5

3 2 1 9 4 5

3 4 2 1 9 5

3 2 4 1 9 5

3 2 9 1 4 5

3 9 2 1 4 5

3 4 2 9 1 5

3 4 9 2 1 5

3 2 4 9 1 5

3 2 9 4 1 5

3 9 2 4 1 5

3 4 2 9 5 1

3 4 9 2 5 1

3 2 4 9 5 1

3 2 9 4 5 1

3 9 2 4 5 1

Příklad 2

depot.in

2

2 1 2

1 3

depot.out

3 1 2

1 3 2

Hodnocení. Pokud výstupní soubor obsahuje nějaké chybné pořadí nebo je prázdný, za tento test nedostanete žádný bod. V opačném případě je počet bodů za test určen takto: Jestliže výstupní soubor obsahuje všechna možná pořadí a každé z nich právě jednou, získáte 4 body; pokud obsahuje alespoň polovinu z možných pořadí a každé právě jednou, získáte 2 body; jestliže obsahuje méně než polovinu možných pořadí nebo se nějaké pořadí opakuje, obdržíte 1 bod.

6. Hra

Hru hrají dva hráči na hracím plánu tvořeném soustavou N políček (očíslovaných od 1 do N), mezi nimiž vedou šipky. Každé políčko patří jednomu z hráčů, kterému budeme říkat vlastník tohoto políčka. Navíc je každému políčku přiřazeno jedno kladné celé číslo (hodnota políčka),

příčemž hodnoty všech políček jsou navzájem různé. Políčko číslo 1 je určeno jako startovní.

Hra se hraje takto: Na začátku mají oba hráči skóre 0 a na startovním políčku leží kamínek. Hráči kamínek v průběhu hry přemísťují mezi políčky; to políčko, na kterém kamínek právě leží, budeme značit c . Každý tah provádí vlastník políčka c . Tah sestává z následujících operací:

- ▷ Pokud je hodnota políčka c větší než současné skóre vlastníka políčka c , stává se hodnota políčka c novým skóre vlastníka. Pokud není větší, žádné skóre se nemění.
- ▷ Poté vlastník políčka c vybere jednu z šipek vycházejících z tohoto políčka a kamínek přesune na políčko, kam tato šipka vede. (Jestliže cílové políčko tahu patří témuž hráči, bude příští tah provádět opět on.)

Jakmile se kamínek vrátí na startovní políčko, hra končí a vítězí ten hráč, jehož skóre je vyšší.

O hracím plánu je známo, že:

- ▷ Z každého políčka vede vždy alespoň jedna šipka.
- ▷ Každé políčko p je dosažitelné ze startovního políčka, tzn. existuje posloupnost na sebe navazujících šipek, která vede ze startovního políčka do p .
- ▷ Po konečném počtu tahů se při jakékoliv hře dostane kamínek opět na startovní políčko (tzn. hra vždy skončí).

Napište program, který bude hru hrát a pokaždé vyhraje. Soupeř použitý při vyhodnocování hraje optimálně, tzn. když může, vyhraje.

Vstup a výstup. Program čte vstupní data ze standardního vstupu a zapisuje výsledky do standardního výstupu. Váš program je hráč 1, soupeř je hráč 2. Nejprve musí váš program načíst popis hracího plánu. První řádek tohoto popisu obsahuje jediné celé číslo: počet políček N ($1 \leq N \leq 1000$). Každý z následujících N řádků obsahuje N celých čísel, j -té číslo na i -tém z těchto řádků je 1, jestliže z políčka i vede šipka na políčko j , v opačném případě je zde uvedena 0. Další řádek vstupu obsahuje N celých čísel — určení vlastníka jednotlivých políček (každé z nich je 1 nebo 2). Následuje opět řádek s N celými čísly — hodnoty jednotlivých políček, všechny v rozsahu 1 až N a všechny navzájem různé.

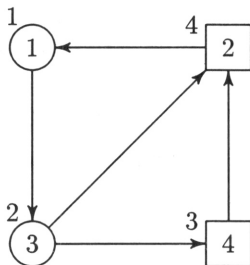
Váš program po načtení popisu hracího plánu hraje následovně:

- ▷ Je-li váš program na tahu, vypíše číslo políčka, na které se jeho tahem přesune kamínek.

- ▷ Je-li na tahu soupeř, váš program načte číslo políčka, na které se kamínek přesunul po soupeřově tahu.

Při testování budou voleny vždy takové hrací plány, na nichž může hráč 1 vyhrát (ať už hru začíná či nikoliv).

Příklad. Hrací plán je vyobrazen na obr. 35. Políčka označená kroužkem patří hráči 1, políčka označená čtverečkem hráči 2. Hodnoty políček jsou zapsány v kroužcích a čtverečcích, čísla políček jsou uvedena u jejich vnějšího okraje.



Obr. 35 ~

Hra může probíhat například takto:

stdin	stdout	vysvětlení
4		<i>N</i>
0 1 0 0		Šipky z políčka 1
0 0 1 1		Šipky z políčka 2
0 0 0 1		Šipky z políčka 3
1 0 0 0		Šipky z políčka 4
1 1 2 2		Vlastníci políček
1 3 4 2		Hodnoty políček
	2	Tah hráče 1
	4	Tah hráče 1
1		Tah hráče 2 na startovní políčko — konec hry.

Po skončení této hry má hráč 1 skóre 3 a hráč 2 skóre 2. Hráč 1 tedy zvítězil.

Návod k programování vstupu a výstupu. Používáte-li ke komunikaci standardní prostředky jazyka, postupujte podle následujících příkladů. V těchto příkladech integerová proměnná `target` obsahuje číslo políčka s kamínkem.

Programujete-li v C++ a používáte `iostreams`, čtete vstup a zapisujete výstup následovně:

```
cin>>target;  
cout<<target<<endl<<flush;
```

Programujete-li v C nebo C++ a používáte `scanf` a `printf`, postupujte takto:

```
scanf("%d", &target);  
printf("%d\n", target); fflush(stdout);
```

Programujete-li v Pascalu, použijte:

```
Readln(target);  
Writeln(target);
```

Pomůcka. Máte k dispozici program (pod Linuxem `score2`, pod Windows `score2.exe`), který si ze souboru `score.in` přečte popis hracího plánu (ve stejném formátu jako má mít vstup vašeho programu) a poté hraje hru jako hráč 2, přičemž jeho výstup odpovídá přesně vstupu vašeho programu a naopak. Své tahy tento program volí náhodně.

Hodnocení. Pro každou z testovaných her získáte plný počet bodů, pokud váš program vyhraje, jinak žádné body. Při vyhodnocování každé hry váš program nejprve hraje s testovacím programem, časový limit je přitom o 1 sekundu vyšší, než je uvedeno v zadání. Průběh hry je zaznamenán. Poté je váš program spuštěn znovu, vstup dostane přeměřovaný ze souboru a je změřena skutečná doba jeho běhu (časový limit podle zadání), přičemž výstup programu musí přesně odpovídat výstupu při prvním testování.

doc. RNDr. Leo Boček, CSc., RNDr. Karel Horák, CSc.,
RNDr. Tomáš Pitner, Dr., doc. RNDr. Jaromír Šimša, CSc.,
RNDr. Jaroslav Švrček, CSc., doc. RNDr. Pavel Töpfer, CSc.

**PADESÁTÝ ROČNÍK
MATEMATICKÉ OLYMPIÁDY
NA STŘEDNÍCH ŠKOLÁCH**

Vydala Jednota českých matematiků a fyziků,
Žitná 25, 117 10 Praha 1
jako svou publikaci č. 57-551-04
Sazbu programem $\text{\TeX}$ připravil Karel Horák
Vytisklo Polygrafické středisko
Univerzity Palackého v Olomouci
1. vydání
Praha 2001

ISBN 80-7015-828-X

**matematická
olympiáda**

MO

ISBN 80-7015-828-X