Leták letnej časti VI. ročníka

Ahojte milí riešitelia.

Sme veľmi radi, že ste sa dozvedeli o PRASKu a asi by vás zaujímalo, čo to vlastne je, ako to celé funguje a prečo by ste to mali riešiť. Na všetko z toho sa vám teraz pokúsime odpovedať.

Čo to je a pre koho je to určené?

PRASK je korešpodenčný seminár určený pre všetkých základoškolákov, ktorých zaujíma matematika, informatika alebo by sa chceli naučiť programovať. Je to súťaž zameraná hlavne pre siedmakov a starších, môžete ju však riešiť aj keď ste v nižšom ročníku.

Seminár je organizovaný študentami informatiky na Fakulte matematiky, fyziky a informatiky na Univerzite Komenského.

Priebeh súťaže

Počas roka prebiehajú dve nezávislé časti – letná a zimná. Priebeh častí je už potom úplne rovnaký. Každá časť pozostáva z dvoch sérií piatich príkladov – dvoch teoretických, jedného praktického a dvoch programátorských. Ak aj neviete programovať nezúfajte. Namiesto programátorských úloh si môžete prejsť programátorským tutoriálom, ktorý vás to naučí a naviac v ňom získate body, ktoré sa vám rátajú do PRASKu.

Na riešenie série je vyhradených niekoľko týždňov. Až do dňa odovzdania môžete doma riešiť zadané príklady. Môžete riešiť ľubovoľné príklady z danej série, nemusíte vyriešiť všetko, nemusíte vyriešiť ani celú úlohu[1]. Najneskôr do dňa odovzdania (ktorý je napísaný na zadaniach aktuálnej série) je potrebné poslať vaše riešenia pomocou webového rozhrania.

Po konci série si pozrieme vaše odovzdané riešenia a opravíme ich. Pre každý príklad je v zadaní napísané, koľko bodov sa zaň dá dostať. Samozrejme, je možné získať čiastkové body, aj keby ste nevyriešili celú úlohu, alebo by vaše riešenie nebolo úplne správne. Dokonca, ak nás prekvapíte originálnym riešením, môžete získať bonusové body. Opravené riešenie vám potom pošleme späť aj s poznámkami ohľadom vašeho riešenia.

Prečo to chcem riešiť?

Riešenie korešpodenčného seminára prináša mnoho výhod. Riešením úloh a čítaním našich vzorových riešení **objavíte a naučíte sa** mnoho nových vecí, ktoré by ste sa v škole skoro určite nenaučili. Napríklad sa môžete naučiť **programovať**. To vám potom vie **pomôcť pri prijímačkách**, či už na stredné alebo vysoké školy. Takisto vám to pomôže pri **riešení Olympiády z informatiky alebo Korešpodenčného Seminára z Programovania**. No a v neposlednom rade, pri **pohovoroch** do veľkých firiem ako Google, Facebook alebo Eset častokrát zaváži znalosť algoritmického programovania, ktoré si môžete pomocou nášho semináru trénovať.

Je tu však ešte jedna výhoda určená pre najlepších riešiteľov. Dvakrát ročne sa bude organizovať týždenné **sústredenie**. Naň pozývame niekoľko[2] najlepších riešiteľov. Na sústredení si užiješ kopec zábavy, športu, nových ľudí a možno sa aj niečo naučíš.

A samozrejme, víťazov čakajú pekné **vecné ceny** vo forme knižky, hry alebo menšej elektroniky.

Ako má vyzerať správne riešenie

To závisí od typu úlohy, ktorú riešite. Pri teoretických úlohách musí správne riešenie okrem výsledku obsahovať aj popis postupu, akým ste sa k danému výsledku dopracovali. Dôraz sa pri opravovaní dáva hlavne na tento slovný popis, ktorý by mal byť napísaný čo najzrozumiteľnejšie, aby sme si pri opravovaní nemuseli lámať hlavu. Mal by obsahovať všetky podstatné kroky, ktoré vás viedli k riešeniu.

V prípade praktických úloh sa to líši. Občas od vás chceme slovný popis, občas sa stačí dostať k nejakému tajnému heslu alebo kliknúť na správnu linku. Presný spôsob nájdete v zadaní.

[1] Aj keď budeme radi, ak sa vám to podarí.

[2] zhruba 15, ale aj nižšie umiestení riešitelia sa môžu dostať ako náhradníci

No a pri programátorských úlohách a programátorskej liahni odovzdávate iba váš program, ktorý sa vám okamžite automaticky otestuje a do pár sekúnd sa dozviete, či ste úlohu vyriešili správne. A ak nie, môžete skúsiť odovzdať opravený program znova.

A nebojte sa, ak ste ešte nikdy nespisovali postupy svojich riešení. Keď vám riešenia opravíme, napíšeme vám k nim aj komentáre, ktoré vám môžu pomôcť v riešení ďalšej série. To je najlepší spôsob, ako sa zlepšovať.

Spôsob odovzdávania

Ako prvú vec, ktorú musíte urobiť pred tým, ako budete môcť odovzdávať svoje riešenia, je **zaregistrovanie** sa na našej webovej stránke `prask.ksp.sk`. V časti `Zadania a vzoráky` nájdete okrem zadaní aj odkaz, na ktorom môžete odovzdať vaše riešenie.

Riešenie každej teoretickej úlohy má byť jeden súbor formátu **.pdf**. Ten nahráte na našu stránku a stlačíte zelené tlačítko `Submit`. Opravovať sa bude **posledné odovzdané** riešenie, takže si dajte pozor, aby ste si niečo neprepísali.

Myslím, že vytvoriť pdf súbor by pre vás nemal byť problém, ak by ste s tým predsa len problém mali, pokúste sa použiť nejaký online converter ako napríklad `www.freepdfconvert.com`.

V prípade programátorských úloh sa dá rovnakým spôsobom odovzdať zdrojový kód vášho programu, teda súbor s príponou **.cpp**, **.py** alebo **.pas**.

Úlohy 1. kola letnej časti

Termín odoslania riešení tejto série je pondelok **27. apríla 2020**.

Teoretické úlohy

V tejto časti ťa čaká niekoľko matematickejších úloh, ktoré úzko súvisia s informatikou. Ako riešenie týchto úloh treba poslať podrobne spísaný postup toho, ako si riešil danú úlohu.

A ak by ťa to zaujímalo, podobné úlohy môžeš nájsť aj v Olympiáde v informatike, kategória B (http://oi.sk). Vrelo ti ju odporúčame riešiť tiež, naučíš sa veľa nových vecí a môžeš sa dostať aj na krajské kolo Olympiády.

1. Poskakujúci Super Mário

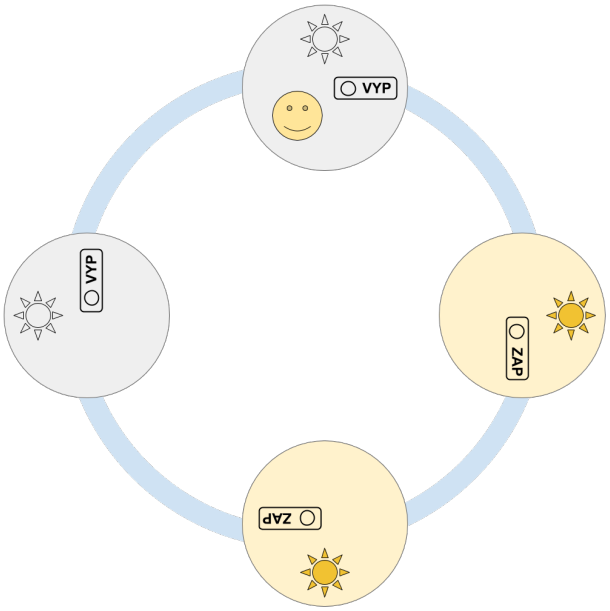
15 b popis, 0 b program

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Romanovi na roman.sobkuliak@trojsten.sk

Super Mário spokojne prežíval jeden zo svojich troch životov – jedol hríby, poskakoval medzi stromami, a naháňal sa za princeznou. Pri jednom skoku sa ale pošmykol a spadol do veľkej zelenej rúry. Keď precitol, zistil, že sa nachádza v nevedno akom veľkom *systéme miestností*.

Tento systém je tvorený niekoľkými navzájom nerozoznateľnými miestnosťami usporiadanými do kruhu. Z každej miestnosti vedú dve prechodové komory do susedných miestností. Okrem toho je tam už len jedno stropné svetlo a vypínač, ktoré toto svetlo ovláda. Pomocou neho vie Mário v danej miestnosti vypnúť alebo zhasnúť svetlo.

Na obrázku nižšie je jednoduchý systém so štyrmi miestnosťami. Mário sa nachádza v hornej z nich a v miestnosti je aktuálne zhasnuté.



Mário by rád zistil, v akom rozsiahlom systéme sa nachádza, teda určil počet miestností, ktoré ho tvoria. Má však niekoľko problémov. Síce zapnutie a vypnutie svetiel ovláda iba Mário pomocou vypínačov v miestnosti, netuší ako sú tieto svetlá rozsvietené na začiatku. V niektorých miestnostiach sú zapálené, v iných nie. No najmä, napriek tomu že je Super, si nevie v miestnostiach nechávať žiadne značky ani ich iným spôsobom označovať. Je teda naozaj odkázaný iba na zapínanie a vypínanie svetiel.

Úloha

Mário môže robiť dva typy akcií: prechádzať dverami medzi skrýšami a rozsvecovať alebo zhasínať svetlo v jednotlivých skrýšach. Vašou úlohou je pomôcť mu s nasledujúcimi problémami.

a) (2 body) Vymyslite spôsob, ktorým Mário zistí, či je v systéme práve jedna miestnosť alebo ich je viac. Ak je systém miestností tvorený iba jednou miestnosťou, tak jediná prechodová komora spája jedny jej dvere s druhými. Ak teda Mário opustí miestnosť jedným koncom, vojde druhým koncom do tej istej miestnosti.

b) (3 body) Nájdite ľubovoľný spôsob, ktorým Mário zistí, či sú miestnosti aspoň tri.

c) (6 body) Nájdite ľubovoľný postup, ktorým vie Mário zistiť počet miestností v systéme, bez ohľadu na to, aký je veľký.

d) (4 body) Označme si počet miestností v systéme ako n (ktoré Mário stále nepozná). Nájdite taký spôsob na určenie tejto hodnoty, pri ktorom Máriovi stačí spraviť nanajvýš $20n + 47$ akcií (táto hodnota výrazne prevyšuje počet akcií, ktoré potrebuje spraviť vzorové riešenie). Odôvodnite, prečo váš postup nepotrebuje viac akcií.

   Ak si nie ste istý, koľko otázok používa váš postup, aj tak nám ho pošlite, určite získate aspoň čiastkové body :)

2. Rebríky a Platformy

15 b popis, 0 b program

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Paulinke na psmolarova@gmail.com

Jedného pekného letného dňa sa Adrianka vybrala do novučičkého Parku Rebríkov a Spúšťacích Kladiek a celé hodiny skúšala rôzne rebríky, lanové dráhy a iné atrakcie. Už od príchodu so zvedavým pohľadom obdivne hľadela na najväčšiu a najkomplikovanejšiu sústavu lanorebríkov akú Park Rebríkov a Spúšťacích Kladiek mal v ponuke.

Starostlivo si vyhliadla moment, keď bol rad ku komplexu najkratší, a hneď sa do neho postavila. Rad pomaličky ubúdal a Adrianka z nudy začala analyzovať ten komplexný systém platforiem pospájaných dokopy rebríkmi z lán.

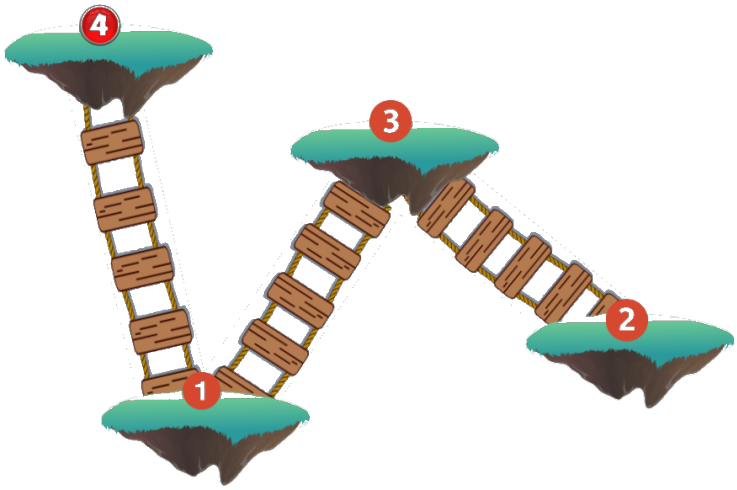
Celá preliezka sa skladala z radu za sebou idúcich platforiem umiestnených v rôznych výškach. Každé dve za sebou idúce platformi boli spojené lanorebríkom, po ktorom sa človek posúval ďalej. Tento lanorebrík buď stúpal alebo klesal v závislosti od vzájomnej výšky platforiem, ktoré spájal. Adrianka si navyše všimla veľmi zaujímavú vlastnosť – žiadne dve platfotmy neboli v rovnakej výške. V propagačnej brožúrke sa dokonca dočítala, že pre každú výšku v metroch od 1 po počet platforiem existuje platforma v danej výške.

Rad sa viac a viac skracoval a čoskoro mala na dráhu nastúpiť aj Adrianka. Zrazu si pri nástupnej plošine všimla plagát: "Uhádni výšky plošín a vyhraj!" Adrianka však pohľadom na plošiny nevedela odhadnúť ako vysoko sú. Počas presunu si teda dôkladne zapamätala aspoň to, na ktorých úsekoch dráhy stúpala, a na ktorých klesala, v nádeji, že jej to pomôže pri odhade.

Pomôžte jej zo získaných dát zistiť možné výšky platforiem!

Príklad

Ak by prvý úsek išiel dole, druhý hore a tretí opäť dole, výška platforiem (v metroch) by mohla byť 4, 1, 3, 2. Rovnako dobré by však bolo aj riešenie 2, 1, 4, 3.



Stúpanie a klesanie vieme zaznačiť schematicky: ↑ značí stúpanie, ↓ klesanie. Napríklad táto lanorebríková sústava sa dá zaznačiť ako ↓↑↓

Úloha

a) (2 body) Pre každú z troch zadaných postupností klesaní a stúpaní nájdite jedno možné rozloženie výšok platforiem, ktoré by mohlo zodpovedať zadanej postupnosti:

- ↓↓↑↓↑ (hľadáte postupnosť 6 platforiem s rôznymi výškami od 1 po 6)
- ↓↓↑↑ (hľadáte postupnosť 5 platforiem s rôznymi výškami od 1 po 5)
- ↓↓↑↑↑↓↓↓↑↑ (hľadáte postupnosť 12 platforiem s rôznymi výškami od 1 po 12)

b) (2 body) Adrianka si povedala, že sa uspokojí aj s tým, keď nájde plošinu, ktorá by mohla byť vo výške 1. Popíšte postup, ktorý pre ľubovoľnú postupnosť stúpaní a klesaní nájde jednu pozíciu pre platformu s výškou 1. Možných pozícii môže byť viac, v takom prípade stačí ak nájdete ľubovoľnú jednu z nich.

Vami navrhnutý postup si môžete otestovať na príkladoch z podúlohy a).

c) (4 body) Navrhnite postup, ktorý pre ľubovoľnú postupnosť stúpaní a klesaní nájde jedno možné priradenie výšok platformám, ktoré spĺňa podmienky zadania.

Vaše riešenie má teda robiť niečo podobné ako ste vy robili v podúlohe a), preto si to na daných postupnostiach môžete vyskúšať. Pozor však na to, že musí fungovať aj pre iné, aj oveľa dlhšie postupnosti. Na overenie správnosti si môžete vytvoriť vlastné postupnosti a skúsiť to aj na nich.

d) (3 body) Na konci dráhy sa Adrianka dozvedela, že cenu útechy dostane aspoň za to, keď uhádne výšku prvej platformy. Rozmýšľa teda, koľko rôznych vyšok je pre túto platformu prípustných. Popíšte, ako by ste pre ľubovoľnú postupnosť stúpaní a klesaní určili, ako rôzne vysoko môže byť prvá platforma.

Ak neviete ako túto podúlohu vyriešiť všeobecne, až 1 bod môžete získať ak ju vyriešite na troch konkrétnych príkladoch z podúlohy a).

e) (4 body) Adrianka sa od manažérky dráhy dozvedela, že lanová dráha je postavená tak, aby bola čo *najadrenalínovejšia*. To sa prejavuje tak, že na začiatok dráhy sú umiestňované čo najvyššie platformy. Ak mali návrhári dráhy viac možností (dodržiac všetky predchádzajúce podmienky vrátane dopredu stanoveného stúpania a klesania) ako umiestniť platformy, vybrali si tú, kde bola prvá platforma čo najvyššia. Ak mali stále viac možností, snažili sa maximalizovať výšku druhej, tretej atď. platformy. Inak povedané, vybrali si lexikograficky najväčšiu vyhovujúcu postupnosť.

Adrianka je presvedčená, že s prihliadnutím na túto podmienku existuje iba jedno riešenie zhodujúce sa s dátami ktoré namerala. Pomôžte jej a navrhnite postup, ktorý pre ľubovoľnú postupnosť stúpaní a klesaní nájde najadrenalínovejšiu lanovú dráhu.

Ako by niečo takéto vyzeralo pre postupnosť z obrázku, teda ↓↑↓↑? Na prvé miesto by išla platforma výšky 4. Stále sú však prípustné dve možnosti – (4, 1, 3, 2) a (4, 2, 3, 1). Vybrať musíme tú druhú, lebo má vyššie umiestnenú druhú platformu.

Programátorské úlohy

Tieto úlohy sú zamerané na praktickú tvorbu programov v niektorom vyššom programovacom jazyku ako je napríklad Python, C++ alebo Pascal. Na stránke odovdzávaš **iba zdrojový kód** svojho programu riešiaceho zadanú úlohu, ktorý bude okamžite automaticky otestovaný a do pár sekúnd sa dozvieš, koľko bodov tvoj program získal. Tieto body ti už nikto nemôže zobrať, ale ak si nezískal plný počet bodov, môžeš opakovane odovzdávať opravený program, až kým nebudeš spokojný s výsledkom.

Ak už vieš programovať, ale ešte si nepracoval s naším testovacím systémom, odporúčame ti zájsť na náš testovač(`http://testovac.ksp.sk`), kde si o tom môžeš prečítať úvodný text a vyriešiť si niekoľko jednoduchých úloh napr. v sade Úvod do programovania.

Ak však **nevieš programovať, tak nezúfaj!** Pripravili sme pre teba tutoriál Úvod do programovania(`https://people.ksp.sk/~ajo/osp/python_tutorial.pdf`), ktorý ťa **naučí základy programovania** v jazyku `Python`. Navyše, za riešenie niektorých úloh na konci kapitol tutoriálu môžeš získať body priamo do PRASKu a tým si vynahradiť neriešenie niektorej z programátorských úloh.

Presnejšie to funguje takto. Na testovači sa nachádza sada úloh s názvom **Úvod do programovania**, zameraná na jazyk `Python`. V tejto sade sa nachádzajú bodované aj nebodované úlohy, ktoré môžeš postupne riešiť a ktoré ti vysvetlia základy programovania. To, ktorými úlohami si vieš nahradiť riešenia jednotlivých programátorských úloh nájdeš na ich začiatku tu priamo v zadaní.

Najskôr si šesticou príkladov môžeš nahradiť úlohy 4 a 5. Samozrejme, toto môžeš urobiť **s každou sadou najviac raz**.

No a v budúcej sérii môžeš za body riešiť ešte ďalšiu šesticu úloh z testovača. Potom by si už mal/a mať dostatočné základy v programovaní, aby si si mal/a šancu s programátorskými úlohami poradiť a nebol/a limitovaný/á tým, že neovládaš žiadny programovací jazyk.

Nezabúdaj, nič ti nebráni riešiť aj úlohy zo sady Úvod do programovania aj klasické programátorské úlohy v PRASKu. Úlohy, ktoré riešiš ako náhradu odovzdávaš na separátnej stránke nášho testovača: `http://testovac.ksp.sk`

3. A tak sa hrali na štrkovisku

0 b popis, 15 b program

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Emovi na emanuel.tesar@trojsten.sk

Jitka a Marcel sa išli spolu hrať von na pieskovisko. Po hodine ich to však prestalo baviť, veď už sú dospelí! A veľké deti sa nehrajú na pieskovisku, hrajú sa na **štrkovisku**. Pobrali si svoje formičky a vybrali sa na neďaleké štrkovisko. No zábava im ani tam nevydržala dlho – skúšali ste robiť pieskový koláčik zo štrku?

Našťastie dostala Jitka nápad na novú hru. Najprv si pozbierali nejaký počet kamienkov a zoradili ich do postupnosti podľa hmotnosti *(Marcel je veľmi citlivý človek, a dokáže bezchybne porovnať váhu ľubuvoľných dvoch kamienkov)*. Následne brali rad radom kamienky, začnúc najťažším. Kamienok si vždy bral hráč, ktorý mal v tom čase menší súčet hmotností kamienkov, poprípade, ak ich kôpky mali rovnakú hmotnosť, vyberala Jitka (**dámy majú prednosť**). Po zobratí posledného kamienku vyhral hráč s väčším súčtom hmotností.

Ako prvá si kamnieok berie Jitka, keďže obaja majú vtedy súčet hmotností 0 a pamätáte sa, ako je to s tou prednosťou.

S veľkým počtom kamienkov však hra trvá dlho. Vedeli by ste im preto pomôcť napísať program, ktorý by zistil, kto z hráčov vyhrá?

Úloha

Na vstupe je **zostupne** usporiadaná postupnosť čísel. Jitka a Marcel sa hrajú hru, pri ktorej postupne berú čísla z postupnosti (začínajúc najväčším). V každom ťahu berie hráč s menším súčtom. Na začiatku alebo v prípade, že majú rovnaký súčet berie Jitka. Vašou úlohou je zistiť, ktorý hráč vyhrá a s akým náskokom.

Formát vstupu

Na prvom riadku vstupu dostanete číslo n ($1 \leq n \leq 10^5$).

Na druhom riadku je n čísel k_1 až k_n – hmotnosti jednotlivých kamienkov. Platí, že $1 \leq k_i \leq 10^6$ a tieto hodnoty sú na vstupe zoradené od najväčších čísel po najmenšie.

Formát výstupu

Na jediný riadok výstupu vypíšte meno víťazného hráča (`Jitka` alebo `Marcel`) a medzerou oddelený náskok s akým daný hráč vyhral. V prípade remízy vypíšte `Remiza`.

Hodnotenie

Príklad obsahuje 5 sád vstupov. Za každú je možné získať najviac 3 body. V prvých troch sadách platí $n \leq 1\,000$.

Tip: Súčty hmotností kamienkov v poslednej sade môžu byť veľmi veľké čísla. Preto si treba uvedomiť, že v niektorých jazykoch majú bežné celočíselné premenné obmedzený rozsah. Napríklad v C++ `int` má klasicky 32 bitov, čo nemusí stačiť. Chcete preto použiť 64 bitové premenné (napr. `long long` v C++, `Int64` v Pascale).

Príklad

vstup

```
5
15 12 11 3 1
```

výstup

```
Marcel 4
```

(Začína Jitka a zoberie kameň hmotnosti 15, potom ďalšie dva kamene zoberie Marcel. Zvyšok už zoberie Jitka, ale to jej na víťazstvo nestačí.)

vstup

```
4
100 20 20 20
```

výstup

```
Jitka 40
```

(Začína Jitka a zoberie 100, zvyšok uz Marcelovi stačiť nebude.)

vstup

```
2
7 7
```

výstup

```
Remiza
```

4. Sekanie bagety

0 b popis, 15 b program

Ak nevieš programovať, nezúfaj! Môžeš sa to naučiť a ešte za to získať body, ktoré sa ti budú počítať namiesto tejto úlohy. Stačí, že pôjdeš na stránku nášho testovača a budeš riešiť sadu **Úvod do programovania**. Každou z úloh Číslo, Zámena, Obdĺžnik 1 si vieš nahradiť 5 bodov z tejto úlohy. Pomôcť si vieš Python tutoriálom, ktorý sme pre teba nachystali. Stačí ak na spodku tejto stránky odovzdáš pdf-ko s prezývkou, ktorú používaš na testovači.

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Maťovi na matej.urban@riesky.sk

Adam si na desiatu spravil masívnu, n centimentrov dlhú bagetu. Potom ale zistil, že sa mu celá nezmestí do ruksaku. Rozhodol sa, že z nej aspoň vyreže úsek dlhý k cm a ten už zbalí bez problémov. Rozloženie surovín v bagete ale nie je rovnomerné a na niektorých miestach chutí lepšie ako na iných. Adam sa preto rozhodol, že si vyberie taký úsek, ktorý chutí najlepšie.

Úloha

Na vstupe dostanete čísla n, k a popis bagety, teda pre každý centimetrový dielik bagety dostanete na vstupe jedno celé číslo, jeho chuť. Chuť vybraného úseku bagety je súčet chutí dielikov, z ktorých sa skladá. Vašou úlohou je vypísať chuť najchutnejšieho súvislého úseku dĺžky presne k.

Pozor, vami vybraný úsek **nemôže** byť kratší ako k centimetrov, bez ohľadu na to ako nechutný by bol. Adam by tak totiž ostal hladný.

Formát vstupu

Na prvom riadku vstupu sú dve čísla n a k ($1 \leq k \leq n \leq 10^6$) – pôvodná dĺžka bagety a dĺžka úseku, ktorý z nej máte vyrezať.

Na druhom riadku je n medzerou oddelených čísel c_1 až c_n, chuť jednotlivých dielikov. Platí, že $-10^9 \leq c_i \leq 10^9$.

Formát výstupu

Na jediný riadok výstupu vypíšte číslo x – chuť najchutnejšieho súvislého úseku dĺžky k cm.

Hodnotenie

Vaše riešenie bude testované na 5 sadách vstupov, za každý môžete získať 3 body. Pre prvé dve sady platí, že $n \leq 1\,000$.

Príklad

vstup	výstup
7 3 -2 5 7 3 9 -10 14	19

Najchutnejší úsek, ktorý môže Adam z bagety vyrezať obsahuje 3, 4 a 5 centimeter s celkovou chuťou $7+3+9 = 19$. *Zvyšné úseky, napríklad 2 až 4 centimeter s chuťou 15 alebo 5 až 7 centimeter s chuťou 13, nie sú natoľko chutné.*

5. Krvilačný súboj planét

0 b popis, 15 b program

Ak nevieš programovať, nezúfaj! Môžeš sa to naučiť a ešte za to získať body, ktoré sa ti budú počítať namiesto tejto úlohy. Stačí, že pôjdeš na stránku nášho testovača a budeš riešiť sadu **Úvod do programovania**. Každou z úloh Najmenší, Akéže je znamienko?, Prostredný si vieš nahradiť 5 bodov z tejto úlohy. Pomôcť si vieš Python tutoriálom, ktorý sme pre teba nachystali. Stačí ak na spodku tejto stránky odovzdáš pdf-ko s prezývkou, ktorú používaš na testovači.

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Krtkovi na krtko@ksp.sk

Včera sa stalo niečo neslýchané! Niečo, čo bude mať katastrofálne následky! A ostáva nám len dúfať, že to neovplyvní našu časť vesmíru.

Ešte predvčerom pokojní obyvatelia planéty Sysľov lietali na ufách krížom krážom galaxiou, zúčastňovali sa všetkých možných programátorských súťaží a všetky do jednej vyhrávali. Včera však utrpeli strašnú porážku, prvýkrát v histórii, a to práve od obyvateľov planéty Žiab. Aby si zachovali česť, neostáva im nič iné, ako planéte Žiab vyhlásiť vojnu. Nie programátorskú, tam si už nie sú takí istí, ale fyzickú a likvidačnú.

V skratke, v neďalekom vesmíre vypukol Krvilačný Súboj Planét.

Najdôležitejším krokom vo vojne je produkcia bojových robotov. Syslí návrhári vytvorili prototyp, na základe ktorého sa budú stavať. Pri stavbe robota sa však využívajú niektoré prototypom definované suroviny, ktorých je len obmedzene veľa, obmedzený je teda aj počet robotov, ktorých vedia vyrobiť.

Na planéte Sysľov sa však nachádza aj jedna špeciálna surovina – žolícium. Tá sa vyznačuje tým, že sa ňou dá nahradiť ľubovoľná iná surovina. Z dostatku žolícia dokonca môžete postaviť celého robota. Ak sa teda žolícium správne prerozdelí medzi zvyšné suroviny, môže sa zvýšiť počet robotov, ktorých vedia na planéte Sysľov vytvoriť. Obyvatelia planéty Sysľov však nevedia, ako žolícium rozdeliť a naprogramovať si to už netrúfajú.

Úloha

Dostanete cenu jedného robota vyjadrenú ako počet jednotlivých surovín, ktoré na jeho výrobu treba. Okrem toho dostanete aktuálny počet každej zo surovín a počet špeciálnej suroviny, žolícia, ktoré môže byť použité ako ľubovoľná surovina. Riešiť potom musíte dva typy úloh: * `Obrana` – dostanete navyše počet robotov, ktorý by sme chceli vytvoriť a vašou úlohou je zistiť, či sa daný počet da vytvoriť zo zadaných surovín. * `Útok` – zistite **maximálny možný počet** robotov, ktorých vieme poskladať zo zadaných surovín.

Ak viete vyriešiť iba jednu z podúloh, nezúfajte, určite sa vám újdu nejaké body.

Formát vstupu

Na prvom riadku vstupu je jedno slovo, a to `Obrana` alebo `Utok`, podľa toho, či teraz treba brániť alebo útočiť.

Na druhom riadku vstupu sú dve čísla, n ($1 \leq n \leq 10^5$) – počet rôznych typov surovín, a b ($0 \leq b \leq 10^{14}$) – počet špeciálnych surovín, ktoré môžete rozdeliť ako chcete.

Na treťom riadku vstupu je n čísel, i-te z nich znamená, že máte k dispozícii A_i ($0 \leq A_i \leq 10^7$) i-teho typu surovín.

Na štvrtom riadku vstupu je n čísel, i-te z nich znamená, že počet potrebných surovín i-teho typu na jedného robota je B_i ($0 \leq B_i \leq 10^6$). Navyše máte garantované, robot stojí aspoň 1 z nejakej suroviny.

Ak sa jedná o obranu, vstup má ešte jeden riadok, na ňom je jedno kladné celé číslo – počet robotov, ktorých treba vyrobiť, aby sme sa ubránili.

Navyše platí, že bez ohľadu nato, či sa Sysľovčania bránia alebo útočia, nemôžu vyrobiť viac robotov ako počet atómov na planéte Sysľov, teda viac ako 10^{10} robotov.

Formát výstupu

Ak sa jedná o `útok`, vypíšte jedno číslo – najväčší počet robotov, ktoré môžeme vyrobiť.
Ak sa jedná o `obranu`, vypíšte `Ano`, ak sa daný počet robotov dá postaviť, a `Nie` ak sa nedá.

Hodnotenie

Je 5 sád vstupov a za každú je možné získať 3 body.

sada	max n	max b
1	10^3	0
2	10^3	10^5
3	10^3	10^14
4	10^5	10^14
5	10^5	10^14

Navyše platí, že v druhej sade sú len obrany.

Upozornenie: Počet vyrobených robotov aj počty surovín môžu byť veľmi veľké čísla. Preto si treba uvedomiť, že v niektorých jazykoch majú bežné celočíselné premenné obmedzený rozsah. Napríklad v C++ `int` má klasicky 32 bitov, čo nemusí stačiť. Chcete preto použiť 64 bitové premenné (napr. `long long` v C++, `Int64` v Pascale).

Príklad

vstup

```
Utok
9 15
6 8 3 7 7 4 9 4 5
1 2 4 2 4 4 2 4 1
```

výstup

```
2
```

Ak si pridáme 5 z druhého typu, 1 z štvrtého typu a 4 z piateho typu a 4 zo siedmeho typu, budeme mať dostatok na 2 robotov. Ak by sme chceli vyrobiť 3, potrebovali by sme pridať aspoň 9 druhého typu, a 8 z piateho typu. Bohužiaľ máme iba 15 bonusových surovín

vstup

```
Obrana
9 15
6 8 3 7 7 4 9 4 5
1 2 4 2 4 4 2 4 1
3
```

výstup

```
Nie
```