Informácie a pravidlá

Pre koho je súťaž určená?

Do **kategórie B** sa smú zapojiť len tí žiaci základných a stredných škôl, ktorí ešte ani v tomto, ani v nasledujúcom školskom roku nebudú končiť strednú školu.
Do **kategórie A** sa môžu zapojiť všetci žiaci (základných aj) stredných škôl.

Odovzdávanie riešení domáceho kola

Riešitelia domáceho kola odovzdávajú riešenia sami, v elektronickej podobe, a to priamo na stránke olympiády: http://oi.sk/. Odovzdávanie riešení bude spustené niekedy v septembri.

Riešenia kategórie A je potrebné odovzdať najneskôr **15. novembra 2021**.
Riešenia kategórie B je potrebné odovzdať najneskôr **30. novembra 2021**.

Priebeh súťaže

Za každú úlohu domáceho kola sa dá získať od 0 do 10 bodov. Na základe bodov domáceho kola stanoví Slovenská komisia OI (SK OI) pre každú kategóriu bodovú hranicu potrebnú na postup do **krajského kola**. Očakávame, že táto hranica bude približne rovná **tretine maximálneho počtu bodov**.

V krajskom kole riešitelia riešia štyri teoretické úlohy, ktoré môžu tematicky nadväzovať na úlohy domáceho kola. V kategórii B súťaž týmto kolom končí.

V kategórii A je približne najlepších 30 riešiteľov krajského kola (podľa počtu bodov, bez ohľadu na kraj, v ktorom súťažili) pozvaných do **celoštátneho kola**. V celoštátnom kole účastníci prvý deň riešia teoretické a druhý deň praktické úlohy. Najlepší riešitelia sú vyhlásení za víťazov. Približne desať najlepších riešiteľov následne SK OI pozve na týždňové výberové sústredenie. Podľa jeho výsledkov SK OI vyberie družstvá pre Medzinárodnú olympiádu v informatike (IOI) a Stredoeurópsku olympiádu v informatike (CEOI).

Ako majú vyzerať riešenia úloh?

V praktických úlohách je vašou úlohou vytvoriť program, ktorý bude riešiť zadanú úlohu. Program musí byť v prvom rade korektný a funkčný, v druhom rade sa snažte aby bol čo najefektívnejší.

V kategórii B môžete použiť ľubovoľný programovací jazyk.

V kategórii A musíte riešenia praktických úloh písať v jednom z podporovaných jazykov (napr. C++, Pascal alebo Java). Odovzdaný program bude automaticky otestovaný na viacerých vopred pripravených testovacích vstupoch. Podľa toho, na koľko z nich dá správnu odpoveď, vám budú pridelené body. Výsledok testovania sa dozviete krátko po odovzdaní. Ak váš program nezíska plný počet bodov, budete ho môcť vylepšiť a odovzdať znova, až do uplynutia termínu na odovzdávanie.

Presný popis, ako majú vyzerať riešenia praktických úloh (napr. realizáciu vstupu a výstupu), nájdete na webstránke, kde ich budete odovzdávať.

Ak nie je v zadaní povedané ináč, riešenia teoretických úloh musia v prvom rade obsahovať **podrobný slovný popis použitého algoritmu, zdôvodnenie jeho správnosti** a diskusiu o efektivite zvoleného riešenia (t. j. posúdenie časových a pamäťových nárokov programu). Na záver riešenia uveďte program. Ak používate v programe netriviálne algoritmy alebo dátové štruktúry (napr. rôzne súčasti STL v C++), súčasťou popisu algoritmu musí byť dostatočný popis ich implementácie.

Usporiadateľ súťaže

Olympiádu v informatike (OI) vyhlasuje *Ministerstvo školstva SR* v spolupráci so *Slovenskou informatickou spoločnosťou* (odborným garantom súťaže) a *Slovenskou komisiou Olympiády v informatike*. Súťaž organizuje *Slovenská komisia OI* a v jednotlivých krajoch ju riadia *krajské komisie OI*. Na jednotlivých školách ju zaisťujú učitelia informatiky. Celoštátne kolo OI, tlač materiálov a ich distribúciu po organizačnej stránke zabezpečuje IUVENTA v tesnej súčinnosti so Slovenskou komisiou OI.

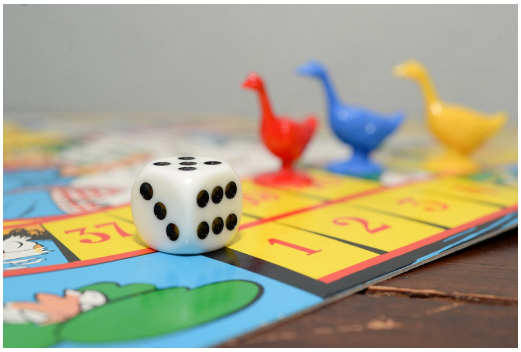
Covid-19

Krajské kolo súťaže je plánované na 19. januára 2021 a celoštátne kolo na 24.-27. marca 2021. V súčasnosti ešte nevieme zaručiť, či tieto kolá prebehnú tradičnou prezenčnou formou.

Slovenská komisia OI dôsledne sleduje aktuálny vývoj epidemiologickej situácie a v prípade potreby mu prispôsobí organizáciu súťaže. Domáce kolo súťaže určite prebehne v tradičnej podobe, keďže naň v súčasnosti platné opatrenia nemajú žiaden vplyv. O presnej forme a spôsobe organizácie ďalších kôl súťaže budú postupujúci súťažiaci informovaní priebežne.

B-I-1 Jamy a rebríky

*Toto je **praktická úloha**. Pomocou webového rozhrania odovzdajte **funkčný, odladený program** v jednom z podporovaných jazykov. Detailnejšie pokyny k písaniu a odovzdávaniu programov nájdete tu: http://oi.sk/oisubmit/howto.php.*



V mestečku Las Vegetas sa nehrajú na náhodu, všetci majú vždy neuveriteľné šťastie. Z balíka kariet vždy vytiahnu žolíka, na minci vždy hodia hlavu a na kocke každému padne presne to, čo si zaželá pred hodom.

Všetky kasína už dávno skrachovali, väčšina doskových hier stratila zmysel, ale jedna spoločenská hra je medzi obyvateľmi stále populárna. *Jamy a Rebríky* sú variácia na známu hru *Hady a rebríky*, v ktorej sú namiesto hadov jamy. Presné pravidlá sú popísané v časti Súťažná úloha.

Aj napriek tomu, že celá hra spočíva v hádzaní kockou, nie je až také jednoduché (minimálne pre obyvateľov Las Vegetas) naplánovať si hody kockou tak, aby prešli celý plánik na čo najmenej hodov. Aj keď si vymyslia nejakú stratégiu, ako majú vedieť, či našli najlepšie riešenie?

Napíšte program, ktorý pre daný herný plánik *Jám a Rebríkov* zistí, na koľko najmenej hodov kockou sa dá plánik prejsť.

Súťažná úloha

Hrací plán hry *Jamy a Rebríky* pozostáva z n políčok očíslovaných 1 až n. Hráč začína na políčku 1 a jeho cieľom je dostať sa pomocou hodov kockou na políčko n. Každým hodom sa hráč posunie dopredu o jedno až šesť políčok, podľa toho, koľko bodiek padlo na kocke. (Po šestke sa už nehádže znova, ak padne šestka, hráč sa pohne dopredu o šesť políčok.)

Pripomíname, že v Las Vegetas si vie každý hráč pred každým hodom zvoliť, čo mu padne na kocke, a teda o koľko políčok sa posunie.

Z niektorých políčok vedú rebríky na políčka s vyšším číslom. Pokiaľ hráč v nejakom kole ukončí svoj pohyb dopredu na políčku, kde je začiatok rebríka, automaticky sa presunie na koniec tohto rebríka. Tento posun po rebríku je ešte súčasťou toho istého ťahu. Hráč nemá na výber, vyššie popísaný pohyb po rebríku musí vykonať bez ohľadu na to, či chce alebo nechce. Pokiaľ sa hráč po presune rebríkom ocitne na začiatku ďalšieho rebríka, už sa neposúva ďalej.

Napríklad si predstavme, že hráč stojí na políčku 7*, z políčka* 10 *vedie rebrík na políčko* 15 *a z políčka* 15 *vedie rebrík na políčko* 20*. Pokiaľ hráč hodí na kocke štyri, posunie sa v tomto ťahu z políčka* 7 *na políčko* 11*. Ak by však hodil na kocke číslo tri, presunie sa na políčko* 10*, odtiaľ vylezie po rebríku na políčko* 15 *a tam ťah skončí.*

Na niektorých políčkach sa nachádzajú jamy. Na tieto políčka hráč nesmie nikdy stúpiť. Hráč nesmie ani použiť rebrík, ktorý začína alebo končí na políčku s jamou.
Zistite, na koľko najmenej hodov kockou sa dá dostať z políčka 1 na políčko n.

Formát vstupu a výstupu

V prvom riadku vstupu je celé číslo $n > 1$ – počet políčok herného plánika.
V druhom riadku je n medzerou oddelených celých čísel a_i.

- $a_i = -1$, ak je na i-tom políčku jama
- $a_i > 0$, ak z i-teho políčka vedie rebrík na políčko $i + a_i$.
- $a_i = 0$, ak na i-tom políčku nie je ani jama ani rebrík.

Prvé políčko nikdy neobsahuje jamu. Všetky rebríky vedú do hracieho plánika. (Ak $a_i > 0$, tak $i + a_i \leq n$.)

Na výstup vypíšte jeden riadok obsahujúci jedno číslo: najmenší možný počet ťahov, za ktorý sa vieme dostať do cieľa. Ak sa nedá dostať do cieľa, vypíšte namiesto toho číslo -1.

Obmedzenia a hodnotenie

Je desať sád vstupov. Ak tvoj program vyrieši všetky vstupy v danej sade, dostaneš za sadu jeden bod. V niektorých sadách sa nenachádzajú rebríky ($a_i \leq 0$) v niektorých sa nenachádzajú jamy ($a_i \geq 0$), podľa tabuľky nižšie.

číslo sady	1	2	3	4	5	6	7	8	9	10
maximálne n	15	100	1 000	1 000	1 000	10 000	100 000	100 000	100 000	100 000
bez rebríkov	nie	nie	áno	nie	nie	nie	áno	nie	nie	nie
bez jám	nie	nie	nie	áno	nie	nie	nie	áno	nie	nie

Príklady

vstup

```
13
0 4 6 0 -1 -1 -1 0 2 0 -1 1 0
```

výstup

```
2
```

V prvom ťahu hodíme na kocke 2, čím sa posunieme na políčko 3 a odtiaľ rebríkom na políčko 9. V druhom ťahu hodíme na kocke 4, čím sa presunieme na cieľové políčko 13.

vstup

```
8
0 -1 -1 -1 -1 -1 -1 0
```

výstup

```
-1
```

Všetky políčka, na ktoré vieme skočiť v prvom ťahu, obsahujú jamu.

vstup

```
5
4 3 2 1 -1
```

výstup

```
-1
```

Aj keď z každého políčka vedie rebrík do cieľa, v cieli je jama, a teda tam nevieme prísť.

vstup

```
20
0 0 6 0 0 3 0 0 11 0 0 0 0 0 4 0 0 0 0 0
```

výstup

```
2
```

Postupne hodíme dve štvorky. V prvom ťahu teda prejdeme z políčka 1 na políčko 5. Následne v druhom ťahu prejdeme odtiaľ na políčko 9 a z políčka 9 po rebríku rovno do cieľa. Všimnite si, že ak by sme použili prvý alebo druhý rebrík (z políčka 3 na políčko 9, resp. z políčka 6 na políčko 9), potrebovali by sme celkovo aspoň tri ťahy.

B-I-2 Zaujímavé faktoriály

*Toto je **praktická úloha**. Riešiť sa dá v ľubovoľnom programovacom jazyku, odovzdávajú sa len správne riešenia pre vopred pripravené testovacie dáta. Detailnejšie pokyny k odovzdávaniu riešení sú uvedené nižšie.*

Romanku odjakživa fascinovali faktoriály. Niet teda divu, že keď sa v škole na hodine slovenčiny rozoberá literárny realizmus v 19. storočí, tak si Romanka radšej kreslí a počíta. Zaujalo ju, ako rýchlo hodnoty faktoriálu rastú: už také číslo 30! je obrovské. Ešte zaujímavejšie však je, že toto číslo končí až siedmimi nulami.
Romanka už vie pre konkrétny faktoriál určiť počet núl na jeho konci. Teraz jej napadlo, čo by sa stalo, keby tých faktoriálov bolo viacero. Nedalo by sa napríklad z počtu núl na konci čísla 30! odvodiť počet núl v číslach 60! alebo 31!? Aby si úlohu zjednodušila, povedala si, že ju bude zaujímať vždy iba niekoľko za sebou idúcich faktoriálov. Táto úloha je však nad jej sily, a tak poprosila o pomoc vás.

Súťažná úloha

Hodnota „n faktoriál“ (značíme $n!$) je súčinom prvých n prirodzených čísel. Napr. $5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$. Špeciálne platí, že $0! = 1$.
Na vstupe dostanete dve čísla a a b: začiatok a koniec rozsahu, o ktorý sa Romanka zaujíma.
Váš program by mal na výstup vypísať celkový počet núl na konci čísel $a!$, $(a+1)!$, …, $(b-1)!$, $b!$.

Formát vstupu a výstupu

Vstup pozostáva z dvoch riadkov. V prvom je číslo a, v druhom číslo b. Na výstup vypíšte jediný riadok obsahujúci práve jedno číslo x – súčet počtov núl nachádzajúcich sa na konci čísel $a!, (a+1)!, \ldots, (b-1)!, b!$.

Odovzdávanie riešení

Toto je praktická úloha. Napíšte **v ľubovoľnom programovacom jazyku** program, ktorý ju rieši.
Zo stránky `http://oi.sk/` stiahnite ZIP archív obsahujúci 10 testovacích vstupov, nazvaných `01.txt` až `10.txt`. (Linky: verzia pre linux, verzia pre windows.)
Vyrobte k čo najviac vstupom správne výstupy a uložte ich do súborov `sol01.txt` až `sol10.txt`.
Odovzdajte ZIP archív obsahujúci **zdrojový kód vášho programu** a tieto výstupné súbory.
Za každý správny výstupný súbor získate 1 bod.

Veľkosti vstupov

Pre všetky vstupy platí obmedzenie, že $0 \leq a \leq b \leq 10^9$.
Pre niektoré vstupy platia obmedzenia navyše:
V prvých dvoch vstupoch platí $b \leq 5000$, v ďalších dvoch $a = b$ a v ďalších troch platí $a = 0$.

Čísla na vstupe sa zmestia do bežnej 32-bitovej celočíselnej premennej (napr. `int` v C/C++/Jave). Toto ale nemusí vždy platiť aj pre výsledok. Zaručujeme, že správny výsledok sa zmestí do 64-bitovej celočíselnej premennej (napr. `long long` v C/C++ alebo `long` v Jave).

Príklad

vstup	výstup
4 10	7

Faktoriály v tomto rozsahu majú nasledovné hodnoty: $4! = 24$, $5! = 120$, $6! = 720$, $7! = 5\,040$, $8! = 40\,320$, $9! = 362\,880$ a $10! = 3\,628\,800$. *Vidíme, že čísla* $5!$ *až* $9!$ *majú na konci po jednej nule, číslo* $10!$ *končí dvoma nulami. Celkový počet núl na koncoch týchto faktoriálov je teda 7.*

vstup	výstup
1578 6784179	5753103797469

B-I-3 Domáca úloha z elektrotechniky

Toto je teoretická úloha. Pomocou webového rozhrania odovzdajte súbor vo formáte PDF, obsahujúci riešenie, spĺňajúce požiadavky uvedené v pravidlách.

Roman je študent elektrotechniky. Škola je to pomerne náročná, najmä kvôli veľkému počtu rôznych projektov, Roman neustále niečo konštruuje, spájkuje, programuje... Naposledy im učiteľ prikázal vytvoriť mechanické zariadenie, ktoré dostane pás guličiek a tieto guličky usporiada podľa hmotnosti od najľahšej po najťažšiu.
Romanovi sa hneď vybavil problém „triedenia“, o ktorom počul na sústredení KSP. Spomenul si, že preň existujú také algoritmy, ktorým stačí vymieňať susedné prvky. Za takéto riešenie síce nedostane v škole plný počet bodov, bude sa ale aspoň ľahko zostrojovať ako stroj.
Skonštruoval teda jednoduché robotické rameno, ktoré sa pohybovalo popri páse guličiek, vedelo ich dvíhať, vážiť a v prípade potreby vymieňať. Pri návrhu a meraní sa však pomýlil a rameno **nevymieňa susedné** guličky, ale vie vymeniť iba takú dvojicu guličiek, medzi ktorými je **práve jedna** iná gulička.
Roman vie, že takéto riešenie je nesprávne – napríklad rad guličiek s váhami $(4, 5, 2)$ sa mu usporiadať nepodarí. Jediná dvojica guličiek, ktorú vie vymeniť, sú totiž guličky s váhami 2 a 4 (len medzi nimi je práve jedna iná gulička). Jediné iné poradie, ktoré vie vyrobiť, je teda poradie $(2, 5, 4)$, to však tiež nie je usporiadané.
Všimol si však, že niektoré postupnosti guličiek predsa len usporiadať vie. Napríklad postupnosť $(7, 8, 10, 4, 2)$ usporiada tak, že vymení guličky 10 a 2 na $(7, 8, 2, 4, 10)$, potom 7 a 2 na $(2, 8, 7, 4, 10)$ a následne 4 a 8.
Nemá čas svoje riešenie prerábať, skúsi preto učiteľa presvedčiť, že jeho riešenie funguje tak, že mu ukáže také postupnosti guličiek, ktoré fungujú. Ktoré to však sú?

Súťažná úloha

Na vstupe dostanete postupnosť n čísel predstavujúcich váhy guličiek v rovnakom poradí, v akom sú uložené na páse. Vašou úlohou je zistiť, či sa takáto postupnosť guličiek dá usporiadať od najľahšej po najťažšiu pomocou ľubovoľného počtu takých výmien, ktoré menia dvojicu guličiek, medzi ktorými je jedna iná gulička. Teda každá výmena vymení guličky na pozíciách k a $k + 2$ pre nejaké $1 \leq k \leq n - 2$.

Formát vstupu a výstupu

V prvom riadku vstupu je číslo n – počet guličiek v postupnosti. V druhom riadku je n medzerou oddelených kladných čísiel do 10^9, ktoré označujú váhy jednotlivých guličiek. Váhy **nemusia byť** navzájom rôzne.
Na výstup vypíšte `ano` alebo `nie` podľa toho, či Romanovo zariadenie vie usporiadať postupnosť zo vstupu.

Obmedzenia a hodnotenie

V popise riešenia nezabudnite zdôvodniť, prečo vaše riešenie funguje a rozoznáva všetky postupnosti, ktoré je možné usporiadať.
Plných 10 bodov môžu získať riešenia, ktoré efektívne vyriešia ľubovoľný vstup, v ktorom platí $n \leq 10^5$.
Za riešenia, ktoré efektívne vyriešia ľubovoľný vstup s $n \leq 5000$, môžete získať najviac 5 bodov.
Za ľubovoľné správne riešenie je možné získať aspoň 3 body.

Príklady

vstup

```
3
4 5 2
```

výstup

```
nie
```

vstup

```
5
7 8 10 4 2
```

výstup

```
ano
```

vstup

```
5
7 7 1 4 5
```

výstup

```
ano
```

B-I-4 Školský výlet

*Toto je **praktická optimalizačná úloha**. Riešiť sa dá v ľubovoľnom programovacom jazyku. Odovzdáva sa najlepšie nájdené riešenie. Čím je lepšie, tým viac bodov zaň udelíme.*

Krištofova trieda sa rozhodla zorganizovať si výlet: postupne navštívia všetkých 141 slovenských miest. Teraz plánujú, ako ich všetky navštíviť. Dohodli sa, že spravia jednu veľkú okružnú cestu, počas ktorej sa zastavia v každom z týchto miest **práve raz**. Začínať aj končiť budú v domovskom Lučenci.
Jedným z najväčších nákladov takejto cesty je, samozrejme, cestovné. Objednaný autobus sa platí od každého prejdeného kilometra, je preto veľmi dôležité si zvoliť správne poradie navštívených miest. Po Bratislave sa asi neoplatí ísť hneď do Košíc, ak ešte nenavštívili Senec.

Autobusová spoločnosť im poskytla cestné vzdialenosti medzi všetkými mestami. Nájdete ich tu: http://oi.sk/archiv/2021/sl-2021-1-ine-B4-vzdialenosti.txt
V súbore je uložená tabuľka rozmerov 142x142 vo formáte CSV (comma separated values – čiarkou oddelené hodnoty). V prvom riadku aj stĺpci tabuľky sú názvy miest, vo zvyšku tabuľky sú vzdialenosti medzi nimi. Teda napríklad druhý riadok súboru obsahuje postupné vzdialenosti z Bánoviec nad Bebravou do Bánoviec nad Bebravou (0 km), do Banskej Bystrice (112 km), do Banskej Štiavnice (78 km), a tak ďalej.

Tabuľka je symetrická: vzdialenosť z `A` do `B` je vždy rovnaká ako vzdialenosť z `B` do `A`. **Ale pozor!** Autobusová spoločnosť si cestu volí podľa toho, medzi ktorými dvoma mestami práve ide. Nemusí ísť nutne vždy o najkratšiu možnú cestu, zohľadňujú aj iné faktory (napríklad obmedzenia rýchlosti). V dátach, ktoré máte, preto občas nastávajú prípady, keď pre tri mestá `A`, `B` a `C` platí to, že priama cesta z `A` do `C` je dlhšia ako súčet cesty z `A` do `B` a z `B` do `C`. Toto samozrejme môžete šikovne využiť tak, že medzi `A` a `C` naplánujete zastávku v `B`. Ak však pôjdete z `A` priamo do `C`, autobus pôjde dlhšou cestou – tou, ktorú máte v tabuľke.

Krištof so spolužiakmi teraz rozmýšľajú ako navrhnúť trasu výletu a úloha je to teda nesmierne ťažká. Skúste im pomôcť a nájsť **čo najkratšiu okružnú trasu** s práve 141 zastávkami – raz v každom meste.

Súťažná úloha

K dispozícii máte tabuľku vzdieleností medzi všetkými mestami na Slovensku. Vašou úlohou je zvoliť konkrétne poradie, v ktorom tieto mestá navštíviť, každé práve raz. Snažíte sa pritom, aby celková vzdialenosť, ktorú autobus prejde, bola čo najmenšia.
Poradie musí začínať Lučencom. Nezabudnite, že aj návrat z posledného navštíveného mesta späť do Lučenca je súčasťou okružnej cesty.

Formát výstupu

Odovzdajte textový súbor obsahujúci presne 141 riadkov. Jednotlivé riadky musia obsahovať názvy miest v poradí, v ktorom ich chcete navštíviť. Názvy miest uveďte presne tie použité v tabuľke vzdialeností. V prvom riadku výstupu musí teda byť reťazec `Lucenec`.

Hodnotenie

Čím lepšie riešenie nájdete, tým viac bodov zaň dostanete. Približné hranice sú nasledovné: Za riešenie, ktoré najazdí 7000 km, budú 2 body. Za riešenie, ktoré najazdí 4500 km, bude 5 bodov. Za riešenie, ktoré najazdí 3140 km, bude 8 bodov.

TRIDSIATY SIEDMY ROČNÍK OLYMPIÁDY V INFORMATIKE

Príprava úloh: Michal Anderle, Michal Forišek, Jano Hozza, Andrej Korman
Recenzia: Michal Forišek
Slovenská komisia Olympiády v informatike
Vydal: IUVENTA – Slovenský inštitút mládeže, Bratislava 2021