Priebeh krajského kola

Krajské kolo 40. ročníka Olympiády v informatike, kategória B, sa koná 21. 1. 2025 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?

- Slovne popíšte algoritmus.
  Slovný popis riešenia musí byť jasný a zrozumiteľný i bez nahliadnutia do samotného algoritmu/programu.
- Zdôvodnite správnosť vášho algoritmu.
- Uveďte 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.

B-II-1 Byrokracia

Pozor, aj keď sa zadanie veľmi podobá prvej úlohe v domácom kole, ide o inú úlohu.

Vybavovanie žiadostí na ministerstve byrokracie je vždy náročný a zdĺhavý proces. Vyzerá napríklad takto:

Kancelária 4: *„Ak sa chcete registrovať do portálu, musíte mať identifikačné číslo, dajú vám ho v kancelárii 8.“*
Kancelária 8: *„Ak chcete identifikačné číslo, doneste potvrdenie o správnosti osobných údajov z kancelárie 2.“*
Kancelária 2: *„Potvrdenie vám vydáme, ale najskôr si podajte žiadosť o overenie údajov v kancelárii 5.“*
Kancelária 5: *„Nie je problém. Aké máte identifikačné číslo? Že nemáte? Pridelia vám ho v kancelárii 8.“*
Kancelária 8: *„Ak chcete identifikačné číslo, doneste potvrdenie o správnosti osobných údajov z kancelárie 2.“*
...

Ministerstvo má n kancelárií, očíslovaných od 1 po n. Človek môže svoju návštevu ministerstva začať v ľubovoľnej z nich. Každá kancelária má jednoznačne určenú jednu ďalšiu kanceláriu, kam posielajú všetkých ľudí: vždy, keď príde človek do kancelárie s číslom i, pošlú ho do kancelárie a_i.
Na takto fungujúcom ministerstve sa každému človeku stane, že časom navštívi niektorú kanceláriu druhýkrát. Zaujímalo by nás, koľko krokov pred tým spraví.
Pre každú kanceláriu zistite jedno číslo – keď človek začne návštevu ministerstva v tejto kancelárii, koľko kancelárií navštívi pred tým, ako sa v niektorej kancelárii ocitne druhýkrát?

Formát vstupu a výstupu

Na prvom riadku vstupu je jedno celé číslo n – počet kancelárií na ministerstve. Na druhom riadku je n medzerou oddelených čísel $a_1 \dots a_n$, popisujúcich, do ktorej kancelárie nás pošlú z kancelárie číslo i ($1 \leq a_i \leq n$).

Na výstup vypíšte n celých čísel, pričom i-te z nich sa musí rovnať počtu kancelárií, ktoré návštevník ministerstva začínajúci v kancelárii s číslom i navštívi skôr, než navštívi niektorú kanceláriu, v ktorej už bol.

Obmedzenia a hodnotenie

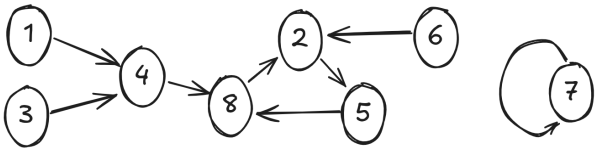
Plný počet bodov môžu získať riešenia efektívne pre $n \leq 2\,000\,000$.
Najviac 8 bodov môžu získať riešenia efektívne pre $n \leq 200\,000$.
Za ľubovoľné funkčné riešenie viete získať aspoň 3 body.

Čiastočné body môžete získať za riešenia, ktoré fungujú **za dodatočného predpokladu**, že na ministerstve existuje práve jeden cyklus. Ak riešite túto verziu úlohy, napíšte to na začiatok svojho popisu.
Najviac 5 bodov môžu získať riešenia tejto jednoduchšej úlohy ktoré sú efektívne pre $n \leq 2\,000\,000$.
(Cyklus je taká postupnosť $p \geq 1$ kancelárií $k_1, k_2 \dots k_p$, pre ktorú platí, že $a_{k_1} = k_2$, $a_{k_2} = k_3$, ..., $a_{k_{p-1}} = k_p$ a následne $a_{k_p} = k_1$. Cykly považujeme za rôzne, ak sú tvorené rôznou množinou kancelárií.)

Príklad

vstup

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



výstup

```
5 3 5 4 3 4 1 3
```

Na obrázku vľavo môžeme vidieť schematický nákres ministerstva zodpovedajúceho vstupu z príkladu.

*Napríklad si všimnime, že ak návštevník začne v kancelárii **6**, postupne navštívi 6, 2, 5, 8, 2, čiže navštívi **štyri** kancelárie pred tým, ako sa zopakuje kancelária číslo 2. Preto je **šiestou** hodnotou na výstupe číslo **4**.*

B-II-2 Asfaltovanie

Lukáš bol vyslaný Kocúrkovskou Spoločnosťou Prepravy skontrolovať opravu hlavnej cesty. Rekonštrukcia cesty podlieha veľmi prísnym reguláciám a Lukášovou úlohou je skontrolovať, že tieto regulácie neboli porušené.
Hlavnú cestu si môžeme predstaviť ako dlhú úsečku rozdelenú na n po sebe idúcich úsekov. Do výzvy na opravu sa zapojilo k firiem. Každá vyhrala **práve jednu** zákazku. Zákazka určuje firme niekoľko **po sebe idúcich** úsekov, ktoré musí nanovo vyasfaltovať. Zákazky sa môžu prekrývať – ten istý úsek cesty môže byť postupne opravený aj viackrát. Platí však, že každý úsek cesty dostala vyasfaltovať aspoň jedna firma.
Regulácie opráv určujú, že firmy musia pracovať *postupne* a celú svoju zákazku spraviť *naraz*. To znamená, že celá rekonštrukcia musí prebehnúť v k fázach, pričom v každej fáze musí jedna firma kompletne vyasfaltovať jej pridelenú postupnosť úsekov. Poradie, v ktorom firmy plnia svoje zákazky, môže byť ľubovoľné.
Lukáš prišiel na inšpekciu po dokončení všetkých zákaziek. Zo zloženia asfaltu na každom úseku cesty zistil, ktorá firma ho opravovala ako posledná. Teraz Lukáša zaujíma, či z týchto údajov nevyplýva porušenie regulácií.

Na vstupe dostanete zadaný počet úsekov cesty n a počet firiem k, ktoré vyhrali zákazku. Navyše, pre každý úsek dostanete informáciu o poslednej firme, ktorá ho asfaltovala. Vašou úlohou je určiť, či takéto vyasfaltovanie mohlo vzniknúť dodržiac platné regulácie.

Formát vstupu a výstupu

Na prvom riadku vstupu sú dve celé kladné čísla n a k – počet úsekov a počet firiem.
Na druhom riadku je n medzerou oddelených čísel $a_1, a_2, \ldots, a_n$. Hodnota a_i je číslo firmy, ktorá úsek i opravovala ako posledná. Pre všetky čísla firiem platí, že $1 \leq a_i \leq k$. (Tieto čísla slúžia len na označenie firiem. Nepredpokladajte teda, že firma s menším číslom asfaltovala skôr alebo neskôr ako firma s väčším číslom.)
Na výstup vypíšte reťazec „`ANO`“ ak dané vyasfaltovanie mohlo vzniknúť v súlade s reguláciami, resp. reťazec „`NIE`“ ak regulácie nutne porušilo.

Obmedzenia a hodnotenie

Plný počet bodov získa riešenie, ktoré efektívne vyrieši ľubovoľný vstup, v ktorom $n \leq 10^6$ a $k \leq 10^6$.
Nanajvýš 5 bodov viete získať za riešenia efektívne pre $n \leq 1000$.
Za ľubovoľné správne riešenie môžete získať aspoň 3 body.

Ak neviete riešiť plnú verziu súťažnej úlohy, môžete ju vyriešiť aspoň za dodatočného predpokladu, že $k = n/2$ a medzi hodnotami a_1 až a_n sa číslo každej firmy nachádza **práve dvakrát**.
Ak riešite túto jednoduchšiu verziu úlohy, napíšte to na začiatok svojho popisu. Za jej riešenie efektívne pre $n \leq 10^6$ môžete dostať nanajvýš 7 bodov, za pomalšie riešenia primerane menej.

Príklady

vstup

```
9 5
1 1 4 4 2 2 5 4 1
```

výstup

```
ANO
```

Existujú viaceré poradia asfaltovania, po ktorých vznikne takto vyzerajúca cesta. Napr. je možné, že najskôr bol aplikovaný asfalt firmy 1 na celú cestu, potom asfalt firmy 3 na úseky 4 až 6, potom asfalt firmy 4 na úseky 3 až 8, ďalej asfalt firmy 2 na úseky 5 a 6, a nakoniec asfalt firmy 5 na úsek 7. (Všimnite si, že na výslednej ceste už nie je žiaden asfalt firmy 3, hoci asfaltovala ako druhá v poradí.)

vstup

```
8 4
3 2 2 3 4 1 4 1
```

výstup

```
NIE
```

Tento výsledný stav cesty nemohol vzniknúť podľa regulácií. Ak by firma 1 vykonala svoju zákazku skôr ako firma 4, musela by potom firma 4 prekryť asfalt firmy 1 na úseku 6. A naopak, ak by firma 4 opravovala cestu skôr ako firma 1, musela by firma 1 preasfaltovať úsek 7.
(Tento vstup je navyše príkladom vstupu, v ktorom sa každé číslo firmy vyskytuje práve dvakrát.)

B-II-3 Pečieme makrónky

Kubík chce pripraviť n dokonalých okrúhlych makrónok. Makrónka je koláč, ktorý pozostáva z dvoch nadýchaných piškótových škrupiniek (ktoré budeme volať polmakrónky) spojených čokoládovou ganážou.
V prvom kroku upiekol $2n$ makrónkových polovičiek. Keď ich však šiel spájať, všimol si, že zďaleka nie sú všetky ani kruhové, ani rovnakej veľkosti. Keby ich pospájal len tak, hala-bala, nevznikli by mu také estetické makrónky ako by chcel. Musí ich pospájať lepšie!
Pre každú dvojicu polmakrónok si Kubík určil, či ich vôbec má zmysel lepiť dokopy, a ak áno, ako pekne by spolu vyzerali. Teraz húta, ako ich čo najlepšie pospájať. Pomôžte mu!

Súťažná úloha

Kubík má $2n$ polmakrónok, ktoré si očísloval od 1 po $2n$. Pre každé $1 \leq i, j \leq 2n$ Kubík určil, že krása makrónky získanej spojením i-tej a j-tej polmakrónky je $k_{i,j}$. Ak je $k_{i,j} = -1$, znamená to, že tieto dve polmakrónky v žiadnom prípade spojiť nemôže.
Kubík by chcel vyrobiť n makrónok tak, aby **súčet** krás týchto makrónok bol najväčší možný. Samozrejme, každá polmakrónka sa musí použiť **v práve jednej** makrónke.

Prezradíme vám, že síce táto úloha má efektívne riešenia[1], tie sú tak komplikované, že neočakávame, že na nejaké z nich prídete. Namiesto toho nám, podobne ako v domácom kole, bude stačiť *šikovné* riešenie hrubou silou – teda výskúšaním *takmer* všetkých možností.
Vaše riešenia budú hodnotené práve podľa počtu možností, ktoré budú potrebovať vyskúšať.

Formát vstupu a výstupu

Na prvom riadku vstupu je číslo n – počet výsledných makrónok.
Na každom z ďalších $2n$ riadkoch sa nachádza $2n$ čísel. Presnejšie, i-ty z týchto riadkov obsahuje čísla $k_{i,1}, \ldots, k_{i,2n}$ oddelené medzerou.
Je zaručené, že platí $k_{i,i} = -1$, keďže žiadnu polmakrónku nevieme spojiť samu so sebou. Navyše pre každú dvojicu polmakrónok platí, že $k_{i,j} = k_{j,i}$.

Ak neexistuje žiaden spôsob ako polmakrónky pospájať do n makrónok, vypíšte reťazec „`neda sa`“. V opačnom prípade nájdite ľubovoľné pospájanie, ktoré maximalizuje celkovú krásu vytvorených makrónok a toto pospájanie vypíšte. Na prvý riadok výstupu vypíšte celkovú krásu. Na každý zo zvyšných n riadkov výstupu vypíšte dvojicu čísel – čísla polmakrónok, ktoré tvoria i-tu makrónku.

Hodnotenie

Váš program bude primárne hodnotený podľa toho, koľko možností vyskúša v najhoršom prípade. Vo vašom riešení nezabudnite uviesť čo najpresnejší odhad tohto počtu v závislosti od hodnoty n.

Pre zopakovanie pripomíname, že číslo $n!$ *(*n *faktoriál) označuje súčin* $n \cdot (n-1) \ldots 2 \cdot 1$*. Pri používaní faktoriálov si dajte pozor na uzátvorkovanie. Hodnota* $(2n)!$ *sa rovná* $2n \cdot (2n-1) \ldots 2 \cdot 1$*, avšak hodnota* $2n!$ *je rovná iba číslu* $2 \cdot n! \; = \; 2 \cdot (n \cdot (n-1) \ldots 2 \cdot 1)$*.*

Riešenie, ktoré v najhoršom prípade vyskúša $(2n)!$ možností vie získať 3 body.
Riešenie, ktoré v najhoršom prípade vyskúša $(2n-1)!$ možností vie získať až 5 bodov.
Lepšie riešenia vedia získať až 9 bodov na základe toho, ako efektívne sú. Vzorové riešenie vyskúša pre $n = 5$ zhruba 1000 možností.

Posledný bod viete získať za také vylepšenia programu, ktoré mu umožnia *niekedy* (pre vhodné hodnoty na vstupe) vyskúšať aj menší počet možností pospájania makrónok, hoci v najhoršom prípade sa počet skúšaných možností nezmení.

[1] t.j. bežiace v polynomiálnom čase

Aj keď sa jedná o teoretickú úlohu, nezabudnite napísať aj **pseudokód** implementácie vášho algoritmu (v ľubovoľnom zrozumiteľnom programovacom jazyku). Táto časť je hodnotená približne polovicou bodov, pričom dôraz bude kladený na implementáciu hlavných myšlienok.

Príklady

Poznámka: Vo všetkých príkladoch vstupu sme hodnoty $k_{i,j}$ naformátovali pridaním dodatočných medzier do lepšie čitateľnej tabuľky.

vstup

```
2
-1  2  1 10
 2 -1 -1  5
 1 -1 -1  1
10  5  1 -1
```

výstup

```
6
1 3
2 4
```

V tomto prípade chce Kubík vytvoriť dve makrónky zo štyroch polmakrónok. Druhú polmakrónku vie spojiť len s prvou alebo štvrtou polmakrónkou. Ak ju spojí s prvou polmakrónkou, musí spojiť tretiu a štvrtú, čím dostane celkovú krásu $2 + 1 = 3$*. Ak ju ale spojí so štvrtou polmakrónkou, prvú polmakrónku spojí s treťou a dostane lepšiu celkovú krásu* $5 + 1 = 6$*.*

vstup

```
2
-1 10 10 10
10 -1 -1 -1
10 -1 -1 -1
10 -1 -1 -1
```

výstup

```
neda sa
```

Pre tento vstup neexistuje žiadny spôsob ako vytvoriť 2 makrónky tak, aby bol Kubík spokojný. Akonáhle sa prvá polmakrónka spojí s inou, zvyšné dve sa k sebe vôbec nehodia.

vstup

```
3
-1  1 -1 -1 -1  2
 1 -1  1 -1 -1 -1
-1  1 -1  2 -1 -1
-1 -1  2 -1  1 -1
-1 -1 -1  1 -1  1
 2 -1 -1 -1  1 -1
```

výstup

```
4
1 2
3 4
5 6
```

V tomto prípade má Kubík len dva spôsoby ako makrónky pospájať. Oba majú rovnakú celkovú krásu. Správnym riešením by teda bolo aj pospájanie $(1, 6)$*,* $(2, 3)$ *a* $(4, 5)$*.*

B-II-4 Opravujúce sa svetielkové kódovanie

Rekapitulácia domáceho kola

V domácom kole ste sa stretli s dvoma kamarátkami, Alicou a Hankou, ktoré riešili zapeklitý problém. Dievčatá sa chceli dohodnúť na čase stretnutia, kvôli rozdielnym rozvrhom sa však nemohli porozprávať. Informáciu o čase si preto posielali pomocou `micro:bitu` v učebni informatiky.
Na komunikáciu používali *štvorcovú* doštičku LED svetielok. Doštička je mriežka s n riadkami a n stĺpcami, pričom v každom políčku je jedno svetielko, ktoré sa dá rozsvietiť nezávisle od ostatných.
Alica s Hankou sa dopredu dohodli na spôsobe, akým budú kódovať rôzne časy stretnutia. Alica, ktorá mala hodinu informatiky ako prvá, naprogramovala doštičku $n \times n$ tak, aby sa rozsvietila špecifickým spôsobom. Hanka si ju potom zapla na svojej hodine a podľa dohodnutého spôsobu z doštičky prečítala čas stretnutia.

Príklady jednoduchého kódovania
Predstavte si, že sa Alica s Hankou chcú stretnúť o nejakej presnej minúte medzi 16:00 a 16:20. V takomto prípade by sa mohli vopred dohodnúť napríklad tak, že Alica použije doštičku 5×5 a naprogramuje ju tak, aby sa na nej rozsvietilo x svetielok, kde x je minúta, o ktorej sa majú stretnúť. Hanka potom po zapnutí doštičky jednoducho spočíta, koľko svetielok na doštičke svieti, a podľa toho určí čas stretnutia.
Všimnite si, že väčšinu časov vie Alica zakódovať veľa rôznymi spôsobmi, keďže pri vyššie popísanom kódovaní vôbec nezáleží na tom, ktoré svetielka sú zapnuté, len na ich celkovom počte. Napr. pre zakódovanie času stretnutia 16:12 by si mohla vybrať ľubovoľných 12 políčok, ktoré by sa rozsvietili.

Iný spôsob kódovania by mohol vyzerať tak, že si dievčatá svetielka na doštičke očíslujú od 0 po 24. (Toto vedia spraviť napr. po riadkoch zhora dole a v každom riadku zľava doprava – teda prvý riadok dostane čísla 0-4, druhý 5-9, a tak ďalej.) Na zakódovanie minúty x by potom Alica rozsvietila práve jedno svetielko – to, ktoré má priradené číslo x. Napr. v prípade času 16:10 by musela rozsvietiť svetielko číslo 10, teda najľavejšie v treťom riadku.

Odhalenie chyby
Doštičky, ktoré dievčatá používajú, nie sú úplne spoľahlivé. Presnejšie, stáva sa, že keď Hanka doštičku zapne, **práve jedno svetielko** sa nebude správať podľa Alicinho programu ale presne opačne – teda ak Alica chcela, aby bolo zhasnuté, tak bude Hanke svietiť, a naopak.

V domácom kole sa dievčatá snažili nájsť také kódovanie, ktoré dá Hanke možnosť zistiť, či na jej doštičke nastala chyba.
Všimnite si, že druhé kódovanie z vyššie uvedeného príkladu má túto vlastnosť. Alica vždy rozsvieti práve jedno svetielko. Ak nastane chyba, budú nutne na Hankinej mriežke svietiť buď dve svetielka alebo žiadne. Ak teda Hanka vidí práve jedno svetielko, vie, aký čas Alica poslala, a ak vidí iný počet svetielok, vie, že nastala chyba.
V domácom kole ste dievčatám pomohli nájsť efektívnejšie spôsoby kódovania, ktoré tiež mali túto istú vlastnosť: ak chyba nenastala, Hanka vedela určiť posielaný čas, a ak chyba nastala, Hanka zaručene vedela odhaliť, že sa tak stalo.

Nový problém

Dievčatá začali používať vaše šikovné riešenia z domáceho kola, rýchlo však zistili, že nie sú úplne praktické. Problém bol v tom, že ak na doštičke nastala chyba, Hanka to síce **odhalila**, ale chybu väčšinou nevedela **opraviť** – teda nevedela určiť čas, ktorý chcela Alica poslať a nemohli sa stretnúť.
(Aj druhý z vyššie uvedených príkladov má túto vlastnosť: ak Hanka vidí rozsvietených svetielok nula alebo dve, síce vie, že nastala chyba, ale vo všeobecnosti nevie, ktoré svetielko je to správne.)

Vašou dnešnou úlohou je vymyslieť nový spôsob kódovania, pri ktorom bude Hanka vedieť ľubovoľnú jednu chybu nielen odhaliť, ale aj opraviť. Presnejšie, budú nás zaujímať kódovania, ktoré majú nasledovné vlastnosti:

- Alica chce Hanke poslať presnú minútu z rozsahu **od 14:00 po 22:31** vrátane.
  Pre každý možný čas z tohto intervalu musí byť jednoznačne určené, ktoré svetielka má Alica rozsvietiť.
- Hanka musí z Alicou rozsvietených svetielok vedieť jednoznačne určiť, v ktorom čase sa majú stretnúť.
- Navyše, kódovanie musí byť *odolné voči jednej chybe*. To znamená, že aj ak sa práve jedno zo svetielok na mriežke rozsvieti opačne ako chcela Alica, Hanka bude naďalej vedieť správne určiť Alicou posielaný čas.

Všimnite si, že na kódovaní sa Alica s Hankou dohadujú dopredu, obe preto budú poznať zvolený spôsob. Navyše majú obe naozaj dobrú pamäť a vedia si zapamätať ľubovoľne komplikované pravidlá, výnimky a špecifikácie. Jediná vec, ktorú nepoznajú dopredu, je konkrétny čas, ktorý budú kódovať – vedia len, že bude aspoň 14:00 a nanajvýš 22:31.

Súťažné úlohy

a) (2 body) Dokážte, že pre zadanú úlohu neexistuje žiadne kódovanie, ktorému by stačila mriežka 3×3.

b) (8 body) Navrhnite kódovanie, ktoré bude mať všetky vyššie popísané vlastnosti.

   Vaše riešenie bude hodnotené podľa toho, ako veľkú **štvorcovú doštičku** budú musieť Alica s Hankou na kódovanie využiť. Čím menšiu doštičku bude váš navrhnutý spôsob potrebovať, tým viac bodov môžete získať.

   Plných 8 bodov môžu získať riešenia, ktorým postačuje mriežka 4×4.

   Až 5 bodov môžu získať riešenia, ktorým postačuje mriežka 5×5.

   Najviac 3 body vie získať ľubovoľné správne riešenie bez ohľadu na veľkosť mriežky, ktorú využíva.

V riešení podúlohy b) nezabudnite popísať:

- Jednoznačný spôsob, akým Alica zo zadaného času určí, ktoré svetielka sa majú rozsvietiť.
- Jednoznačný spôsob, akým Hanka zo svetielok (v ktorých je možno jedna chyba) zistí poslaný čas.

ŠTYRIDSIATY ROČNÍK OLYMPIÁDY V INFORMATIKE

Príprava úloh: Michal Anderle, Ján Hozza, Andrej Korman, Paulína Smolárová
Recenzia: Michal Forišek
Slovenská komisia Olympiády v informatike
Vydal: NIVAM – Národný inštitút vzdelávania a mládeže, Bratislava 2024