Korešpondenčný seminár z programovania

Leták zimnej časti XLIII. ročníka

Korešpondenčný seminár z programovania (KSP) je súťaž programátorov – stredoškolákov a mladších – pripravovaná skupinou študentov Fakulty matematiky, fyziky a informatiky Univerzity Komenského v Bratislave. Naším cieľom je zdokonaliť žiakov v programovaní a v algoritmickom myslení.

Riešením súťažných úloh a štúdiom vzorových riešení sa zlepšíš v programovaní a naučíš sa algoritmicky rozmýšľať. Získané poznatky a skúsenosti využiješ v iných súťažiach v programovaní (napríklad pri riešení Olympiády v informatike), v bežnom živote, počas vysokoškolského štúdia, dokonca aj na prijímacích pohovoroch do zamestnania. Naši riešitelia sa každoročne zúčastňujú a úspešne umiestňujú na medzinárodných olympiádach v informatike (v Austrálii, Taliansku, Kazachstane, Taiwane, …). Mnoho našich bývalých riešiteľov sa tiež bez ťažkostí zamestnalo v špičkových IT spoločnostiach ako Google, Facebook, ESET, …

Ak študuješ na strednej škole a zaujíma ťa programovanie, neváhaj a zapoj sa do KSP:

Ako sa zapojiť do KSP?

- **Prečítaj** si zadania. Nájdeš ich v tomto letáku a na našej stránke `https://www.ksp.sk/ulohy`. Každý rok máme zimnú a letnú časť, obe majú dve kolá s ôsmimi úlohami.
- Ťeš sa, aké sú tento rok pekné úlohy.
- **Vyrieš** úlohy. Nemusíš vyriešiť všetky, nemusíš ich vyriešiť najlepšie ako sa dá. Aj za čiastočné riešenia sa dostávajú body, za každú úlohu za dá získať 0 až 20 bodov.
- Na riešenie úloh jedného kola máš približne mesiac a pol a môžeš ich riešiť doma bez toho, aby si niekam cestoval. Termín odovzdania úloh je napísaný aj na našej stránke, aj v PDF zadaniach. Úlohy sa nedajú odovzdávať po termíne, takže si to, prosím, nenechaj na poslednú chvíľu.
- Úlohy rieš samostatne a neprezrádzaj riešenia ostatným riešiteľom. Odpisovanie riešení a prezradenie riešení pred termínom kola je porušením pravidiel KSP. Po skončení kola sa, samozrejme, o riešeniach rozprávať môžeš. :)
- **Odovdzaj** riešenia úloh. Odkaz na odovzdávanie úloh nájdeš pod webovým zadaním každej úlohy alebo na stránke `https://www.ksp.sk/odovzdavanie`. Na odovzdávanie sa treba prihlásiť, aby sme vedeli, komu máme dať body.
  - Vo väčšine úloh odovzdávaš program a popis.
  - Program je hneď po odovzdaní otestovaný testovačom a hneď vidíš, koľko bodov za program máš. Program môžeš odovzdávať znova a znova, až kým nie si spokojný s výsledkom. Ak nevieš, ako majú vyzerať odovzdané programy, pozri si `https://www.ksp.sk/odovzdavanie-programov`
  - Do popisu slovne napíšeš, ako tvoje riešenie funguje, prečo funguje a tiež odhad časovej a pamäťovej zložitosti programu. Viac sa dozvieš na stránke `https://www.ksp.sk/ako-riesit`. Popis opravia a obodujú vedúci KSP po skončení kola.
- Po skončení kola si **prečítaj vzorové riešenia** úloh (veľa sa z toho naučíš), pozri svoje opravené popisy (či ti tam vedúci nenapísali nejaké poučné komentáre), pozri sa do výsledkovky a **teš sa**, koľko máš bodov. Vo výsledkoch sa hodnotí samostatne letná a zimná časť. V každej časti je dôležitý celkový súčet bodov.
- Prečo sa máš tešiť z bodov? Čítaj ďalej.

Čo môžem vyhrať?

- Okrem neoceniteľných vedomostí, skúseností a zručností, ktoré získaš pri riešení semináru, môžeš vyhrať množstvo skvelých vecí.
- Všetci víťazi od nás dostanú **vecné ceny**.
- Pre 36 najlepších riešiteľov organizujeme každoročne dve týždenné **sústredenia**. Sústredenie je niečo ako tábor, na ktorom spoznáš nových priateľov s podobnými záujmami, naučíš sa čosi viac nielen o programovaní a zažiješ kopec zábavy. Sústredenia sú fakt skvelé akcie, najmä, keď ich organizuje Trojsten.

- Aby ste sa mohli pochváliť ostatným, akí ste šikovní, víťazom všetkých levelov udelíme a pošleme **diplomy**.
- Aj keď sa nedostaneš medzi víťazov, stále môžeš byť úspešným riešiteľom. Úspešný riešiteľ je ten, kto získal aspoň polovicu bodov počas celej časti (letnej, či zimnej).

Pravidlá a levely

Počnúc tridsiatym piatym ročníkom rušíme staré kategórie a prechádzame na nový systém *levelov*.

Každý riešiteľ má level, číslo od 1 po 4. Noví riešitelia začínajú na leveli 1 a pokiaľ sa im v riešení darí, level im postupne rastie. Svoj level si môže každý riešiteľ pozrieť na našej stránke. Riešiteľom s levelom L sa započítavajú body len za úlohy s číslami L až 8.

Vo výsledkových listinách (https://www.ksp.sk/vysledky) sa každému riešiteľovi počíta **5 najlepšie vyriešených úloh**. Celkovo sa dá za časť (dve kolá) získať 200 bodov. Riešitelia, ktorí sa v nejakej výsledkovke umiestnili na jednom z prvých dvoch miest a majú aspoň 150 bodov sú **víťazi**. Najlepších 36 riešiteľov pozývame na sústredenie.

Podrobnejšie pravidlá si môžete prečítať na https://www.ksp.sk/pravidla.

Registrácia

Pred odovzdaním riešenia je potrebné sa zaregistrovať na našej webstránke a vyplniť požadované kontaktné údaje. Odporúčame sa zaregistrovať aspoň pár dní pred odovzdávaním riešenia (pre prípad, že by ste mali počas registrácie nejaké problémy).

Účasťou v KSP nám dávate súhlas spracovať a archivovať údaje, ktoré nám poskytnete pri registrácii, ako aj zverejniť vaše meno, školu, ročník a získané body vo výsledkovej listine.

Korešpondenčný seminár z programovania
XLIII. ročník, 2025/26
FMFI UK,
Mlynská Dolina, 842 48 Bratislava

Úlohy 1. kola zimnej časti

Termín odoslania riešení tohto kola je **6. októbra 2025**. Doprogramovávanie končí 20. októbra 2025.

1. Kobry sú plaché

12 b za popis, 8 b za program

Vedúci KSP boli na chate. Ako tak pripravovali nové zadania, poriadne im vyhladlo. Čakali, čakali a ešte dlhšie čakali, keď zrazu Dušan dostal geniálny nápad. Rozhodol sa, že si spolu s vedúcimi pôjdu ten obed uloviť. A tak vedúci išli do lesa s nádejou, že sa konečne budú môcť poriadne najesť.

V lese pobudli celkom dlhý čas, avšak nenašli nič, čo by sa dalo uloviť a zjesť. Po chvíli vyšli na lúku s vysokou trávou. Tam konečne zahliadli niečo, čo by si mohli uloviť. V tráve sa totiž hemžilo mnoho hadov rôznych dĺžok. A tak sa vedúci miesto lovenia si obedu pustili do hádania sa, kto z nich vidí hadov akej dĺžky. Pomôž vedúcim zistiť, aký je súčet dĺžok hadov v tráve, skôr, ako ich vedúci svojím krikom vyplašia.

Úloha

Na vstupe dostanete postupnosť rôznych znakov. Naším cieľom je spočítať súčet dĺžok všetkých hadov v tráve. Had sa skladá z hlavičky `~O` dĺžky 1 (jazyk nepočítame), tela pozostávajúceho zo znakov `=`, pričom každý znak má dĺžku 1 (pozor, had môže mať aj telo dĺžky 0), a chvostíka `>` dlžky 1. Všetky hady sú orientované rovnakým smerom, a to takým, že vľavo sa nachádza hlavička a vpravo chvostík.

Formát vstupu

Vstup je reťazec dĺžky N.

V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2	3	4
$1 \leq N \leq$	500	10 000	100 000	10^6

Formát výstupu

Vypíš jeden riadok a v ňom jedno celé číslo, ktoré zobrazuje súčet dĺžky hadov.

Príklady

vstup	výstup
`~O==>~O>`	`6`

V tomto prípade sú v tráve 2 hady, prvý má dĺžku 4 a druhý 2. Súčet ich dĺžok je teda 6.

vstup	výstup
`~O===8>`	`0`

V tomto prípade v tráve nevidíme žiadne hady.

2. Ešte čakajte

12 b za popis, 8 b za program

Predstavte si typický deň vo firme *Kódíme S Prestávkami*. Je 10:50 a všetci hladní programátori sa pýtajú jedinú otázku: "Kedy bude obed?"

V tejto firme však nechodia na obed len tak hala-bala. Na veľkej obrazovke v kuchynke svieti dlhý reťazec núl a jednotiek, ktorý určuje čo sa práve deje:

- Blok núl: "Obed sa ešte pripravuje."
- Blok jednotiek: "Obed je hotový, môžete ísť."

Samozrejme, signál musí byť vždy pekne v poradí – najprv všetky nuly, potom všetky jednotky.

No ako to už býva, systém sa občas pokazí. Namiesto pekného prechodu z núl na jednotky sa na obrazovke objaví niečo takéto:

```
0101110010
```

Výsledkom je panika na chodbe. Máme ísť na obed? Ešte počkať? Máme si objednať pizzu?

Programátori si však vedia poradiť. Predpokladajú, že sa v signáli stalo čo najmenej chýb, a tak sa snažia zistiť, koľko čísel by stačilo zmeniť, aby sa signál opravil a všetci konečne vedeli, čo majú robiť.

Úloha

Dostanete reťazec tvorený nulami a jednotkami. Zistite, koľko najmenej čísel treba v zadanom reťazci zmeniť tak, aby sa z neho stal signál tvaru: 000…111 Teda aby ste dostali reťazec tvorený súvislým blokom núl a potom súvislým blokom jednotiek, pričom oba súvislé bloky musia mať dĺžku aspoň 1.

Formát vstupu

Na prvom riadku vstupu je číslo n ($2 \leq n \leq 10^6$) udávajúce dĺžku reťazca. V druhom riadku sa nachádza reťazec tvorený n znakmi len 0 alebo 1.

V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2	3	4
$2 \leq n \leq$	15	1000	10^5	10^6

Formát výstupu

Vypíš jeden riadok a v ňom jedno celé číslo udávajúce najmenší počet zmien, ktoré musíme spraviť, aby sme sa dostali do požadovaného stavu.

Príklad

vstup	výstup
7 0101101	2

Môžeme zameniť nuly na tretej a šiestej pozícii za jednotky, vznikne nám reťazec 0111111. Alebo môžeme zmeniť jednotku na druhej pozícii a nulu na šiestej pozícii. Vtedy vznikne reťazec 0001111.

vstup	výstup
5 00000	1

Reťazec má obsahovať súvislý blok núl aj jednotiek, preto potrebujeme urobiť jednu zmenu - pridať na koniec reťazca jednotku. Vznikne nám tak reťazec 00001.

3. Dole kopcom

12 b za popis, 8 b za program

Dušan je vášnivý turista. Pri svojej poslednej návšteve Vysokých Tatier sa rozhodol písať si denník. Po každom kilometri si zapíše, či išiel hore alebo dole kopcom jeden výškový meter. Dušanovi však zmokol batoh a niektoré zápisky niesú čitateľné. Po príchode domov chcel svojím kamarátom ukázať, kde jedol fajnový obed. Pamätal si, že obed jedol na najvyššom kopci. Avšak kvôli nekompletným záznamom si nie je istý, kde to naozaj bolo. Pomôž mu zistiť koľko takýchto miest mohlo cestou byť.

Úloha

Pre jednoduchosť môže predpokladať stúpanie + a klesanie - za konštantné, posunieme sa hore alebo dole vždy o tú istú hodnotu. **?**, znamená, že z poznámok nie je jasné kam sa Dušan posunul ale určite vieme povedať, že išiel buď dole alebo hore – nikdy neostal na tej istej výške. Chceme zistiť koľko kopcov mohlo byť tými najvyššími, a ktoré to mohli byť.

Formát vstupu

V prvom riadku vstupu je číslo n udávajúce počet zápiskov.

Na druhom riadku sa nachádza n zápiskov (`+`, `-`, `?`), označujúcich Dušanov pohyb hore a dole. `?` znamená, že nevieme určiť, či Dušan išiel hore alebo dole.

V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2	3	4	5	6	7	8
$1 \leq n \leq$	1 000 000	10	100	1 000	5 000	100 000	500 000	1 000 000
$0 \leq \#? \leq$	0	5	50	500	1 000	10 000	100 000	300 000

Formát výstupu

Vypíš dva riadky. V prvom jedno celé číslo, ktoré hovorí, na koľkých pozíciach sa mohol nachádzať najvyšší kopec Dušanovej cesty. Na druhý riadok napíš pozície, na ktorých sa tieto kopce mohli nachádzať. Na poradí vypísaných pozícií nezáleží.

Príklady

vstup

```
2
?-
```

výstup

```
2
0 1
```

Vrchol môže byť na 0., kde stál pred tým ako sa pohol. Alebo na 1. pozícii. Ak by sa Dušan najprv posunul dole kopcom (`?` -> `-`) tak ak začíname vo výške 0, by mohol stáť vo výškach 0, -1, -2. Ak by sa Dušan najprv posunul hore kopcom (`?` -> `+`) tak ak začíname vo výške 0, by mohol stáť vo výškach 0, 1, 0 .

vstup

```
6
+??-+-
```

výstup

```
4
1 2 3 5
```

Rozoberme si postupne všetky možnosti:

- *`+++-+-` -> 0, 1, 2, 3, 2, 3, 2*
- *`+---+-` -> 0, 1, 0,-1,-2,-1,-2*
- *`+-+-+-` -> 0, 1, 0, 1, 0, 1, 0*
- *`++--+-` -> 0, 1, 2, 1, 0, 1, 0*

Najvyššie miesto cesty môže byť na 1., 2., 3. alebo 5. pozícii. V prvom prípade by najvyšší kopec bol ten na tretej pozícii a piatej pozícii. V druhom prípade by to bol iba ten na prvej pozícii. v treťom prípade by to bol ten na prvej, tretej a piatej pozícii a v poslednom prípade by to bol kopec na druhej pozícii.

4. Yoooy, pauza na obed

0 b za popis, 20 b za program

Ak máš hocijaké otázky k tejto úlohe, napíš Andrejovi na andrej.lackovic+yoooy@trojsten.sk

Vedúci pripravovali úlohy a z toľkého pripravovania poriadne vyhladli. Rozhodli sa teda ísť na obed, ale predtým delegovali svoju robotu veľkým jazykovým modelom.

Keď sa spokojne naobedovali, zistili, že veľké jazykový modely úspešne napísali programy, tie ale obsahovali chyby. S plným bruchom sa im ale nechcelo programy opravovať, a tak to nechali na teba.

Úloha

Dostaneš niekoľko programov, ktoré majú bugy. Tvojou úlohou je kódy opraviť. Aby to ale nebolo také jednoduché, musíš to spraviť s čo najmenej úpravami pôvodného kódu.

Na plný počet bodov budeš pravdepodobne potrebovať uplatniť viacero syntaktických trikov.

Malý hint: v pythone vieš dať viacero príkazov na jeden riadok, pokiaľ ich oddelíš bodkočiarkou.

Bubble sort

Vstup

Na vstupe je n celých čísiel ($n \leq 1000$).

Výstup

Vypíš ich zoradené.

Program

Tento listing sa nachádza iba v HTML formáte.

Halda

Dostávaš požiadavku `+ x` a `-`, pri každej vypíšeš aktuálny najväčší prvok haldy.

Vstup

Na prvom riadku je číslo n - počet požiadaviek.
Nasleduje n riadkov, buď `+ x` (pridať do haldy prvok x) alebo `-` (odstrániť najväčší prvok).

Výstup

Pri každej operácií vypíš aktuálne najväčší prvok.

Program

Tento listing sa nachádza iba v HTML formáte.

Fibonacci++

Teraz máš funkčný program, ale je napísaný v C++, nie v Pythone… Treba ho teda prepísať do Pythonu.

Vstup

Na vstupe je jedno celé číslo n ($0 \leq n \leq 10000$).

Výstup

Na výstupe vypíš jedno celé číslo – n-té Fibonacciho číslo.

Program

Tento listing sa nachádza iba v HTML formáte.

Knapsack

Tentokrát máme typický knapsack[1].

Vstup

Na prvom riadku vstupu sa nachádzajú dve čísla - n (počet predmetov) a W (nosnosť batohu).
Na druhom riadku vstupu sa nachádza n medzerou oddelených čísiel, ktoré reprezentujú váhy jednotlivých predmetov w_i.
Na treťom riadku vstupu sa nachádza n medzerou oddelených čisiel, ktoré reprezentujú vzácnosti jednotlivých predmetov v_i.

Výstup

Na jedinom riadku výstupu vypíšte jedno číslo – najväčšiu možnú vzácnosť predmetov, ktorá sa vám zmestí do batohu.

Program

Tento listing sa nachádza iba v HTML formáte.

Quick select

Nájdi k-ty najmenší prvok.

Vstup

Na prvom riadku sú čísla n a k ($1 \leq k \leq n \leq 1000000$).
Na druhom riadku je n medzerou oddelených čísel, ktoré reprezentujú pole.

Výstup

Na jedinom riadku výstupu vypíš jedno číslo – k-ty najmenší prvok.

[1] https://en.wikipedia.org/wiki/Knapsack_problem

Program

Tento listing sa nachádza iba v HTML formáte.

Bodovanie

Za každý opravený program sa dajú získať 4 body. Tieto body sa prenásobia e/x, kde e je očakávaný počet zmenených znakov (podľa vzoráku) a x je počet tebou zmenených znakov.

Počet zmenených znakov sa počíta ako levenshteinova vzdialenosť[2] dvoch stringov.

Pri počítaní ti vie pomôcť tento Python skript:

Listing programu (Python)

```python
import sys

def levenshtein(a: str, b: str):
    a = a.strip()
    b = b.strip()

    if len(a) == 0:
        return len(b)
    if len(b) == 0:
        return len(a)

    n = len(a)
    m = len(b)
    lev = [[0 for _ in range(m + 1)] for _ in range(n + 1)]

    for i in range(0, n + 1):
        lev[i][0] = i
    for i in range(0, m + 1):
        lev[0][i] = i

    for i in range(1, n + 1):
        for j in range(1, m + 1):
            insertion = lev[i - 1][j] + 1
            deletion = lev[i][j - 1] + 1
            substitution = lev[i - 1][j - 1] + (1 if a[i - 1] != b[j - 1] else 0)
            lev[i][j] = min(insertion, deletion, substitution)

    return lev[n][m]

with open(sys.argv[1], "r") as a, open(sys.argv[2], "r") as b:
    print(levenshtein(a.read(), b.read()))
```

Ak chceš porovnať súbory `a.py` a `b.py`, stačí spustiť:
`python3 test.py a.py b.py`

Odovzdávanie

Odovzdaj upravený nasledovný súbor. K tejto úlohe sa neodovzdáva popis.

Tento listing sa nachádza iba v HTML formáte.

5. Obedujeme koláčiky

12 b za popis, 8 b za program

Mačička a jej partia veľmi radi chodia do cukrárne. Keďže je ale partia veľmi veľká, tak sa tam nebaví

[2] https://en.wikipedia.org/wiki/Levenshtein_distance

každý s každým. Členovia, ktorý sa spolu bavia tvoria dvojice, ktoré voláme kamarátstva. Kamaráti sa spolu potom stretávajú pri koláčiku. V Kocúrkove sú presne dve cukrárne - jedna podáva iba ríbezľové koláče a druhá iba banánové koláče. Každý kamarát má samozrejme svoju preferenciu a je šťastný, iba ak idú do cukrárne, ktorá predáva jeho obľúbený koláč. V Kocúrkove sú všetci veľmi tvrdohlavý a nemajú radi zmenu. Preto každé kamarátstvo chodí, a aj vždy chodiť bude do tej istej cukrárne. Chodia tam dokonca aj vtedy, keď obaja kamaráti preferujú druhú príchuť koláčiku.

Kocúrkovo sa nevyhlo inflácií a chodiť na predražené koláčiky je v dnešnej dobe veľmi neekonomické. Mačička chce, aby jej partií ostali nejaké peniaze aj na spoločné výlety (napríklad na Mars, aj keď nevedia, čo stojí lístok) a povedala si, že takto to ďalej nepôjde. Dobre vie, že na to aby jej partia s n členmi zostala pokope je potrebných iba $n-1$ kamarátstiev (partia je pokope, ak sa ľubovoľní dvaja členovia vedia skontaktovať buď priamo, alebo cez nejakých svojich kamarátov. Ako všetci dobre vieme, nemôžete napísať človeku, s ktorým nie ste kamarát). Mačičke nerobí problém pomôcť niektorým kamarátstvam aby sa rozpadli, ale chce, aby jej partia zostala pokope a všetci boli šťastní. To znamená, že každému kamarátovi musí ostať aspoň jedno kamarátsvto, ktoré chodí do jeho obľúbenej cukrárne.

Úloha

Máme partiu o n členoch ktorá je pokope a m kamarátstiev. Každý člen má rád buď ríbezľové alebo banánové koláčiky. Každé kamarátstvo má dané, do ktorej z dvoch cukrární chodí. Pomôžte Mačičke nájsť $n-1$ kamarátstiev, ktoré keď ostanú tak bude platiť, že skupina je pokope a každý člen má aspoň jedno kamarátstvo, ktoré chodí do jeho obľúbenej cukrárne alebo jej povedzte že sa to nedá.

Formát vstupu

V prvom riadku vstupu sú čísla n a m udávajúce veľkosť partie a počet kamarátstiev. Nasleduje m riadkov. Na i-tom riadku sú medzerou oddelené čísla a, b ($1 \leq a, b \leq n$) určujúce indexy členov partie ktorý sa kamarátia a znak s (s {R, B}), ktorý určuje, do ktorej cukrárne chodia na koláčiky. Na poslednom riadku je string n znakov p, p_j {R, B}, kde p_j je preferencia kamaráta j.

Sada	1	2	3	4
$2 \leq n \leq$	20	100	10^5	$3 * 10^5$
$1 \leq m \leq$	1 000	10^4	10^5	$3 * 10^5$

Je zaručené, že v prvej a tretej sade kamarátstva nevytvárajú cyklus (teda pre ľobovolných dvoch členov partie platí, že ak sa chcú prostrednictvom kamarátstiev skontaktovať, tak existuje práve jedna možnosť ako to vedia docieliť).

Formát výstupu

Ak riešenie neexistuje, vypíš jeden riadok, na ktorom bude napísané `Neda sa`. Ak riešenie existuje vypíš medzerou oddelené i určujúce indexy kamarátstiev, ktoré ostali.

Príklad

Input:

```
3 3
1 2 R
1 3 B
2 3 B
RRB
```

Output:

```
3 1
```

Necháme tretie a prvé kamarátstvo. Správne riešenie by tiež bolo nechať druhé a prvé kamarátstvo.

Input:

```
3 4
1 2 R
```

```
1 2 B
1 3 B
2 3 B
RRR
```

Output:

```
Neda sa
```

Žiadne kamarátstvo 3 člena nechodí do cukrárne, kde predávajú ríbezľové koláčiky.

6. Bazošové pásy

12 b za popis, 8 b za program

Vedúci KSP sú hladní a nevedia sa dočkať, kedy konečne bude obed hotový. Krtko dostal extrémne dobrý nápad, že si poskladá robota, ktorý nám bude pripravovať obedy, aby to netrvalo tak dlho. Po dlhej chvíle pozerania rôznych možností na e-shopoch sa rozhodol, že najefektívnejšie bude si kúpiť továrenské pásy z Bazošu a tie zapojiť za seba. Pásy samozrejme nestačia – potrebujeme robotov, ktorý ten samotný obed budú pripravovať, kým sa vozí na páse. Zároveň sa krtkovi podarilo na matfyze vybaviť miestnosť, kde sa nám tento dlhý pás zmestí.

Úloha

Máme miestnosť, ktorá je dlhá l. Na začiatku miestnosti máme kopu prázdnych tanierov na ktoré potrebujeme pripraviť obedy. Na to aby bol obed hotový, potrebuje, aby počas prípravy prešiel cez aspoň r robotov. Zároveň potrebujeme zaručiť, aby došiel presne na koniec miestnosti, kde je výdajné okienko. Na bazoši je viacero inzerátov s rôznymi pásmi – každý z nich má nejakú priepustnosť p (koľko obedov za hodinu vie tento pás previesť), dĺžku d a buď pri tomto páse je, alebo nie je robot.

Zistite, ktoré pásy má Krtko kúpiť, aby jeho výtvor vedel pripravovať obedy čo najrýchlejšie.

Formát vstupu

V prvom riadku vstupu sú 3 čísla n ($1 \leq n \leq 1000$), l ($1 \leq n \leq 10^5$), r ($1 \leq r \leq n$) udávajúce počet inzerátov na bazoši, dĺžku miestnosti a to koľko robotov potrebujeme na prichystanie obedu.

Potom nasleduje n riadkov – popis jednotlivých inzerátov. Pre každý inzerát máme 3 hodnoty d_i, p_i ($1 \leq p_i \leq 10^6$), r_i. d_i je dĺžka pásu na tomto inzeráte, p_i je jeho priepustnosť a r_i je `A` ak tento pás obsahuje robota a `N` ak nie.

Sada	1	2	3	4
$1 \leq N \leq$	20	100	700	700
$1 \leq l \leq$	500	10^4	25 000	25 000

Navyše v 3. sade je $r = 0$, teda netreba žiadnych robotov.

Formát výstupu

Na výstup vypíš jedno číslo - najväčšiu priepustnosť, ktorú vie mať pás dlhý **presne** l a s aspoň r robotmi. Ak pás zostrojiť nevieme, tak vypíš `-1`.

Príklady

vstup

```
4 100 0
40 1500 N
60 700 A
25 850 N
35 2700 A
```

výstup

```
850
```

Najviac sa oplatí kúpiť prvý, tretí a štvrtý pás. Mohli by sme aj prvý a druhý, ale potom by sme mali menšiu priepustnosť

vstup

```
3 200 2
100 1700 N
100 1500 A
100 1300 A
```

výstup

```
1300
```

Musíme použiť všetky pásy s robotmi, inak nevieme pripraviť obed

vstup

```
1 100 0
110 4747 A
```

výstup

```
-1
```

Jediný inzerát, čo máme nám ponúka až príliš dlhý pás - nezmestí sa nám do miestnosti

7. Exotické korenia

12 b za popis, 8 b za program

Marcel si pred vedúcovskou chatou KSP kúpil automatický robotický systém na uskladňovanie korenia od firmy Kuchárske Stroje a Potreby s.r.o., prístroj má tvar kruhovej poličky, v ktorej je jeden rad priehradok, v každej priehradke je niekoľko vrecúšok s jedným druhom korenia.

Prístroj má hlavicu, ktorá vyberá a podáva korenia. Hlavica vždy ukazuje na jedno vrecúško korenia.

Marcel si pozerá svoje recepty a rozmýľa nad tým, ako si rozmiestniť korenia tak, aby nestrávil veľa času čakaním na to, kým mu robot podá korenie. Aby vedel optimalizovať toto rozloženie, tak by od vás potreboval vykonávať 2 typy udalostí:

- Zmeň počet vrecúšok s korením v nejakej priehradke.
- Odpovedaj na otázku, ako dlho bude robotovi trvať vybrať konkrétny druh korenia, teda dostať sa zo začiatočnej pozície, vrchu priehradky x, na koncovú pozíciu, spodok priehradky x', odkiaľ vie vybrať dané korenie.

Úloha

Máme jednu cyklickú radu priehradok, dlhú n (v každej priehradke je niekoľko vrecúšok korenia jedného typu). V priehradke i sa nachádza a_i vrecúšok.

Poličku obsluhuje robot, ktorý sa vie hýbať tromi spôsobmi:

- Pohyb cyklicky doprava - z vrchu priehradky x na vrch priehradky $x+1$, alebo zo spodu priehradky x na spodok priehradky $x+1$,resp na 0 ak sa $x = n-1$, tento pohyb trvá 1 sekundu.
- Pohyb cyklicky doľava - z vrchu priehradky x na vrch priehradky $x-1$, alebo zo spodu priehradky x na spodok priehradky $x-1$, respektíve na $n-1$ ak sa $x = 0$, tento pohyb trvá 1 sekundu.
- Pohyb dole - Ak sa robot nachádza na vrchu priehradky i, tak sa vie za a_i sekúnd dostať na spodok priehradky i.

Máme dva typy udalostí:

- Zmeň počet vrecúšok v priehradke x_i na a_{new}.
- Otázka: Zisti, za koľko najmenej sekúnd sa vie robot presunúť, ak sa na začiatku nachádza na vrchu priehradky x, a potrebuje sa dostať na spodok priehradky x'? Robot môže použiť ľubovoľne veľa pohybov doprava a doľava, a najviac jeden pohyb dolu.

Formát vstupu

V prvom riadku vstupu sú medzerou oddelené čísla n, q, kde $1 \leq n, q \leq 5 \cdot 10^5$, udávajúce počet priehradok a otázok.

Na nasledujúcom riadku sú medzerou oddelené hodnoty a_i, počty vrecúšok na kôpkach v priehradkách, kde $1 \leq a_i \leq 10^9$ a $0 \leq i \leq n-1$.

Na nasledujúcich q riadkoch sa nachádzajú udalosti 2 typov:

- "! x a_{new}" - zmeň hodnotu a_x na a_{new}, kde $0 \leq x \leq n-1$ a $1 \leq a_{new} \leq 10^9$.
- "? x x'" - vypíš minimálny potrebný čas na to, aby sa robot dostal z vrchu priehradky x na spodok priehradky x', kde $0 \leq x, x' \leq n-1$.

Obmedzenia a hodnotenie

Táto úloha má 8 testovacích sád. V nich sú nasledovné obmedzenia:

Sada	1	2	3	4	5	6	7	8
$1 \leq n \leq$	100	1 000	10^6	10^6	10^6	10^6	10^6	$2 \cdot 10^6$
$1 \leq q \leq$	100	1 000	10^5	10^5	10^5	10^5	10^5	$5 \cdot 10^5$

V sadách $3, 5, 6$ sa navyše nachádzajú iba udalosti typu ?.

V sadách $3, 4$ platí, že vo všetkých udalostiach typu ? je $|x - x'| = N/2$, a N je párne.

Vstupy a výstupy pre túto úlohu sú veľké, preto odporúčame používať FastIO a `\n` namiesto `endl` v C++. Upozorňujeme vás, že pomalšie jazyky ako Python nemusia prechádzať v časovom limite.

Formát výstupu

Pre každú udalosť typu ? vypíš na samostatný riadok odpoveď.

Príklady

vstup

```
5 2
1 1 1 1 1
? 0 0
? 0 4
```

výstup

```
1
2
```

V prvej otázke sa robot vie posunúť dole, za 1 sekundu.

V druhej otázke sa robot posunie doľava a dole, všimnite si, že tento pohyb je cyklický.

vstup

```
5 3
9 9 9 9 9
? 0 0
! 4 1
? 0 0
```

výstup

```
9
3
```

V prvej otázke sa robot vie posunúť dole za 9 sekúnd.

Po zmene sa v druhej otázke oplatí prejsť doľava, dole za 1 sekundu a doprava, taktiež si všimnite cyklickosť pohybov do strán.

8. Diverzita obedov

12 b za popis, 8 b za program

Diverzita obedov hmyzu je v súčasnom spoločenskom diskurze prekvapivo málo diskutovaná téma. V KSP sa však dlhodobo snažíme o vyváženú stravu pre všetkých - nech už ide o akékoľvek sústredenie či letnú školu. Je ale pravda, že na hmyz sme doteraz veľmi nemysleli. Niektorí by dokonca mohli oprávnene namietať, že vôbec. Práve preto nastáva zmena. Zmena paradigmy, aká sa neobjavuje každé kolo, každú sériu, ba dokonca ani každý ročník. A keďže skutočná zmena musí prísť zdola, rozhodli sme sa zapojiť vás, milí riešitelia — ako inak než tematickou úlohou!

Bol raz jeden strom. A na ňom žilo mnoho rôznych druhov hmyzu. Ako to už v prírode chodí, silnejší hmyzáci mohli na obed zjesť tých slabších a následne obsadiť ich miesto na strome. Všetci hmyzáci však túžili po jedinom — mať možnosť zjesť čo najviac iných hmyzákov a získať tak čo najväčšiu diverzitu obedov. Bolo im jasné, že nemožno vyhovieť každému. No boli by radi, keby boli po strome rozmiestnení tak, aby bol súčet diverzít ich obedov čo najväčší.

Úloha

Je zadaný strom. Dobrú cestu nazveme takú cestu v strome, ktorej čísla vrcholov tvoria klesajúcu postupnosť dĺžky aspoň 2. Úlohou je zistiť, koľko najviac dobrých ciest sa môže v strome nachádzať po ľubovoľnom prečíslovaní vrcholov.

Formát vstupu

V prvom riadku vstupu je číslo n ($1 \leq n \leq 200000$) udávajúce počet vrcholov stromu.

Nasleduje $n-1$ riadkov, ktoré popisujú hrany daného stromu. Na každom z nich sú čísla u, v ($1 \leq u, v \leq n$), ktoré označujú, že medzi vrcholmi u a v je hrana.

V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2	3	4
$1 \leq n \leq$	15	200	2000	200000

V druhej sade navyše platí, že všetky vrcholy majú stupne ≤ 10.

Formát výstupu

Vypíšte jeden riadok a v ňom jedno celé číslo - najväčší počet dobrých ciest, ktorý vieme dosiahnuť prečíslovaním vrcholov.

Príklad

vstup

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

výstup

```
9
```

Ak prečíslujeme vrcholy $1, \ldots, 5$ postupne na $3, 4, 5, 2, 1$, tak dostaneme 9 dobrých ciest.

vstup

```
8
5 6
1 4
2 8
3 6
1 3
3 8
3 7
```

výstup

```
22
```