16. Mezinárodní olympiáda v informatice

In: Karel Horák (editor); Jan Kára (editor); Jaromír Šimša (editor); Jaroslav Švrček (editor); Pavel Töpfer (editor); Jaroslav Zhouf (editor): 53. ročník matematické olympiády na středních školách. Zpráva o řešení úloh ze soutěže konané ve školním roce 2003/2004. 45. mezinárodní matematická olympiáda. 16. mezinárodní olympiáda v informatice. (Czech). Praha: Jednota českých matematiků a fyziků, 2006. pp. 193–203.

Persistent URL: http://dml.cz/dmlcz/405081

16. mezinárodní olympiáda v informatice

Ve dnech 11.–18. 9. 2004 se konala v Athénách v Řecku 16. mezinárodní olympiáda v informatice IOI 2004 (International Olympiad in Informatics). Soutěže se zúčastnilo 300 studentů ze 77 zemí celého světa. Česká národní delegace byla sestavena na základě výsledků celostátního kola 53. ročníku Matematické olympiády v kategorii P (programování). Naše družstvo mělo následující složení:

Tomáš Gavenčiak, absolvent gymnázia M. Koperníka v Bílovci,
Daniel Marek, student gymnázia Ch. Dopplera v Praze 5,
Petr Škoda, absolvent gymnázia v Praze 8, Ústavní,
Martin Vejnár, student gymnázia na tř. Kpt. Jaroše v Brně.

Vedoucími české delegace byli doc. RNDr. *Pavel Töpfer*, CSc. a Mgr. *Martin Mareš*, oba z Univerzity Karlovy v Praze, Matematicko-fyzikální fakulty.

Zvláštností letošního ročníku Mezinárodní olympiády v informatice byla její přímá návaznost na sportovní letní olympijské hry. Soutěž se konala jen asi dva týdny po skončení olympijských her v objektu nově vybudovaném jako dočasné sídlo sportovních novinářů v těsném sousedství hlavního olympijského stadionu. Ihned po skončení IOI v Athénách pro změnu začínala paralympiáda tělesně postižených sportovců.

Mezinárodní olympiáda v informatice byla organizátory výborně zajištěna. Po stránce odborné připravili pořadatelé soutěže kvalitní úlohy přiměřené obtížnosti, z hlediska celkového pobytu pak vhodným způsobem doplnili odborný program dvěma zajímavými výlety pro všechny účastníky — mezi oběma soutěžními dny se konal půldenní výlet na Akropolis, po skončení soutěže pak celodenní plavba lodí mezi řeckými ostrovy.

Vlastní soutěž probíhala jako vždy ve dvou soutěžních dnech. V každém z nich dostali studenti zadány vždy tři soutěžní úlohy a na jejich vyřešení 5 hodin času. Po tuto dobu mohli pracovat na přiděleném osobním počítači vybaveném základním systémovým prostředím a překladači

programovacích jazyků Pascal, C a C++. Cílem každé úlohy je zvládnout zadaný problém algoritmicky a navržený algoritmus pak také naprogramovat. Vytvořené programy se testují pomocí předem připravené sady vstupních dat, takže práce musí být skutečně dovedena až do podoby spolehlivě odladěného programu. Všechny testy jsou navíc vázány na předem známé časové limity. Tím je zajištěno, že program založený na méně efektivním algoritmu stihne včas doběhnout jen pro malá vstupní data, zatímco při výpočtu s velkými daty je běh programu předčasně přerušen, aniž by se program dobral k výsledku a jeho autor obdržel za příslušný test body. Jednotlivá testovací data (zpravidla se používá 20 sad dat hodnocených po 5 bodech) se liší ve své velikosti i složitosti, takže teoreticky správný, ale pomalý program získá ve výsledném hodnocení jen část z celkového dosažitelného množství bodů.

Za každou z šesti soutěžních úloh bylo možné získat nejvýše 100 bodů, celkově tedy až 600 bodů. Skutečné bodové zisky většiny účastníků byly ale samozřejmě výrazně nižší. Podle počtu dosažených bodů bylo stanoveno výsledné pořadí. Lepší polovina účastníků olympiády byla oceněna medailemi, přičemž zlaté, stříbrné a bronzové medaile se rozdělují přibližně v poměru 1 : 2 : 3. Celkem bylo letos uděleno 26 zlatých, 49 stříbrných a 71 bronzových medailí.

Naši studenti se podobně jako v minulých letech umístili mezi úspěšnějšími týmy, i když dosažené výsledky jsou letos o něco horší než loni:

67.–72.	Petr Škoda	370 bodů	stříbrná medaile
73.–75.	Daniel Marek	365 bodů	stříbrná medaile
	Martin Vejnár	220 bodů	–
	Tomáš Gavenčiak	145 bodů	–

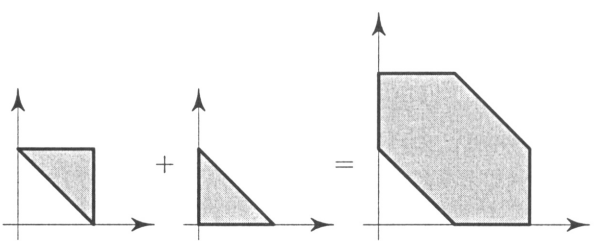
Mezinárodní olympiáda v informatice je soutěží jednotlivců a žádné oficiální pořadí zúčastněných zemí se v ní nevyhlašuje. Podle dosažených výsledků bychom se však řadili přibližně na 25.–30. místo v celkovém pořadí zemí. Nejúspěšnějšími zeměmi byly letos Čína a Rusko se čtyřmi zlatými medailemi, další v pořadí jsou USA, Polsko, Írán, Rumunsko, Bulharsko, Slovensko, Lotyšsko a Tchaj-wan.

Příští, v pořadí již 17. mezinárodní olympiáda v informatice IOI 2005 se uskuteční v polském městě Nowy Sacz nedaleko Krakowa ve dnech 18.–25. 8. 2005.

1. Polygon

Polygon, čili též mnohoúhelník jistě všichni znáte. Ujasněme si jen, že za jeho body považujeme jak body ležící na hranici, tak body uvnitř. Polygon je konvexní právě tehdy, když pro libovolné dva jeho body X a Y platí, že všechny body úsečky XY jsou součástí polygonu. Každý polygon v této úloze bude konvexní, bude mít alespoň dva vrcholy, přičemž všechny jeho vrcholy budou navzájem různé a budou mít celočíselné souřadnice. Žádné tři vrcholy polygonu neleží na jedné přímce.

Minkowského součet polygonů A a B se skládá ze všech bodů ve tvaru $(x_1 + x_2, y_1 + y_2)$, kde (x_1, y_1) je bod polygonu A a (x_2, y_2) je bod polygonu B. Laskavý, ale trpělivý čtenář si může snadno dokázat, že Minkowského součet dvou polygonů je opět polygon. Následující obrázek ukazuje příklad dvou trojúhelníků a jejich součtu:



My se zaměříme na inverzní operaci k Minkowského součtu. Pro zadaný polygon P budeme hledat dva polygony A a B takové, že:

- P je Minkowského součtem polygonů A a B,
- A má 2 až 4 vrcholy, čili je to buďto úsečka (2 vrcholy), trojúhelník (3 vrcholy), nebo čtyřúhelník (4 vrcholy),
- A má největší možný počet vrcholů, čili:
  - A musí být čtyřúhelník, pokud to je možné;
  - jinak to musí být trojúhelník;
  - když není možné ani to, musí to být úsečka.

Je zřejmé, že ani A, ani B nemůže být roven P, neboť druhý sčítanec by musel být bod, což není polygon.

Dostanete několik vstupních souborů, z nichž každý obsahuje popis jednoho polygonu P. Pro každý vstupní soubor nalezněte polygony A a B podle pravidel uvedených výše a vytvořte výstupní soubor s popisem

polygonů A a B. Pro každý vstupní soubor bude řešení existovat. Pokud existuje více řešení, popište libovolné jedno z nich.

Neodevzdávejte žádný program, pouze výstupní soubory.

Vstup: Dostanete 10 vstupních souborů pojmenovaných postupně `polygon1.in` až `polygon10.in`, přičemž číslo na konci jména souboru je číslo vstupu. Každý vstupní soubor vypadá následovně. První řádek obsahuje jediné celé číslo N: počet vrcholů zadaného polygonu. Následujících N řádků popisuje jednotlivé vrcholy polygonu v pořadí proti směru hodinových ručiček. $(I+1)$-ní řádek (pro $1 \leqq I \leqq N$) se skládá ze dvou nezáporných celých čísel X_I a Y_I oddělených jednou mezerou, která udávají x-ovou a y-ovou souřadnici I-tého vrcholu polygonu.

Výstup: Odevzdejte 10 výstupních souborů odpovídajících jednotlivým vstupním souborům. Každý z těchto výstupních souborů bude obsahovat popis polygonů A a B pro příslušný vstupní polygon. Tento popis musí začínat řádkem ve tvaru:

`#FILE polygon` I

kde celé číslo I $(1 \leqq I \leqq 10)$ je pořadové číslo vstupního souboru.

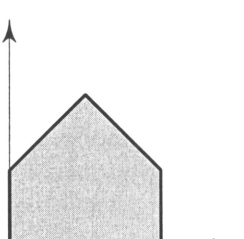
Zbytek výstupního souboru je v podobném tvaru, jako soubor vstupní. Druhý řádek obsahuje jedno celé číslo N_A: počet vrcholů polygonu A $(2 \leqq N_A \leqq 4)$. Následuje N_A řádků popisujících jednotlivé vrcholy polygonu A v pořadí proti směru hodinových ručiček. $(I+2)$-hý řádek (pro $1 \leqq I \leqq N_A$) obsahuje dvě celá čísla X a Y oddělená jednou mezerou, což jsou souřadnice I-tého vrcholu polygonu A.

(N_A+3)-tí řádek pak obsahuje jediné celé číslo N_B počet vrcholů polygonu B $(2 \leqq N_B)$. Následujících N_B řádků popisuje vrcholy polygonu B v pořadí proti směru hodinových ručiček. (N_A+J+3)-tí řádek (pro $1 \leqq J \leqq N_B$) obsahuje dvě celá čísla X a Y oddělená jednou mezerou, což jsou souřadnice J-tého vrcholu polygonu B.

Příklad vstupu a výstupu:

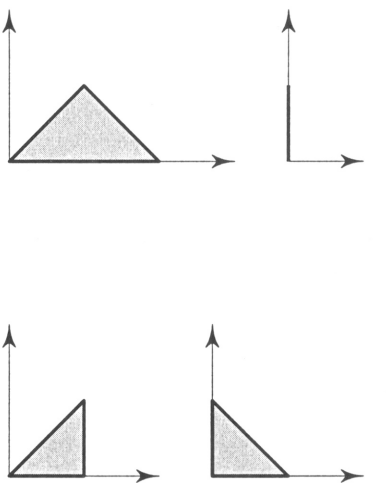
`polygon0.in`

```
5
0 1
0 0
2 0
2 1
1 2
```



Pro tento vstupní soubor jsou správně oba následující výstupní soubory (viz též obrázky). V obou případech je A trojúhelník a nemůže to být čtyřúhelník.

```
#FILE polygon 0
3
0 0
2 0
1 1
2
0 1
0 0
#FILE polygon 0
3
0 0
2 0
1 1
3
0 1
0 0
1 0
```



2. Artemis

Zeus obdaroval bohyni lovu Artemis obdélníkovým územím, aby tam mohla vysadit les. Jelikož Zeus byl tak trochu pedant, levý okraj území byl rovnoběžný s y-ovou souřadnou osou a dolní okraj s x-ovou osou, přičemž levý dolní roh byl v bodě $(0, 0)$. Navíc Artemidě dovolil stromy sázet pouze do bodů s celočíselnými souřadnicemi. Artemis se snažila, aby její les vypadal alespoň trochu přirozeně, a proto stromy sázela tak, aby žádné dva neměly stejnou x-ovou ani y-ovou souřadnici.

Jednoho dne si Zeus usmyslil, že pro něj Artemis musí porazit několik stromů, a to následovně:

1. musí jich být poraženo alespoň T,
2. musí být poraženy právě všechny stromy v nějakém obdélníku,
3. strany obdélníka musí být rovnoběžné se souřadnými osami,
4. v protilehlých vrcholech obdélníka musí růst stromy a ty budou také poraženy.

Artemis má ale stromy moc ráda, takže chce tyto podmínky splnit a přitom porazit co možná nejméně stromů. Vás požádala, abyste napsali

program, který pro zadané rozmístění stromů a minimální počet stromů k poražení T nalezne příslušný obdélník, kde se má porážet.

Vstup: Vstupní soubor se jmenuje `artemis.in`. Jeho první řádek obsahuje jediné celé číslo N: počet stromů v lese. Druhý řádek obsahuje rovněž jediné celé číslo T: minimální počet stromů, které mají být poraženy. Následujících N řádků popisuje polohy jednotlivých stromů: na každém z nich jsou dvě celá čísla X a Y, která představují x-ovou a y-ovou souřadnici příslušného stromu.

Výstup: Výstupní soubor se jmenuje `artemis.out`. Je tvořen jediným řádkem, na němž jsou dvě celá čísla I a J oddělená jednou mezerou. Říkají, že Artemis má kácet stromy v obdélníku, v jehož protilehlých vrcholech jsou stromy s čísly I a J (popsané na řádcích $I+2$ a $J+2$ ve vstupním souboru). Na pořadí stromů I, J ve výstupu nezáleží. Pro každá vstupní data existuje alespoň jedno řešení, pokud jich je více, vypište jedno libovolné z nich.

Příklad vstupu a výstupu:

`artemis.in`	`artemis.out`
`3`	`1 2`
`2`	
`1 1`	
`2 3`	
`5 6`	

Omezení: Pro všechny vstupy platí: $1 < N \leqq 20\,000$, $0 \leqq X, Y \leqq 64\,000$ a $1 < T \leqq N$. Navíc v 50 % vstupů je $1 < N < 5\,000$.

3. Hermes

I řečtí bohové jdou s dobou. Nedávno se z vrcholu Olympu přestěhovali do velkoměsta s pravoúhlou sítí ulic rovnoběžných se souřadnými osami. Ulice mají celočíselné souřadnice a pro každé celé číslo existuje svislá i vodorovná ulice s tímto číslem. Dvojice celých čísel pak určují křižovatky ulic. Horké letní dny bohové tráví v kavárnách umístěných právě na těchto křižovatkách. Posel bohů Hermes dostal za úkol doručit odpočívajícím bohům světelné signály. Může se při tom pohybovat pouze ulicemi města. Každý signál je určen jedinému bohovi, ale nevadí, když ho spatří i někteří z ostatních bohů.

Hermes dostane souřadnice jednotlivých kaváren v pořadí, v jakém do nich mají být signály doručeny. Svou cestu začne v bodě $(0, 0)$. Bohové

signály vidí na libovolnou vzdálenost, ale pouze podél ulic. Má-li tedy Hermes doručit signál bohovi do kavárny na křižovatce (X_i, Y_i), stačí mu dostat se do libovolného bodu na téže vodorovné ulici (s y-ovou souřadnicí Y_i) nebo na téže svislé ulici (s x-ovou souřadnicí X_i). Když odešle poslední signál, jeho mise končí.

Napište program, který dostane zadanou posloupnost souřadnic křižovatek a nalezne délku nejkratší cesty, při níž Hermes doručí všechny signály.

Vstup: Vstupní soubor se jmenuje `hermes.in`. Jeho první řádek obsahuje jediné celé číslo N: počet signálů k doručení. Následujících N řádků obsahuje souřadnice N křižovatek, na které mají být jednotlivé signály doručeny. Pořadí řádků odpovídá pořadí doručování. Na každém z těchto N řádků se nacházejí dvě celá čísla: nejprve x-ová a pak y-ová souřadnice příslušné křižovatky.

Výstup: Výstupní soubor se jmenuje `hermes.out`. Obsahuje jediný řádek s jedním celým číslem: minimální vzdáleností, kterou musí Hermes urazit, aby doručil všechny signály.

Příklad vstupu a výstupu:

```
hermes.in                    hermes.out
5                            11
8 3
7 -7
8 1
-2 1
6 -5
```

Omezení: Ve všech vstupech je $1 \leqq N \leqq 20\,000$, $-1\,000 \leqq X_i, Y_i \leqq$ $\leqq 1\,000$. Mimo to je v 50 % vstupů $1 \leqq N \leqq 80$.

4. Empodia

Starověký matematik a filosof Pythagoras věřil, že pravá podstata tohoto světa je matematická. Současní biologové také občas kráčejí v jeho stopách. Mimo jiné studují vlastnosti *biosekvencí*. To jsou posloupnosti M celých čísel, které:

- ▷ obsahují každé z čísel $0, 1, \ldots, M-1$,
- ▷ začínají nulou a končí číslem $M-1$,
- ▷ neobsahují žádný prvek E bezprostředně následovaný prvkem $E+1$.

Souvislým podposloupnostem biosekvencí se říká *segmenty*. *Ohraničený interval* (dále již jen *interval*) je takový segment, který obsahuje

všechna celá čísla z rozmezí od hodnoty prvního prvku tohoto segmentu do hodnoty posledního prvku segmentu. Navíc první prvek musí být nejmenším prvkem segmentu a poslední prvek největším. První a poslední prvek musí být různé. Interval, který neobsahuje žádný kratší interval, se nazývá *empodio*.

Kupříkladu biosekvence $(0, 3, 5, 4, 6, 2, 1, 7)$ je sama o sobě interval. Jelikož ale obsahuje segment $(3, 5, 4, 6)$, který je také intervalem, není celá tato biosekvence empodio. Interval $(3, 5, 4, 6)$ již empodio je, jelikož žádný kratší interval neobsahuje. Žádná jiná empodia v uvažované biosekvenci nejsou.

Napište program, který pro danou biosekvenci vypíše všechna empodia v ní obsažená.

Vstup: Vstupní soubor se jmenuje `empodia.in`. Jeho první řádek obsahuje jediné celé číslo M: počet čísel ve vstupní biosekvenci. Následujících M řádků obsahuje jednotlivé prvky biosekvence, což jsou celá čísla v pořadí, jak po sobě v biosekvenci následují.

Výstup: Výstupní soubor se jmenuje `empodia.out`. Na jeho prvním řádku je jediné celé číslo H: počet empodií vyskytujících se ve vstupní biosekvenci. Následuje H řádků popisujících jednotlivá empodia v pořadí určeném tím, kde v biosekvenci začínají. Každý z těchto řádků obsahuje dvě celá čísla A a B (v tomto pořadí) oddělená jednou mezerou, kde A je pořadové číslo prvního prvku empodia v biosekvenci a B pořadové číslo posledního prvku empodia.

Příklad vstupu a výstupu:

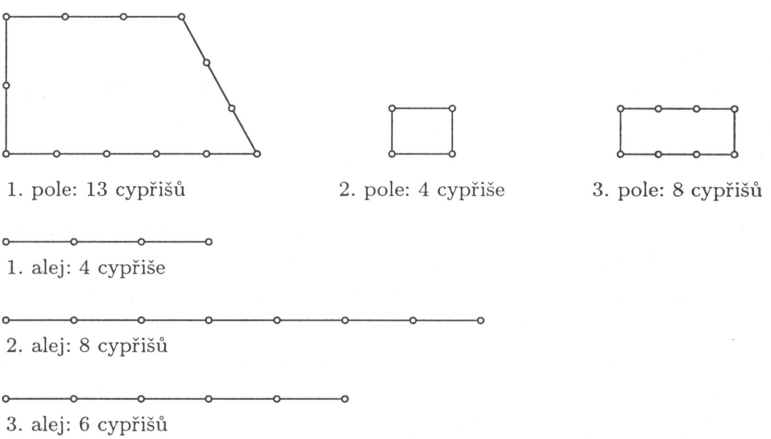
`empodia.in`	`empodia.out`
8	1
0	2 5
3	
5	
4	
6	
2	
1	
7	

Omezení: Ve všech vstupech mimo jednoho je $1 \leqq M \leqq 60\,000$. V jednom vstupu je $1\,000\,000 \leqq M \leqq 1\,100\,000$. Mimo to, v 50 % vstupů je $M \leqq 2\,600$.

5. Sedlák

Byl jednou jeden sedlák. Měl několikero polí lemovaných cypřišovými stromy (inu, řecký sedlák). Na každém ze svých dalších pozemků vysázel ještě jednu samostatnou alej, tvořenou jednou řadou cypřišů. Jak kolem polí, tak v alejích ovšem mezi každými dvěma sousedními cypřiši stojí ještě jeden olivovník. Jiné cypřiše ani olivovníky už naštěstí nemá.

Roky plynuly a život starého sedláka se zvolna chýlil ke konci. Jednoho dne si zavolal svého nejstaršího syna a pravil: „Dám ti libovolných Q cypřišů, které si vybereš, a s nimi dostaneš i každý olivovník, který sousedí se dvěma tebou vybranými cypřiši.“ Z každého pole a každé aleje si syn může vybrat libovolnou kombinaci cypřišů. Miluje ovšem olivy, a proto si chce vybrat takových Q cypřišů, aby s nimi získal co nejvíce olivovníků.



Obr. 63. Příklad rozmístění cypřišů. Olivovníky nejsou zobrazeny.

V příkladě vyobrazeném na obr. 63 si má syn vybrat $Q = 17$ cypřišů. Aby získal co nejvíce olivovníků, vybere si všechny cypřiše na 1. a 2. poli a dostane s nimi 17 olivovníků.

Napište program, který na základě čísla Q a údajů o polích a alejích spočte maximální možný počet olivovníků, které může syn získat.

Vstup: Vstupní soubor se jmenuje `farmer.in`. Jeho první řádek obsahuje tři celá čísla Q, M a K. Číslo Q udává počet cypřišů, které si syn má vybrat, M určuje počet sedlákových polí a K počet alejí. Na druhém řádku je M celých čísel $N_1, N_2, \ldots, N_M$, kde N_I určuje počet cypřišů na I-tém poli. Třetí řádek obsahuje K celých čísel $R_1, R_2, \ldots, R_K$, přičemž R_J udává počet cypřišů v J-té aleji.

Výstup: Výstupní soubor se jmenuje `farmer.out`. Obsahuje jediný řádek s jediným celým číslem, které udává maximální počet olivovníků, jež může syn získat.

Příklad vstupu a výstupu:

farmer.in	farmer.out
17 3 3	17
13 4 8	
4 8 6	

Omezení: Ve všech vstupech je $0 \leqq Q \leqq 150\,000$, $0 \leqq M \leqq 2\,000$, $0 \leqq K \leqq 2\,000$, $3 \leqq N_1 \leqq 150$, $3 \leqq N_2 \leqq 150$, ..., $3 \leqq N_M \leqq 150$, $2 \leqq R_1 \leqq 150$, $2 \leqq R_2 \leqq 150$, ..., $2 \leqq R_K \leqq 150$. Celkový počet všech cypřišů je alespoň Q. Mimo to, v 50 % vstupů je $Q \leqq 1\,500$.

6. Phidias

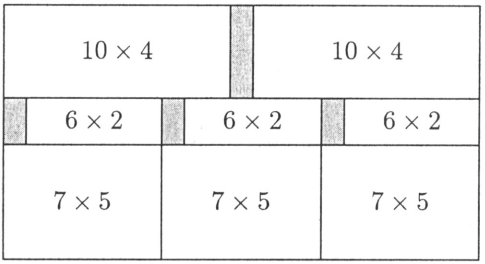
Slavný řecký sochař Phidias (u nás od nepaměti známý spíše pod jménem Feidias) se chystá stvořit další z divů světa. Proto potřebuje obdélníkové desky z mramoru o velikostech $W_1 \times H_1$, $W_2 \times H_2$, ..., $W_N \times$ $\times H_N$.

Před pár dny Phidias připadl na velký obdélníkový blok mramoru. Rozhodl se ho rozřezat, aby získal desky požadovaných velikostí. Jakýkoli kus mramoru (ať už to je původní blok nebo části z něj nařezané) může přeříznout svisle nebo vodorovně na dvě obdélníkové části s celočíselnou šířkou i výškou, přičemž řez musí vždy být rovný a vést od jednoho okraje k druhému. Nijak jinak řezat nelze a ani není možné nařezané kusy slepovat k sobě. A jelikož povrch mramoru má kresbu, nelze kusy ani otáčet: pokud Phidias uřízne část o rozměrech $A \times B$, nemůže ji použít jako desku velikosti $B \times A$, leda že by bylo $A = B$. Od každé požadované velikosti může vyrobit libovolný, třeba i nulový počet desek.

Odpadem jsou ty desky, které po provedení všech řezů nemají žádnou z požadovaných velikostí. Phidias by rád věděl, jak má svůj blok rozřezat, aby se do odpadu dostalo co nejméně mramoru.

Například na níže uvedeném obrázku máme blok o šířce 21 a výšce 11. Potřebujeme desky o velikostech 10×4, 6×2, 7×5 a 15×10. Tehdy je nejmenší možná plocha odpadu 10 a obr. 64 ukazuje jedno možné rozřezání s tímto množstvím odpadu.

Vaším úkolem je napsat program, který pro zadanou velikost původního bloku a požadované velikosti desek spočte nejmenší možnou plochu odpadu.



Obr. 64

Vstup: Vstupní soubor se jmenuje `phidias.in`. Jeho první řádek obsahuje dvě celá čísla: nejprve W, šířku původního bloku, a následně H, jeho výšku. Druhý řádek obsahuje jediné celé číslo N: počet požadovaných velikostí desek. Na následujících N řádcích jsou popsány jednotlivé požadované velikosti desek. Na každém z těchto řádků jsou dvě celá čísla udávající příslušnou velikost: šířka W_i následovaná výškou H_i $(1 \leqq i \leqq N)$.

Výstup: Výstupní soubor se jmenuje `phidias.out`. Obsahuje jediné celé číslo: minimální možnou plochu odpadu.

Příklad vstupu a výstupu:

`phidias.in`	`phidias.out`

`21 11`
`4`
`10 4`
`6 2`
`7 5`
`15 10`
	`10`

Omezení: Ve všech vstupech je $1 \leqq W \leqq 600$, $1 \leqq H \leqq 600$, $0 < N \leqq 200$, $1 \leqq W_i \leqq W$ a $1 \leqq H_i \leqq H$. Navíc 50 % vstupů má $W \leqq 20$, $H \leqq 20$ a $N \leqq 5$.