



Priebeh krajského kola

Krajské kolo 39. ročníka Olympiády v informatike, kategória A, sa koná 23. 1. 2024 v dopoludňajších hodinách. Na riešenie úloh majú súťažiaci **4 hodiny čistého času**. Rôzne úlohy riešia súťažiaci na samostatné listy papiera. Akékoľvek pomôcky okrem písacích potrieb (napr. knihy, výpisy programov, kalkulačky) sú zakázané.

Čo má obsahovať riešenie úlohy?

- Slovné popíšte algoritmus.
Slovný popis riešenia musí byť jasný a zrozumiteľný i bez nahliadnutia do samotného algoritmu/programu.
- Zdôvodnite správnosť vášho algoritmu.
- Uvedte a zdôvodnite jeho časovú a pamäťovú zložitosť.
- Podrobne uveďte dôležité časti algoritmu, ideálne vo forme programu v nejakom bežnom programovacom jazyku (napr. C++, Python, Java, Pascal).
- V prípade, že používate vo svojom programovacom jazyku knižnice, ktoré obsahujú implementované dátové štruktúry a algoritmy (napr. STL pre C++), v popise algoritmu stručne vysvetlite, ako by ste napísali program s rovnakou časovou zložitosťou bez použitia knižnice.

Hodnotenie riešení

Za každú úlohu môžete získať od 0 do 10 bodov.

Pokiaľ nie je v zadaní povedané ináč, najdôležitejšie dve kritériá hodnotenia sú v prvom rade **správnosť** a v druhom rade **efektívnosť** navrhnutého algoritmu. Na výslednom počte bodov sa môže prejaviť aj kvalita popisu riešenia a zdôvodnenie tvrdení o jeho správnosti a efektívnosti.

Efektívnosť algoritmu posudzujeme vypočítaním jeho časovej zložitosti – funkcie, ktorá hovorí, ako dlho vykonanie algoritmu trvá v závislosti od veľkosti vstupných parametrov. Nezávisí pri tom na konštantných faktoroch, len na rádovej rýchlosti rastu tejto funkcie.

V zadaní úloh uvádzame časť „Hodnotenie“, v ktorej nájdete približné limity na veľkosť vstupných údajov. Pod pojmom „efektívne vyriešiť“ chápeme to, že váš program spustený na modernom počítači by mal dať odpoveď nanajvýš do niekoľkých sekúnd.

Údaje z tejto časti zadania by mali slúžiť hlavne na to, aby ste o riešení, ktoré vymyslíte, vedeli približne povedať, koľko bodov zaň dostanete.



A-II-1 Peťa lyžuje¹

Peťa sa čoskoro spustí dole traťou druhého kola slalomu. Ak by celú trať zišla čisto, momentálne vedúcu Michaelu by porazila o t stotín sekundy.

Na trati je n bránok, očíslovaných zhora dole od 1 po n . O každej bránke vieme dve čísla: pravdepodobnosť p_i (v percentách) toho, že ju Peťa prejde čisto, a čas s_i (v stotínach sekundy), ktorý oproti optimálnej jazde stratí, ak sa jej čistý prejazd tejto bránky nepodarí.

Súťažná úloha

Napíšte program, ktorý vypočíta pravdepodobnosť toho, že Peťa tento slalom vyhrá – teda toho, že dokopy chybami na trati stratí nanajvýš t stotín sekundy. Pozorne si pozrite v nasledujúcej sekcii **obmedzenia**, pre ktoré máte túto úlohu vyriešiť.

Váš program smie pracovať s reálnymi číslami. Pri jeho písaní môžete predpokladať, že všetky operácie s nimi sú matematicky presné. (Inými slovami, nemusíte sa zaoberať zaokrúhľovacími chybami, ktoré by nastali pri behu vášho programu na skutočnom počítači.)

V prvom riadku vstupu sú čísla n a t . Zvyšok vstupu tvorí n riadkov, v i -tom z nich sú hodnoty p_i a s_i . Všetky čísla na vstupe sú **celé**. Pravdepodobnosti p_i sú v percentách a sú z rozsahu od 1 po 99 (teda nie 0 ani 100).

Na výstup vypíšete jedno reálne číslo: pravdepodobnosť v percentách toho, že Peťa vyhrá.

Obmedzenia a hodnotenie

Plný počet bodov môžu získať riešenia efektívne pre $n, t \leq 5\,000$. (Optimálna pamäťová zložitosť nie je nutná.)

Najviac 8 bodov môžu získať riešenia efektívne pre $n, t \leq 5\,000$, ktoré fungujú **za dodatočného predpokladu**, že všetky straty s_i majú rovnakú hodnotu – teda že platí $s_1 = s_2 = \dots = s_n$.

Najviac 5 bodov môžu získať riešenia efektívne pre $n \leq 5\,000$, ktoré fungujú **za silnejšieho dodatočného predpokladu**, že $t = 25$ a $s_1 = s_2 = \dots = s_n = 10$.

Najviac 3 body môžu získať riešenia pôvodnej úlohy efektívne pre $n \leq 20$.

Ak vaše riešenie využíva nejaký dodatočný predpoklad, na začiatku **výrazne** uveďte, **ktorý** predpoklad využíva.

Príklady

vstup

```
3 25
90 10
90 10
70 10
```

výstup

```
99.7
```

Na trati sú tri bránky. Peťa má náskok $t = 25$ stotín sekundy. Chyba na každej bránke ju stojí 10 stotín sekundy. O výhru teda príde len vtedy, ak spraví chyby

na všetkých troch bránkach. To sa stane s pravdepodobnosťou $10\% \times 10\% \times 30\% = 0.1 \times 0.1 \times 0.3 = 0.003 = 0.3\%$. S pravdepodobnosťou $100\% - 0.3\% = 99.7\%$ teda vyhrá.

vstup

```
1 10
90 10
```

výstup

```
100
```

Aj s chybou Peťa ešte stále vyhrá. (Presnejšie, skončí na delenom prvom mieste, ale aj to sa ešte počíta.)

vstup

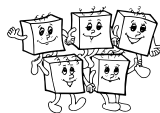
```
4 25
50 27
80 11
70 19
90 12
```

výstup

```
45.8
```

Prvú bránku Peťa musí prejsť čisto, inak rovno stratí prvé miesto. Ak pokazí tretiu, zvyšné dve (druhú a štvrtú) už musí prejsť čisto. Ak prejde čisto prvú aj tretiu, môže pokaziť hocijakú podmnožinu zvyšných.

¹Zadania krajského kola boli distribuované tesne pred Svetovým pohárom v Jasnej 2024. Peti držíme palce aj naďalej, želáme jej skoré uzdravenie a následný úspešný návrat do svetovej špičky.



A-II-2 Jeden reverz

Máme pole $A[0..n-1]$. Toto pole obsahuje práve raz každé z čísel od 0 po $n-1$.

Upratanosť poľa je rovná počtu indexov i , pre ktoré platí $A[i] = i$.

Čoskoro príde na návštevu Inšpektor polí a my by sme mu radi ukázali čo najviac upratané pole.

Je zjavné, že najväčšia možná upratanosť poľa je n a že dosiahnuť ju vieme tak, že pole A usporiadame vzostupne.

Na to však bohužiaľ nemáme čas: kým príde Inšpektor, stíhame už spraviť len jeden jediný reverz – teda zvoliť si nejaký súvislý úsek poľa A a obrátiť poradie prvkov v ňom.

Súťažná úloha

Napíšte program, ktorý načíta popis poľa A a spomedzi všetkých možných reverzov nájde ten, ktorým z poľa A vyrobíme pole s najväčšou možnou upratanosťou.

V prvom riadku vstupu je číslo n , zvyšok vstupu tvoria postupne hodnoty $A[0]$, $A[1]$, až $A[n-1]$.
(Je zaručené, že tieto hodnoty tvoria permutáciu čísel od 0 po $n-1$.)

Na výstup vypíšete dve čísla $i \leq j$: index prvého a posledného prvku v úseku, ktorý treba reverznúť.
(Ak existuje viacero optimálnych riešení, môžete vypísať ľubovoľné z nich.)

Obmedzenia a hodnotenie

Plný počet bodov môžu získať len riešenia s lineárnou časovou zložitostou.

Iné riešenia efektívne pre $n \leq 200\,000$ môžu získať nanajvýš 9 bodov.

Najviac 6 bodov môžu získať riešenia efektívne pre $n \leq 7\,000$.

Najviac 3 body môžu získať riešenia efektívne pre $n \leq 300$.

Príklady

vstup

```
7
5 4 3 2 1 0 6
```

výstup

```
0 5
```

Keď reverzneme prvých šesť prvkov tohto poľa, dostaneme usporiadané pole. Toto je zjavne optimálne.

vstup

```
7
6 5 2 4 3 1 0
```

výstup

```
0 6
```

Reverznutím celého poľa dostaneme pole $(0, 1, 3, 4, 2, 5, 6)$. Jeho upratanosť je 4: hodnoty 0, 1, 5, 6 sú na správnych miestach. Všimnite si, že hodnotu 2 tento reverz „pokazil“ – presunul ju preč zo správneho miesta.

vstup

```
7
5 6 0 1 2 3 4
```

výstup

```
2 6
```

Pre tento vstup žiadny reverz nevie vyrobiť pole s upratanosťou väčšou ako 1, preto je správnou odpoveďou ľubovoľný z reverzov, ktoré vyrobí pole s upratanosťou 1. V našom príklade sme si vybrali reverz úseku na indexoch 2 až 6. Ten vyrobí pole $(5, 6, 4, 3, 2, 1, 0)$. Upratanosť tohto poľa je 1, lebo hodnota 3 sa nachádza na indexe 3.

vstup

```
7
0 1 2 3 4 5 6
```

výstup

```
4 4
```

Toto pole je už optimálne, najlepším riešením je nič nepokaziť – teda reverznúť nejaký úsek dĺžky 1.



A-II-3 Ťažký robot skáče

Bludisko pre robota je tvorené $r \times s$ štvorcovými políčkami. Na pláne bludiska sú riadky očíslované od 0 po $r - 1$ zhora dole a stĺpce od 0 po $s - 1$ zľava doprava. Niektoré políčka v bludisku sú prekážky (značíme 'X'), ostatné sú voľné (značíme bodkou).

Robot začína v ľavom hornom rohu plánu bludiska. Tvojou úlohou je na čo najmenší počet akcií dosiahnuť, aby robot stál v pravom dolnom rohu. Existujú dva typy akcií: kroky a skoky.

- Pri kroku si robot vyberie jeden zo štyroch základných smerov a pohne sa ním na susedné políčko. (Samozrejme, krok môže spraviť len vtedy, ak vedie na voľné políčko.)
- Pri skoku si robot tiež vyberie jeden zo štyroch základných smerov, ale tentokrát si taktiež vyberie vzdialenosť $d > 1$. Následne vyskočí zvoleným smerom zo svojho súčasného políčka, preskočí ponad $d - 1$ políčok (tie môžu byť ľubovoľného typu) a dopadne na d -te (to musí byť prázdne).

Skoky však majú aj jednu nevýhodu: náš ťažký robot má veľkú hybnosť, a tak nevie po skoku hneď zastáť. Po **každom skoku** preto musí nasledujúci pohyb (či už je to krok alebo skok) viesť **tým istým smerom**. Pripomíname, že na konci má robot v cieľi ostať stáť, nesmie tam teda prísť skokom.

Súťažná úloha

Napište program, ktorý načíta rozmery a mapu bludiska a zistí, či vieme robota dostať do cieľa a ak áno, na aký najmenší počet akcií to vieme spraviť.

V prvom riadku vstupu sú rozmery bludiska: čísla r a s . Zvyšok vstupu tvorí mapa bludiska: r riadkov, v každom z nich s znakov 'X' a '.'. Je zaručené, že prvý a posledný z týchto znakov (štart a cieľ robota) sú '.'.

Na výstup vypíšete minimálny počet akcií, resp. -1 , ak riešenie neexistuje.

Obmedzenia a hodnotenie

Plný počet bodov môžu získať riešenia efektívne pre $r, s \leq 10\,000$.

Najviac 8 bodov môžu získať riešenia efektívne pre $r, s \leq 1000$.

Najviac 5 bodov môžu získať riešenia efektívne pre $r, s \leq 50$.

Ľubovoľne pomalé korektné riešenie môže získať aspoň 3 body.

Príklady

vstup

```
4 7
..XXXXX
X..XXXX
XX..XXX
XXX....
```

výstup

```
8
Spravíme kroky vpravo, dole, vpravo, dole, vpravo,
dole, potom skok o 2 doprava a potom krok doprava.
```

vstup

```
1 5
.XXX.
```

výstup

```
-1
Skočiť o štyri doprava nesmieme, lebo by sme nevedeli
spraviť následne ďalší pohyb tým istým smerom.
```

vstup

```
4 7
..XXX..
X..XXXX
XX..X.X
XXX....
```

výstup

```
6
Začneme skokom doprava o päť a následným krokom
doprava, čím zastaneme v pravom hornom rohu. Ďalej
spravíme krok späť doľava, odtiaľ skok o dva dodola a
po ňom zabrzdíme ďalším krokom dodola. Ešte krok
doprava a sme v cieľi.
```



A-II-4 O Vekslákbotovi a Pokladničke

K tejto úlohe patrí študijný text uvedený nižšie. Je identický so študijným textom v zadaniach domáceho kola.

Jednotlivé podúlohy sú hodnotené nezávisle, môžete ich riešiť v ľubovoľnom poradí. Pri hodnotení úloh krajského kola budeme prihliadať len na korektnosť vašich programov – na ich efektívnosť (časovej zložitosti) nám nebude záležať. Nezabudnite okrem samotného programu uviesť aj jeho slovný popis, vrátane zdôvodnenia správnosti.

Podúloha A (2 body): prefarbi na maximum

V Pokladničke je na začiatku $c \geq 0$ červených, $m \geq 0$ modrých a $z \geq 0$ zelených žetónov (a nič iné), pričom je zaručené, že tieto počty **sú navzájom rôzne**. Napíšte program, po skončení ktorého bude v Pokladničke presne $c + m + z$ žetónov, pričom všetky musia mať tú farbu, ktorej bolo na začiatku najviac.

Podúloha B (3 body): prefarbi na minimum

Začiatkový stav Pokladničky je rovnaký ako v Podúlohe A. Napíšte program, po skončení ktorého bude v Pokladničke presne $c + m + z$ žetónov, pričom všetky musia mať tú farbu (červenú, zelenú alebo modrú), ktorej bolo na začiatku **najmenej**.

Podúloha C (5 bodov): mocnina troch

Na začiatku je v Pokladničke $c \geq 0$ červených žetónov (a nič iné). Napíšte program, po ktorého skončení bude červených žetónov v Pokladničke presne 3^c . (Navyše v nej môže byť aj ľubovoľne veľa žetónov iných farieb.)

Upozorňujeme, že váš program musí skončiť – pre každé c sa po konečnom počte krokov musí dostať do situácie, v ktorej sa už nedá vykonať žiadna z jeho inštrukcií.

Študijný text: O Vekslákbotovi a Pokladničke

Za 111 horami a 111 dolinami leží jedna prazvláštna krajina. V tejto krajine žijú samí roboti a ako platidlá používajú žetóny všetkých možných farieb od výmyslu sveta. V domčeku na úpätí ďalšej hory si tam spolu nažívajú dvaja hrdinovia nášho príbehu: roboti Vekslákbót a Pokladnička.

Ako možno tušíte z jej mena, Pokladnička v sebe rada uskladňuje všetky možné žetóny. Ako zrejme netušíte z jej mena, Pokladnička tiež veľmi rada vykonáva rôzne programy. A ako ste si po prečítaní predchádzajúcej vety úplne istí (koniec koncov, toto je študijný text Olympiády v informatike a nie len taká rozprávka), práve tieto programy pre ňu budete písať vy ako riešenia súťažných úloh.

Vekslákbót je majster vo vymieňaní žetónov: či už chcete vymeniť dva červené za tri modré alebo žltý a čierny za miliónpäť sivých, Vekslákbót určite pozná robota, ktorý pozná robota, ktorý s ním práve takú výmenu rád spraví. Vekslákbót má veľmi rád Pokladničku, a tak keď ho ona o hocijakú výmenu poprosí, okamžite (alebo, ako hovoríme my, v konštantnom čase) jej ju zabezpečí.

Podme sa teda pozrieť na to, ako vlastne budú vyzeráť programy pre našu Pokladničku.

Základy: vstup, výstup, program

„Vstupom“ pre Pokladničku budú jednoducho žetóny, ktoré má na začiatku v sebe. Neexistuje nijaký výstup. V zadaní rôznych úloh budeme rôzne definovať ciele, ktoré majú vaše programy dosiahnuť.

Program pre Pokladničku sa skladá z dvoch častí: *obmedzení*, ktoré musí dodržať, a *pokynov*, ktoré má vykonávať.

Obmedzenia

Prvou časťou programu pre Pokladničku je **konečná množina** obmedzení. (Táto množina môže byť aj prázdna.) Obmedzenia pre Pokladničku hovoria, najviac koľko žetónov konkrétnej farby, prípadne kombinácií žetónov rôznych farieb, smie mať naraz v sebe.

Formálne, obmedzenia sú lineárne nerovnosti nasledovného tvaru:

$$k_1 \cdot \text{farba}_1 + k_2 \cdot \text{farba}_2 + \dots + k_n \cdot \text{farba}_n \leq \text{limit}$$



pričom všetky koeficienty k_i aj limit sú konkrétne kladné celé čísla, zatiaľ čo farba_i sú premenné označujúce počet žetónov príslušnej farby v Pokladničke.

Príkladmi obmedzení sú napríklad nerovnosti „modrá ≤ 7 “ a „modrá + 2 · červená ≤ 3 “.

Ak sa nejakej farby netýka žiadne obmedzenie, môže byť v Pokladničke ľubovoľne veľa žetónov tejto farby.

Jedna inštrukcia

Každá inštrukcia pre Pokladničku má nasledovný tvar:

$$k_1 \cdot \text{farba}_1, \dots, k_n \cdot \text{farba}_n \rightarrow \ell_1 \cdot \text{farba}_{n+1}, \dots, \ell_m \cdot \text{farba}_{n+m}$$

Všetko naľavo od \rightarrow budeme volať ľavá strana inštrukcie, všetko napravo zase pravá strana.

Všetky k_i aj ℓ_j musia byť kladné celé čísla. V rámci ľavej aj v rámci pravej strany inštrukcie musia byť všetky použité farby rôzne. Je povolené použiť tú istú farbu na oboch stranách inštrukcie. Je povolené mať $n = 0$ alebo $m = 0$, teda inštrukciu, ktorá má niektorú stranu prázdnu.²

Príklady inštrukcií:

- 2 červená \rightarrow 3 modrá
- 1 žltá, 1 čierna \rightarrow 1 000 005 sivá
- 3 červená \rightarrow 333 červená, 334 cyklámenová, 335 purpurová
- 2 zelená \rightarrow \emptyset

Posledná inštrukcia z príkladu má prázdnu pravú stranu. Na zápis prázdnej strany inštrukcie budeme používať symbol prázdnej množiny, aby bolo jasné, že je prázdna úmyselne.³

V našich príkladoch budeme používať reálne názvy farieb. Vo svojich programoch môžete ako názvy farieb používať aj ľubovoľné iné alfanumerické reťazce. Je tiež OK pri zápise vynechať koeficient 1. Druhú z vyššie uvedených inštrukcií teda môžete zapísať aj „žltá, čierna \rightarrow 1 000 005 sivá“.

Pokladnička vykoná inštrukciu tak, že zo seba vyberie sadu žetónov na ľavej strane inštrukcie, dá ich Vekslákbotovi a poprosí ho, aby jej za ne priniesol sadu žetónov z pravej strany. Vekslákbot to samozrejme promptne zabezpečí a Pokladnička do seba vloží žetóny, ktoré jej priniesol.

Ak v sebe Pokladnička nemá všetky žetóny, ktoré si vyžaduje ľavá strana inštrukcie, tak danú inštrukciu v danej chvíli vykonať nevie. Ak by napríklad mala len dva červené žetóny, nevie vykonať inštrukciu „3 červené \rightarrow 333 červených“.

Pokladnička tiež nevie vykonať inštrukciu, ak by po jej vykonaní bolo porušené hociktoré z obmedzení.

Pokyny

Pokyny pre Pokladničku tvorí **konečná postupnosť** inštrukcií vyššie uvedeného typu. Zdôrazňujeme, že ide o postupnosť, a teda **záleží** na poradí inštrukcií.

Vykonávanie programu

Program sa vykonáva v krokoch. V každom kroku Pokladnička začne čítať pokyny od začiatku a číta ich, až kým nenájde prvú inštrukciu, ktorú vie momentálne vykonať. Tú inštrukciu vykoná. (Na zvyšok pokynov sa v tomto kroku už ani nepozrie. V ďalšom kroku začne znova čítať postupnosť inštrukcií od začiatku.)

Vykonávanie programu skončí, keď sa už žiadna inštrukcia v pokynoch nedá vykonať.

Príklad #1: viac červených

Úloha: Na začiatku sú v Pokladničke nejaké červené a nejaké modré žetóny. Napíšte program, po ktorého skončení bude v Pokladničke práve jeden zlatý žetón a nič iné, ak bolo červených žetónov viac ako modrých. Vo všetkých ostatných prípadoch musí Pokladnička skončiť úplne prázdna.

²Povolené je mať prázdne aj obe strany inštrukcie, ale ako sa čoskoro dozvieme, program, v ktorom takúto inštrukciu použijeme, zaručene nikdy neskončí.

³Formálne môžeme povedať, že aj pravá aj ľavá strana každej inštrukcie obsahuje nejakú multimnožinu farieb. Prázdna strana inštrukcie je teda naozaj prázdna multimnožina.



Riešenie 1: bez obmedzení

Nebudeme mať žiadne obmedzenia. Pokyny budú vyzeráť nasledovne:

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

Pri vykonávaní tohto programu bude Pokladnička najskôr používať inštrukciu 1, kým to ide. Keď to prestane ísť, má už v sebe buď len červené alebo len modré žetóny. Ak sú len modré, jediná inštrukcia, ktorá sa dá používať, je inštrukcia 4. Jej opakovaným použitím sa Pokladnička vyprázdni a program skončí.

Ak sú po skončení používania inštrukcie 1 v Pokladničke len červené žetóny, bude to o čosi komplikovanejšie. Jediná inštrukcia, ktorá sa dá v tejto situácii použiť, je inštrukcia 3: vymeníme jeden červený žetón za jeden zlatý. Od tejto chvíle sa už inštrukcia 3 nepoužije, a to preto, že už sa dá používať inštrukcia 2. Tou sa Pokladnička postupne zbaví zostávajúcich červených žetónov. Akonáhle v nej ostane len samotný zlatý žetón, už sa nedá použiť žiadna inštrukcia, a teda výpočet končí.

Riešenie 2: s obmedzením

Budeme mať jedno obmedzenie:

- zlatá ≤ 1

Pokyny budú vyzeráť nasledovne:

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

Tentokrát sa v situácii, kedy sú v Pokladničke len samé červené žetóny, najskôr raz vykoná inštrukcia 2 (jeden vymeníme za zlatý) a potom sa už bude používať inštrukcia 3, až kým sa všetky červené neminú. Obmedzenie, ktoré sme si zvolili, totiž bráni Pokladničke opakovane využiť inštrukciu 2.

Príklad #2: súčet

Úloha: Na začiatku je v Pokladničke $c > 0$ červených žetónov, $m > 0$ modrých a jeden zelený. Napíšte program, po ktorého skončení bude v Pokladničke presne c červených, m modrých a $c + m$ fialových žetónov.

Riešenie. Keby sme len chceli dostať $c + m$ fialových žetónov, stačilo by vymeniť všetky červené aj všetky modré za fialové „s kurzom jedna k jednej“. Ako ale nestratiť pôvodné žetóny?

Naše riešenie nebude mať žiadne obmedzenia a bude mať nasledujúce inštrukcie:

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

Všimnite si, ako náš program používa prítomnosť zeleného a žltého žetónu v Pokladničke: pomocou nich vieme rozlíšiť, či ešte premieňame pôvodné červené a modré žetóny alebo či už naspäť vznikajú nové.

TRIDSATY DEVIATY ROČNÍK OLYMPIÁDY V INFORMATIKE

Príprava úloh: Michal Anderle, Michal Forišek

Recenzia: Michal Forišek

Slovenská komisia Olympiády v informatike

Vydal: NIVAM – Národný inštitút vzdelávania a mládeže, Bratislava 2024