73. ročník Matematické olympiády – 2023/2024

Úlohy ústředního kola kategorie P – 1. soutěžní den

Na řešení úloh máte 4.5 hodiny čistého času. Řešení každé úlohy pište na samostatný list papíru. Při soutěži je zakázáno používat jakékoliv pomůcky kromě psacích potřeb (tzn. knihy, kalkulačky, mobily, apod.).

Řešení každé úlohy má obsahovat:

- *Popis řešení,* to znamená slovní popis principu zvoleného algoritmu, *argumenty zdůvodňující jeho správnost* (případně důkaz správnosti algoritmu), *diskusi o efektivitě* vašeho řešení (časová a paměťová složitost; to se netýká úlohy P-III-3). Slovní popis řešení musí být jasný a srozumitelný i bez nahlédnutí do samotného zápisu algoritmu (do programu). Není možné odkazovat se na vaše řešení úloh z předchozích kol, opravovatelé je nemusí mít k dispozici.
- Doporučujeme uvést *zápis algoritmu* v nějakém dostatečně srozumitelném pseudokódu (případně v programovacím jazyce C/C++, Python nebo podobném). Nemusíte detailně popisovat jednoduché operace jako vstupy, výstupy, implementaci jednoduchých matematických vztahů, vyhledávání v poli, třídění apod. Zápis algoritmu sice není povinnou součástí řešení, ale při nejasností v popisu řešení může opravovatelům pomoci s pochopením algoritmu.

Za každou úlohu můžete získat maximálně 10 bodů. Hodnotí se nejen správnost řešení, ale také kvalita jeho popisu a efektivita zvoleného algoritmu. Algoritmy posuzujeme podle jejich časové složitosti, tzn. závislosti doby výpočtu na velikosti vstupních dat. Záleží přitom pouze na řádové rychlosti růstu této funkce. V zadání každé úlohy najdete přibližné limity na velikost vstupních dat. Efektivním vyřešením úlohy rozumíme to, že váš program spuštěný s takovými daty na současném běžném počítači dokončí výpočet během několika sekund.

P-III-1 Suši bufet

Po pohyblivém pásu v suši bufetu dnes postupně prošlo n talířků. Na i-tém talířku byl kousek suši, který měl kvalitu q_i a vážil v_i gramů.

Víme, že někdy během dne přišla do bufetu Kika, sedla si k pásu a nějaký čas jedla všechny kousky suši, které šly kolem. Nakonec usoudila, že už toho snědla dost, a tak z bufetu odešla. Samozřejmě když sníme deset různých kousků suši a jeden z nich je špatný, pokazí to dojem z celého oběda. Proto platí, že kvalita jejího oběda je rovna *nejmenší* z kvalit suši, které snědla.

Soutěžní úloha

Víme, že Kika se v bufetu zvládla docela dost najíst. Přesněji řečeno, víme, že dohromady snědla alespoň z gramů suši. Zjistěte, jakou největší kvalitu $q_{\max}$ mohl mít Kičin oběd. Následně zjistěte, kolik nejvíce gramů $v_{\max}$ suši mohla dohromady sníst, pokud její oběd opravdu měl kvalitu $q_{\max}$.

Formát vstupu

Na prvním řádku jsou kladná celá čísla n a z. Na druhém řádku jsou kladná celá čísla $q_1, \ldots, q_n$, kvality jednotlivých kousků suši v pořadí, ve kterém šly po pásu. Na třetím řádku jsou kladná celá čísla $v_1, \ldots, v_n$, váhy jednotlivých kousků suši ve stejném pořadí.

Formát výstupu

Na výstup vypište dva řádky, na prvním číslo $q_{\max}$, na druhém číslo $v_{\max}$.

Příklady

Vstup:	*Výstup:*

`6 140`
`400 300 100 400 1000 10`
`50 51 52 53 54 20`
	`100`
	`260`

Kika určitě snědla třetí kousek suši, její kvalita oběda tedy určitě byla nanejvýš 100. *Největší možný oběd s touto kvalitou vypadá tak, že snědla prvních pět kousků suši.*

Vstup:	*Výstup:*

`6 107`
`400 300 100 400 1000 10`
`50 51 52 53 54 20`
	`400`
	`107`

Kika snědla čtvrtý a pátý kousek suši. Tento oběd má kvalitu $\min(400, 1000) = 400$ *a celkovou hmotnost přesně* $53 + 54 = 107$ *gramů.*

Bodování

Můžete předpokládat, že celková váha všech kousků suši i jejich největší kvalita se vejdou do běžné celočíselné proměnné ve vámi zvoleném programovacím jazyce. Tato čísla ale mohou být řádově větší než n, proto například nepředpokládejte, že je můžete setřídit v čase $\mathcal{O}(n)$.

Plný počet bodů získají správná řešení efektivní pro $n \leq 500\,000$. Řešení efektivní pro $n \leq 5\,000$ i v nejhorším případě získají 5 bodů.

P-III-2 Pexeso

Na dlouhém pásku papíru je za sebou nakresleno n stejně velkých čtvercových obrázků. Obrázky na pásku se mohou libovolně opakovat. Kubík si chce vystřihnout kousek tohoto pásku, tj. souvislý úsek tvořený několika po sobě jdoucími obrázky, a vyrobit si z něj pexeso. Jak určitě víte, v pexesu chceme mít každý obrázek přesně dvakrát.

Kubík se rozhodl, že do svého pexesa dá právě ty obrázky, které budou na jeho kousku *přesně* dvakrát (teoreticky by uměl použít i ty obrázky, které má na svém kousku vícekrát, ale to by musel od každého zahodit správný počet nadbytečných kopií a přijde mu, že s tím by bylo moc práce).

Soutěžní úloha

Na vstupu je dán popis pásku jako posloupnost čísel $a_0, \ldots, a_{n-1}$, přičemž stejné číslo představuje stejný obrázek. Napište program, který vypočítá, kolik nejvíc dvojic může mít Kubíkovo pexeso.

Formát vstupu

Na prvním řádku vstupu je kladné celé číslo n, na druhém jsou kladná celá čísla $a_0, \ldots, a_{n-1}$.

Formát výstupu

Na výstup vypište jedno celé číslo: největší možný počet obrázků, které se vyskytují právě dvakrát na souvislém úseku vstupního pásku papíru.

Příklady

Vstup:	*Výstup:*
`14` `7 2 2 1 7 7 999 7 999 7 2 1 1 7`	`3`

Pokud si Kubík vystřihne kousek „2 1 7 7 999 7 999 7 2 1“, může si dát do pexesa dvojice s obrázky 1, 2 a 999. Jiným optimálním řešením je vystřihnout kousek „999 7 999 7 2 1 1“ a do pexesa vzít obrázky 999, 7 a 1. Žádný kousek tohoto papíru neobsahuje více než tři dvojice.

Vstup:	*Výstup:*
`9` `10 20 30 40 50 40 30 20 10`	`4`

Zde je nejlepší možnost vzít úplně celý pásek papíru.

Vstup:	*Výstup:*
`9` `10 10 10 20 20 20 30 30 30`	`2`

Zde je nejlepší vystřihnout úsek „10 10 20 20“, „20 20 30 30“, nebo „10 10 20 20 20 30 30“.

Bodování

O velikostech čísel a_i nepředpokládejte nic kromě toho, že se vejdou do běžných celočíselných proměnných ve vámi zvoleném programovacím jazyce.

Správná řešení efektivní pro $n \leq 10^6$ získají 8–10 bodů v závislosti na přesné časové složitosti. Řešení efektivní pro $n \leq 5\,000$ získají 4–5 bodů v závislosti na přesné časové složitosti.

P-III-3 O Vekslákbotovi a Pokladničce

K této úloze se vztahuje studijní text uvedený na této a následujících stranách, který je totožný se studijním textem ze zadání domácího a krajského kola.

Jednotlivé podúlohy jsou hodnoceny nezávisle, můžete je řešit v libovolném pořadí. Při hodnocení budeme přihlížet jen ke správnosti vašich programů – na jejich efektivitě (časové složitosti) nám nebude záležet. Nezapomeňte kromě samotného programu uvést i jeho slovní popis, včetně zdůvodnění správnosti.

Podúloha A (2 body): nadpoloviční většina

V Pokladničce je na začátku $c > 0$ červených, $m > 0$ modrých a $z > 0$ zelených žetonů (a nic jiného). Je zaručeno, že *některá barva má nadpoloviční většinu*, tedy jejích žetonů je ostře více než polovina všech. Napište program, po jehož skončení bude v Pokladničce přesně jeden žeton, a to té barvy, která měla původně nadpoloviční většinu.

Podúloha B (4 body): logaritmus

Funkce $g(x) = \lceil \log_2 x \rceil$ počítá horní celou část dvojkového logaritmu čísla x. Slovně: $g(x)$ je nejmenší celé číslo y takové, že $2^y \geq x$. Například platí $g(31) = 5$, $g(32) = 5$ (neboť 2^5 je přesně 32) a $g(33) = 6$.

Na začátku je v Pokladničce $c > 0$ červených žetonů a nic jiného. Napište program, po jehož skončení bude v Pokladničce přesně $g(c)$ modrých žetonů. Můžete libovolně používat omezení. Po skončení programu mohou v Pokladničce být i žetony jiných barev.

Podúloha C (4 body): Fibonacci

Fibonacciho čísla jsou definována následovně: $F_0 = 0$, $F_1 = 1$ a pro každé celé číslo $n \geq 2$ platí $F_n = F_{n-1} + F_{n-2}$. Tato posloupnost tedy začíná následovně: $0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$

V Pokladničce je na začátku $c > 0$ červených žetonů a nic jiného. Napište program *bez použití omezení,* po jehož skončení bude v Pokladničce přesně F_c modrých žetonů a nic jiného.

Částečné body můžete dostat za korektní řešení, která použijí omezení a/nebo budou mít na konci v Pokladničce i jiné než modré žetony.

Studijní text

Za devatero horami a devatero řekami byla jednou jedna prazvláštní země. V této zemi žili roboti a místo peněz používali žetony všech možných barev. V domečku na úpatí desáté hory spolu žili dva hrdinové našeho příběhu: roboti Vekslákbot a Pokladnička.

Jak asi tušíte z jejího jména, Pokladnička v sobě ráda ukládá všechny možné žetony. Jak možná z jejího jména netušíte, Pokladnička také velmi ráda vykonává různé programy. A jak jste si po přečtení předchozí věty naprosto jistí (ostatně, toto je studijní text Matematické Olympiády kat. P a ne jen tak nějaká pohádka), právě tyto programy pro ni budete psát vy jako řešení soutěžních úloh.

Vekslákbot je mistr vyměňování žetonů. Ať už chce vyměnit dva červené za tři modré, anebo žlutý a černý za $10^9 + 7$ šedých, Vekslákbot určitě zná robota, který zná robota, jenž s ním právě takovou směnu rád provede. Vekslákbot má moc rád Pokladničku, takže pokud ho ona o nějakou výměnu poprosí, okamžitě (nebo, jak říkáme my, v konstantním čase) ji pro ni provede.

Pojďme se nyní podívat na to, jak budou vypadat programy pro Pokladničku.

Základy: vstup, výstup, program

Vstupem pro Pokladničku budou žetony, jež má na začátku uložené v sobě. Neexistuje žádný výstup. V zadání jednotlivých úloh budeme různě definovat cíle, jichž mají vaše programy dosáhnout.

Program pro Pokladničku se skládá ze dvou částí: *omezení*, která musí dodržet, a *pokynů*, které má vykonávat.

Omezení

První částí programu pro Pokladničku je **konečná množina** omezení (může být i prázdná). Omezení pro Pokladničku určují, kolik nejvýše žetonů konkrétní barvy, případě kombinací žetonů různých barev, může mít najednou v sobě. Formálně, omezení jsou lineární nerovnosti následujícího tvaru:

$$k_1 \cdot \text{barva}_1 + k_2 \cdot \text{barva}_2 + \cdots + k_n \cdot \text{barva}_n \leq \text{limit},$$

kde všechny koeficienty k_i i limit jsou konkrétní kladná celá čísla a barva_i jsou proměnné označující počet žetonů příslušné barvy v Pokladničce. Tečky $(\cdot)$ můžeme vynechat, pokud je zřejmé, kde končí koeficient a kde začíná název barvy. Příklady omezení jsou například nerovnosti modrá ≤ 7 nebo modrá $+ 2$červená ≤ 3.

Pokud se nějaké barvy žádné omezení netýká, může být v Pokladničce libovolné množství žetonů této barvy.

Instrukce

Každá instrukce pro Pokladničku má následující tvar:

$$k_1 \cdot \text{barva}_1, \ldots, k_n \cdot \text{barva}_n \to \ell_1 \cdot \text{barva}_{n+1}, \ldots, \ell_m \cdot \text{barva}_{n+m}$$

Všechno nalevo od $\to$ budeme nazývat levou stranou instrukce, všechno napravo zase pravou stranou. Tečky $(\cdot)$ můžeme vynechat, pokud je zřejmé, kde končí koeficient a kde začíná název barvy.

Všechna k_i i ℓ_j musí být kladná celá čísla. V rámci pravé i levé strany instrukce musí být všechny použité barvy různé. Je povolené použít stejnou barvu na obou stranách instrukce. Je povolené mít $n = 0$ či $m = 0$, tedy instrukci, které má některou stranu prázdnou. (Je povolené mít prázdné i obě strany, ale jak se brzy dozvíme, program, v němž bychom takovou instrukci použili, by nikdy neskončil.)

Příklady instrukcí:

- 2červená $\to$ 3modrá
- 1žlutá, 1černá $\to$ $(10^9 + 7)$šedá
- 3červená $\to$ 333červená, 334cyklámenová, 335purpurová
- 2zelená $\to \emptyset$

Poslední instrukce z příkladu má prázdnou pravou stranu. Pro zápis prázdné strany instrukce budeme používat symbol prázdné množiny, aby bylo jasné, že je prázdná úmyslně.

V našich příkladech budeme používat skutečné názvy barev. Ve svých programech můžete jako názvy barev používat i libovolné jiné alfanumerické řetězce. Je také povoleno v zápisech vynechávat koeficient 1. Druhou z výše uvedených instrukcí lze tedy zapsat také jako žlutá, černá $\rightarrow (10^9 + 7)$šedá.

Pokladnička vykoná instrukci tak, že ze sebe vyndá sadu žetonů na levé straně instrukce, dá ji Vekslákbotovi a poprosí jej, aby jí za ně přinesl sadu žetonů z pravé strany. Vekslákbot to samozřejmě okamžitě zajistí a Pokladnička do sebe vloží žetony, které jí přinesl.

Pokud v sobě Pokladnička nemá všechny žetony, které vyžaduje levá strana instrukce, nelze danou instrukci v tu chvíli provést. Pokud by například měla pouze dva červené žetony, nešlo by provést instrukci 3červená $\rightarrow$ 333červená.

Pokladnička také nemůže provést instrukci, po jejímž provedení by bylo porušené některé omezení.

Program

Program pro Pokladničku tvoří **konečná posloupnost** instrukcí výše uvedeného typu. Zdůrazňujeme, že jde o posloupnost, tedy **záleží** na pořadí instrukcí.

Vykonávání programu

Program se vykonává v krocích. V každém kroku Pokladnička začne číst program od začátku a čte jej, dokud nenajde první instrukci, kterou momentálně může vykonat, a tu vykoná. (Na zbytek programu se v tomto kroku už ani nepodívá a v dalším kroku začne znovu číst program od začátku.)

Vykonávání programu skončí, když už žádnou instrukci v programu nelze vykonat.

Příklad #1: Více červených

Úloha: Na začátku jsou v Pokladničce nějaké červené a nějaké modré žetony. Napište program, po jehož skončení bude v Pokladničce právě jeden zlatý žeton a nic jiného, pokud bylo červených žetonů více než modrých. Ve všech ostatních případech musí Pokladnička skončit úplně prázdná.

Řešení 1 (bez omezení)

Nebudeme mít žádná omezení. Pokyny budou vypadat následovně:

1. červená, modrá $\rightarrow \emptyset$
2. červená, zlatá $\rightarrow$ zlatá
3. červená $\rightarrow$ zlatá
4. modrá $\rightarrow \emptyset$

Při vykonávání tohoto programu bude nejprve Pokladnička používat instrukci 1, dokud to lze. Až to přestane být možné, má v sobě už buď jen červené, anebo jen modré žetony (anebo žádné žetony a program skončí). Pokud jsou modré, jediná

vykonatelná instrukce je instrukce 4, jejímž opakovaným používáním se Pokladnička vyprázdní a program skončí.

Pokud po skončení instrukce 1 zůstanou v Pokladničce červené žetony, bude to o trochu složitější: Jediná instrukce, která se v danou chvíli dá použít, je instrukce 3, která vymění jeden červený žeton za jeden zlatý. Od této chvíle se však instrukce 3 již nepoužije, a to proto, že lze vykonávat instrukci 2, která je v posloupnosti dříve. Pomocí té se Pokladnička postupně zbaví zbývajících červených žetonů. Jakmile v ní zůstane jen samotný zlatý žeton, nelze vykonat žádnou instrukci, a program tedy skončí.

Řešení 1 (s omezením)

Stejnou úlohu můžeme vyřešit s použitím omezení. Vystačíme si s jediným:

- zlatá ≤ 1

Program vypadá následovně:

1. červená, modrá $\rightarrow \emptyset$
2. červená $\rightarrow$ zlatá
3. červená $\rightarrow \emptyset$
4. modrá $\rightarrow \emptyset$

Tentokrát se v situaci, kdy jsou v Pokladničce jen samé červené žetony, nejprve jednou vykoná instrukce 2 (jeden červený žeton vyměníme za jeden zlatý) a potom se už bude používat instrukce 3, dokud červené žetony nedojdou. Omezení, které jsme si zvolili, totiž Pokladničce brání znovu využít instrukci 2.

Příklad #2: Součet

Úloha: Na začátku je v Pokladničce $c > 0$ červených žetonů, $m > 0$ modrých žetonů a jeden zelený. Napište program, po jehož skončení bude v Pokladničce přesně c červených, m modrých a $c + m$ fialových žetonů.

Řešení: Kdybychom pouze chtěli dostat $c + m$ fialových žetonů, stačilo by vyměnit všechny červené i modré žetony za fialové „s kurzem jedna ku jedné“. Jak ale nepřijít o původní žetony?

Naše řešení nebude mít žádná omezení a program bude vypadat následovně:

1. červená, zelená $\rightarrow$ tmavočervená, zelená
2. modrá, zelená $\rightarrow$ tmavomodrá, zelená
3. zelená $\rightarrow$ žlutá
4. tmavočervená, žlutá $\rightarrow$ červená, fialová, žlutá
5. tmavomodrá, žlutá $\rightarrow$ modrá, fialová, žlutá
6. žlutá $\rightarrow \emptyset$

Všimněte si, jak náš program využívá přítomnost zeleného a žlutého žetonu v Pokladničce: Pomocí nich umíme rozlišit, zda ještě měníme původní červené a modré žetony, anebo zda již zpátky vznikají nové.