72. ročník Matematické olympiády – 2022/2023

Úlohy školního kola kategorie P

Řešení úloh odevzdávejte pomocí webového rozhraní, které je dostupné na stránkách olympiády *https://mo.mff.cuni.cz/*. Tam také najdete podrobnější instrukce k odevzdávání a další informace o kategorii P.

Úlohy P-I-1 a P-I-2 jsou praktické. Vaším úkolem v nich je vytvořit a odladit efektivní program v jazyce Pascal, C nebo C++. Řešení těchto dvou úloh odevzdávejte přes webové rozhraní ve formě zdrojového kódu. Odevzdaná řešení budou automaticky vyhodnocena pomocí připravených vstupních dat a výsledky vyhodnocení se dozvíte krátce po odevzdání. Pokud váš program nezíská plný počet 10 bodů, můžete své řešení opravit a znovu odevzdat.

Úlohy P-I-3 a P-I-4 jsou teoretické, za každou z nich lze získat až 10 bodů. Řešení musí obsahovat popis algoritmu, zdůvodnění jeho správnosti a v úloze P-I-3 též odhad časové a paměťové složitosti. Nemusíte psát program, algoritmus stačí zapsat ve vhodném pseudokódu nebo dokonce jenom slovně, ale v tom případě dostatečně podrobně a srozumitelně. Hodnotí se nejen správnost, ale také efektivita zvoleného postupu řešení. Řešení obou teoretických úloh odevzdávejte ve formě souboru typu PDF přes výše uvedené webové rozhraní.

Řešení všech úloh můžete odevzdávat do 15. listopadu 2022. Opravená řešení a seznam postupujících do krajského kola najdete na webových stránkách olympiády.

Experimentální jazyky: V letošním ročníku MO-P je také možné odevzdávat praktické úlohy i v jazycích Python 3 a Java 11. U nich nicméně nezaručujeme, že bude možné získat plný počet bodů – je možné, že program nestihne doběhnout do časového limitu, byť používá algoritmus s optimální časovou složitostí.

P-I-1 Tmavá zákoutí

Po dlouhé práci s přípravou úloh na výběrové soustředění se organizátoři vrací pozdě v noci zpět z Matfyzu na kolej. Jelikož si nesou igelitky plné drahé elektroniky, nechtějí vstupovat do tmavých zákoutí a jsou ochotni jít pouze po dobře osvětlených místech. Bohužel není zaručeno, že se tímto způsobem na kolej z Matfyzu dostat lze, naštěstí se však organizátoři dokázali nabourat do systémů dispečinku Managementu Osvětlení Prahy (MOP), kde dokáží lampy specifickým způsobem zapínat. Aby minimalizovali riziko, že budou objeveni, rádi by toho však využili co nejméněkrát. Pomůžete jim?

Soutěžní úloha

Prahu si můžete představit jako čtvercovou mřížku o rozměrech $n \times n$, Matfyz leží v políčku vlevo nahoře o souřadnicích $(1, 1)$, kolej leží v políčku vpravo dole o souřadnicích (n, n). Jsou-li organizátoři v políčku (x, y), umí se přesunout do políček sousedících s ním přes nějakou hranu (tj. $(x-1, y)$, $(x+1, y)$, $(x, y-1)$ a $(x, y+1)$, pokud tato políčka existují). Některá políčka jsou již od začátku osvětlená; mezi tato políčka patří $(1, 1)$ a (n, n).

Kdykoliv jsou organizátoři v nějakém políčku (x, y), mohou spustit speciální příkaz, který najednou rozsvítí všechny lampy v daném sloupci, tj. se souřadnicemi $(x, 1)$, $(x, 2)$, $\ldots$, (x, n). Zjistěte, zda resp. s kolika nejméně použitími speciálního příkazu se organizátoři mohou dostat z Matfyzu na kolej, pokud mohou vstupovat pouze na osvětlená políčka.

Formát vstupu

Na prvním řádku dostanete dvě mezerou oddělená přirozená čísla n a m. Na každém z následujících m řádků dostanete dvě mezerou oddělená přirozená čísla x, y značící, že políčko (x, y) je osvětlené. Můžete předpokládat, že políčka $(1, 1)$ i (n, n) osvětlená jsou.

Formát výstupu

Na výstup vypište jedno celé číslo, totiž nejmenší počet použití speciálního příkazu, případně -1, pokud se organizátoři z Matfyzu na kolej dostat nemohou.

Bodování

Ve všech vstupech platí $1 \leq n$, $2 \leq m \leq n^2$ a políčka $(1, 1)$ a (n, n) jsou osvětlená.

Jsou čtyři sady testovacích vstupů, v následující tabulce se pro každou sadu dozvíte počet bodů a horní odhady na n a m pro všechny vstupy v dané sadě:

body	n	m
3	10	100
2	100	6 000
3	2 000	6 000
2	2 000	4 000 000

Příklady

Vstup:	*Výstup:*
2 3 1 1 1 2 2 2	0

V tomto případě existuje osvětlená cesta bez jakéhokoliv hackování.

Vstup:	*Výstup:*
3 3 3 3 1 1 2 2	2

Nejprve mohou organizátoři rozsvítit první sloupec, přejít na políčko $(1, 2)$*, odtud na políčko* $(2, 2)$*, rozsvítit druhý sloupec, popojít na* $(2, 3)$ *a odtud už přímo na kolej.*

Vstup:	*Výstup:*
3 3 3 3 1 1 2 1	1

V tomto případě mohou organizátoři popojít doprava do $(2, 1)$*, tento sloupec rozsvítit, dojít po něm až dolů do* $(2, 3)$ *a odtud znovu na kolej.*

Vstup:	*Výstup:*
3 3 3 3 1 1 1 2	-1

V tomto případě nemají organizátoři možnost, jak se dostat do prostředního sloupce.

P-I-2 Inovativní vizualizace

Michal pracuje jako Junior Data Scientist v jedné nejmenované společnosti. Jejích n zaměstnanců je očíslovaných čísly $1, \ldots, n$, přičemž ředitelka má číslo 1. Každý zaměstnanec má nějakého přímého nadřízeného, kromě ředitelky, která žádné nadřízené nemá, a v této struktuře nejsou žádné cykly. Jednoho dne přišla za Michalem ředitelka, položila před něj papír s nakresleným pravidelným n-úhelníkem a řekla mu, že kvůli plánované reorganizaci potřebuje, aby tuhle strukturu vizualizoval tak, že každý zaměstnanec bude v nějakém vrcholu n-úhelníka, každý podřízený bude se svým nadřízeným spojený úsečkou a žádné dvě úsečky se nebudou křížit (tj. jediné místo, kde se mohou úsečky protnout, jsou jejich koncové body).

Protože má Michal rád tvůrčí svobodu, zajímalo by ho, kolik je možností, jak takovou vizualizaci udělat. (Dvě možnosti jsou různé, pokud existuje nějaký vrchol n-úhelníka, do nějž jsou umístěni různí zaměstnanci.) Protože to číslo může být velké, vypište jen jeho zbytek po dělení číslem $10^9 + 7$.

Formát vstupu

Na prvním řádku dostanete jedno přirozené číslo n. Na druhém řádku najdete $n-1$ mezerou oddělených přirozených čísel, i-té z nich značí číslo zaměstnance, který je přímým nadřízeným zaměstnance číslo $i + 1$.

Můžete předpokládat, že pro každé $i \geq 1$ má zaměstnanec $i + 1$ přímého nadřízeného s číslem nejvýše i.

Formát výstupu

Na výstup vypište jedno celé číslo, totiž zbytek počtu možností, jak vytvořit vyhovující vizualizaci, po dělení číslem $10^9 + 7$.

Bodování

Ve vstupech za celkem 3 body bude platit $n \leq 10$.

Ve vstupech za další 2 body bude platit $n \leq 100$, přičemž v polovině z nich (1 bod) bude ředitelka přímou nadřízenou všech ostatních zaměstnanců a ve druhé polovině (1 bod) bude platit, že každý zaměstnanec kromě zaměstnance n má právě jednoho přímého podřízeného.

Ve všech vstupech bude platit $3 \leq n \leq 10^5$.

Příklady

Vstup:	*Výstup:*
3 1 1	6

Firma má pouze tři zaměstnance, ředitelku, Michala a někoho třetího. V tomto případě každé přiřazení zaměstnanců do vrcholů pravidelného trojúhelníka bude vyhovující, dvě nakreslené úsečky se vždy protnou pouze ve vrcholu, kam bude umístěná ředitelka. Tento vstup se může objevit i ve druhé sadě, jelikož ředitelka je zde přímou nadřízenou všech ostatních zaměstnanců.

Vstup:	*Výstup:*
4 1 2 3	16

Jedna z vyhovujících možností je například umístit zaměstnance v pořadí podle hodinových ručiček 1 2 3 4*, nevyhovující možnost je naopak například* 1 3 2 4*, jelikož tehdy by se úsečky* 1 2 *a* 3 4 *křížily. Tento vstup se může objevit i ve druhé sadě, jelikož zde má každý zaměstnanec kromě zaměstnance číslo* 4 *právě jednoho přímého podřízeného.*

Vstup:	*Výstup:*
11 1 1 1 2 2 2 3 4 4 5	38016
Vstup:	*Výstup:*
15 1 1 1 2 2 2 3 4 4 5 8 8 8 10	2488320

P-I-3 Nevhodný dárek

Pepa miluje binární posloupnosti, což Ondra zřejmě nevěděl, jelikož mu dal k narozeninám přirozené číslo c. Jirka nabídl Pepovi, že mu z dárku vyrobí binární posloupnost postupným opakováním následujícího algoritmu:

1. Nechť má v aktuálním kroku Pepa posloupnost $a_1, \ldots, a_n$. (V prvním kroku má Pepa jednoprvkovou posloupnost $a_1 = c$.)
2. Jirka si vybere nějaký takový index $1 \leq i \leq n$, že $a_i > 1$. Pokud takový index neexistuje, $a_1, \ldots, a_n$ je výsledná posloupnost, kterou Jirka vrátí Pepovi.
3. Označme $b = \lfloor \frac{a_i}{2} \rfloor$ a $c = a_i \bmod 2$.
4. Potom Jirka z původní posloupnosti vyrobí posloupnost

$$a_1, \ldots, a_{i-1}, b, c, b, a_{i+1}, \ldots, a_n$$

a opakuje předchozí kroky.

Všimněte si, že vrácená posloupnost nezáleží na konkrétním průběhu Jirkova algoritmu. Například řekněme, že Pepa dostal číslo 6. V prvním kroku si Jirka vybere toto číslo a vyrobí posloupnost 3 0 3. Ve druhém kroku si vybere, řekněme, první trojku a vyrobí 1 1 1 0 3. Nakonec si Jirka vybere trojku na konci, vyrobí posloupnost 1 1 1 0 1 1 1, a jelikož v ní už neexistuje žádný další vhodný index i, tuto posloupnost Pepovi vrátí.

Pepa si Jirkovým návrhem není úplně jistý. Aby se mohl lépe rozhodnout, zajímal by ho počet jedniček ve výsledné posloupnosti mezi indexy ℓ a r (včetně). Dokážete mu tuhle informaci poskytnout?

Soutěžní úloha

Na vstupu dostanete přirozené číslo c a dvě přirozená čísla $1 \leq \ell \leq r$. Nechť $b_1, \ldots, b_n$ je výsledná posloupnost, již by Jirka Pepovi vrátil. Zjistěte počet jedniček v podposloupnosti $b_\ell, \ldots, b_r$. Můžete předpokládat, že $n \geq r$.

Formát vstupu

Na jediném řádku vstupu dostanete tři mezerou oddělená přirozená čísla c, ℓ a r.

Formát výstupu

Na výstup vypište jedno celé číslo, totiž počet jedniček v zadaném úseku.

Bodování

Řešení za 10 bodů by mělo být schopné efektivně vyřešit úlohu i pro $c = 10^{1000}$ (můžete předpokládat, že programovací jazyk umí relevantní aritmetické operace i s takto velkými čísly).

Až 6 bodů můžete získat, když budete navíc předpokládat, že $r - \ell \leq 100$.

Až 3 body můžete získat za řešení, které efektivně vyřeší úlohu pro $c \leq 10^5$.

Až 2 body můžete získat za řešení, které bude předpokládat, že $\ell = 1$ a $r = n$ (tedy zajímá nás počet jedniček v celé výsledné posloupnosti).

Příklady

Vstup:	*Výstup:*
5 2 4	2

Pepa z čísla 5 *vyrobí posloupnost* 1 0 1 1 1 0 1, *úloha se potom ptá na počet jedniček v podposloupnosti* 0 1 1.

P-I-4 Hvězdní věštci

K této úloze se vztahuje studijní text uvedený na následujících stranách. Doporučujeme vám nejprve si přečíst studijní text a až potom se vrátit k samotným soutěžním úkolům. Ze stejného studijního textu budou vycházet i úlohy v dalších kolech soutěže.

a) *(2 body)*

Navrhněte algoritmus, který rozhodne, zda Hvězdné impérium tvoří kružnici, tj. zda lze systémy seřadit do posloupnosti $s_0, s_1, \ldots, s_{N-1}$ tak, aby systém s_0 sousedil pouze se systémy s_1 a s_{N-1}, s_{N-1} pouze se systémy s_{N-2} a s_0 a pro $i = 1, \ldots, N-2$ systém s_i sousedil právě se systémy s_{i-1} a s_{i+1}.

b) *(3 body)*

Navrhněte algoritmus, který rozhodne, zda Hvězdné impérium má perfektní párování, tj. zda lze systémy rozdělit na dvojice tak, aby v každé dvojici systémy sousedily.

c) *(5 bodů)*

Navrhněte algoritmus, který rozhodne, zda Hvězdné impérium má hamiltonovskou cestu, tj. zda lze systémy seřadit do posloupnosti s_0, s_1, …, s_{N-1} tak, aby systém s_0 sousedil se systémem s_1, s_{N-1} se systémem s_{N-2} a pro $i = 1, \ldots, N-2$ systém s_i sousedil se systémy s_{i-1} a s_{i+1} (na rozdíl od úlohy ze studijního textu ale systémy mohou mít i další sousedy).

Připomínáme, že při hodnocení této úlohy *nezáleží na časové a paměťové složitosti* vašeho řešení, pouze na jeho správnosti, hodnotě K a vašem zdůvodnění.

Studijní text

Vědci Hvězdného impéria vyřešili již téměř všechny problémy, které říši trápí. Díky vynálezu hyperkvantových počítačů jsou například schopni provést libovolný výpočet v konstantním čase. Velkým problémem ale zůstává komunikace mezi hvězdnými systémy. Přímo poslat zprávu lze pouze do blízkých systémů, kterým budeme říkat *sousední;* i pro ně ale doručení zprávy trvá jeden rok.

Mezi každými dvěma systémy Hvězdného impéria se sice lze dostat přes posloupnost sousedních systémů (tj. graf, jehož hrany tvoří sousední systémy, je souvislý), ale některé systémy jsou tak daleko od sebe, že zaslání zprávy mezi nimi by trvalo několik tisíciletí, a je tedy zcela nepraktické. Tradičně tak jediným globálním komunikačním kanálem byla císařova telepatická schopnost komukoliv okamžitě předat jakoukoliv informaci. Tento kanál je bohužel jednosměrný, odpovědět obdobně císaři nelze.

Nedávno se podařilo najít alespoň částečné řešení a vytvořit bezdrátovou síť 115. generace, kterou každý může poslat zprávu všem v celém vesmíru a tato zpráva je doručena okamžitě. Komplikaci je ale nízká propustnost této sítě, konkrétně jeden bit za rok (pokud se více osob pokusí v ten samý rok síť použít, doručena je jen první z odeslaných zpráv). K řešení různých otázek proto Hvězdné impérium používá následující protokol, kombinující moderní techniku s tradičními metodami:

- Císař všem sdělí počet N hvězdných systémů, každému hvězdnému systému přidělí číslo od 0 do $N-1$ jako jednoznačný identifikátor a sdělí, jaký (všichni stejný) algoritmus mají použít k řešení dané otázky.
- Vládce každého hvězdného systému zajde do místní věštírny, kde mu sdělí K-bitový řetězec R (ne nutně ve všech hvězdných systémech stejný).
- Na základě R a předepsaného algoritmu pošle všem sousedním (tj. nejvýše jeden světelný rok vzdáleným) systémům zprávu.
- Poté, co přijme zprávy od všech sousedních systémů, na jejich základě dle předepsaného algoritmu rozhodne, zda odpověď na otázku je ANO nebo NE. Je-li odpověď NE, pošle všem bezdrátovou sítí jeden bit a popraví věštce.
- Pokud nikdo po jednom roce tento bit neposlal, odpověď je ANO.

Funkčnost tohoto systému je zaručena tím, že, jak se obecně ví, věštci disponují magickou schopností si zachránit život, pokud je to jen možné. Existuje-li tedy nějaká volba řetězců R taková, že ve všech systémech daný algoritmus dospěje k odpovědi ANO, věštci ji dokážou najít.

Věštci si účtují horentní sumy za každý vyvěštěný bit, naším cílem je tedy najít algoritmus s co nejmenším K.

Příklad 1: Tři týmy

Chceme vědět, zda se hvězdné systémy mohou rozdělit do tří týmů tak, aby žádné dva systémy patřící do jednoho týmu nesousedili. Vládce každého systému si tedy nechá vyvěštit dva bity 01, 10 nebo 11 (pokud by věštec vyvěštil 00, rovnou ho

popraví), udávající číslo týmu, do kterého bude jeho systém patřit. Toto číslo pak pošle všem sousedním systémům. Odpověď je NE, pokud od některého ze sousedních systémů dostane stejné číslo.

Lze-li systémy do tří týmů rozdělit, věštci mohou pro každého uhodnout číslo jeho týmu, všichni přežijí a víme, že na otázku je odpověď ANO. Pokud to není možné, nějací dva sousední věštci musí zvolit stejné číslo, budou popraveni a odpověď je NE.

Popsaná strategie používá $K = 2$.

Příklad 2: Univerzální řešení

Libovolnou úlohu můžeme vyřešit tak, že si od věštce necháme vyvěštit mapu celého hvězdného impéria, tj. matici o rozměrech $N \times N$, kde v i-tém řádku a j-tém sloupci je 1, právě když hvězdné systémy i a j sousedí. Tuto mapu pošleme sousedům, a pokud od některého souseda dostaneme jinou mapu, popravíme věštce. Jelikož je Hvězdné impérium souvislé, tímto zaručíme, že věštci všem musí vyvěštit stejnou mapu. Také všem sousedům pošleme svůj identifikátor a na základě identifikátorů, které obdržíme od sousedů, ověříme, že jsou v mapě správně uvedeni naši sousedé (pokud ne, věštce opět popravíme). Tím zaručíme, že vyvěštěná mapa správně popisuje celé Hvězdné impérium.

Jelikož máme k dispozici neomezenou výpočetní sílu, každý systém zvlášť pak může úlohu vyřešit a odpovědět ANO nebo NE. Tato strategie používá $K = N^2$; obecně proto získáte body pouze za řešení, pro něž je K výrazně menší než N^2.

Popis algoritmu

Algoritmus můžete buď dostatečně přesně popsat, nebo zapsat ve vhodném pseudokódu. V tomto pseudokódu můžete používat:

- Počet systémů N a identifikátor A aktuálního systému, což je přirozené číslo mezi 0 a $N - 1$.
- Proměnnou R libovolného typu, v níž je uložena věštba.
- Přirozené číslo D, udávající počet sousedních systémů.
- Pole `send` obsahující D prvků libovolného typu, na i-tou pozici v tomto poli ukládáte zprávu, kterou pošlete i-tému ze sousedních systémů.
- Pole `receive` obsahující D prvků stejného typu, na i-té pozici naleznete zprávu, kterou obdržíte od i-tého ze sousedních systémů. Toto pole smíte číst až poté, co provedete všechny zápisy do pole `send`.
- Váš program na konci musí vrátit odpověď ANO nebo NE.

Aby bylo zřejmé, kolik bitů věštby používáte, můžete v definici typu R použít například typ `unsigned(s)`, označující s-bitové přirozené číslo.

Ve svých řešeních neurčujte paměťovou a časovou složitost, na té nám nezáleží. Vaše řešení bude hodnoceno pouze na základě jeho správnosti, zdůvodnění a hodnoty K.

Příklad 3: Cesta

Nechť $N \geq 2$. Chceme rozhodnout, zda Hvězdné impérium tvoří cestu, tj. zda lze systémy seřadit do posloupnosti $s_0, s_1, \ldots, s_{N-1}$ tak, aby systém s_0 sousedil pouze se systémem s_1, s_{N-1} pouze s s_{N-2} a pro $i = 1, \ldots, N-2$ systém s_i sousedil právě se systémy s_{i-1} a s_{i+1}.

Má-li některý systém více než dva sousedy, odpověď je zjevně NE. Pokud všechny mají nejvýše dva sousedy, pak jelikož je Hvězdné impérium souvislé, musíme pouze rozhodnout, zda tvoří cestu (tj. právě dva systémy mají jen jednoho souseda) nebo kružnici (všechny systémy mají dva sousedy). To můžeme rozhodnout například tak, že nám věštec řekne naši pozici p v posloupnosti a to, který z našich dvou sousedů je před námi, na základě těchto informací oběma sousedům pošleme jejich pozici v posloupnosti (jednomu z nich $p-1$ a druhému $p+1$) a odpovíme ANO jen tehdy, když od obou sousedů dostaneme p. Systémy s pouze jedním sousedem postupují obdobně, jen ještě ověří, zda jejich pozice je 0 nebo $N-1$.

V pseudokódu můžeme algoritmus zapsat následovně (funkce `ceil` značí horní celou část a `log` je dvojkový logaritmus):

```
typedef unsigned(ceil(log(N))) pozice;

struct
{
  pozice p;
  unsigned(1) pred;
} R;

pozice send[D], receive[D];

if (D > 2)
  return NE;

if (D == 2)
  {
    if (R.p == 0 || R.p > N - 2)
      return NE;

    send[R.pred] = R.p - 1;
    send[1 - R.pred] = R.p + 1;

    if (receive[0] != R.p || receive[1] != R.p)
      return NE;
  }
else // D == 1
  {
    if (R.p == 0)
      send[0] = 1;
    else if (R.p == N - 1)
      send[0] = N - 2;
    else
      return NE;

    if (receive[0] != R.p)
      return NE;
  }

return ANO;
```

Pokud Hvězdné impérium tvoří kružnici, uvažme systém, kterému věštec vyvěští nejmenší hodnotu p. Tento systém jednomu ze svých sousedů pošle zprávu $p-1$, která je nutně menší než jemu vyvěštěná hodnota. Tento soused tedy nutně odpoví NE. Naopak, pokud Hvězdné impérium tvoří cestu a všichni věštci správně vyvěští pozici v posloupnosti, všichni odpoví ANO.

Dostáváme tedy řešení s $K = \lceil \log_2 N \rceil + 1$.