Проблем 1. Моћни ренџери

Моћни ренџери су суперхероји који бране Земљу од напада злих сила предвођених краљицом Ритом и господаром Зедом. У сталној борби против зла помажу им њихови роботи звани зордови. Ипак, зордови немају неисцрпан извор енергије, већ се морају допуњавати с времена на време. У једној бици ангажован је одређен број зордова: неки од њих се боре (и тако троше енергију), док се други допуњавају. По завршеној бици, зордови који су се борили потрошили су сву енергију, и не могу се борити у наредним биткама све док се не допуне, а зордови који су се допуњавали остају пуни и спремни да се боре кад затреба. С обзиром на то што зле силе у намери да поробе Земљу стално шаљу све моћнија и моћнија чудовишта, у свакој бици потребно је ангажовати све више и више зордова: прецизније, у i-тој по реду бици потребно је ангажовати тачно i зордова (од којих се неки боре, а остали се допуњавају).

Зордови се допуњавају помоћу солартомске енергије. Вођа Ренџера, Зордон, мора да води прецизну евиденцију колико зордова се допуњава у којој бици, како би знао да свом помоћнику, роботу Алфи, каже колико солартомске енергије треба да купи у дућану на углу. Ипак, времена су тешка, те и Зордон купује, народски речено, 'на рецку'; другим речима, након што Ренџери успешно одбране Земљу и протерају зле силе из васионе (за шта им треба много битака), Зордон измирује рачуне с дућаном.

Овај пут посао Ренџера био је заиста тежак: од n зордова које поседују (где је n непаран број, да зликовци не би могли поделити ренџере на два једнака дела и напасти сваки део понаособ), у последњој бици, n-тој по реду, морали су да ангажују све! Срећом, у тој бици извојевали су коначну победу и заувек раскрстили са злим силама, па неће бити више напада (нећемо ни помишљати шта би било да су се поквареňаци успели задржати још мало и послати још моћније чудовиште, које би захтевало $n+1$ зордова; само једно је сигурно: у том случају нико од вас не би прошао на ИОИ, јер не би ни било ИОИ, јер не би било ни Земље, али оставимо ситнице по страни). У свој тој голготи Зордон је негде затурио цедуљче на које је бележио колико се зордова допуњавало у којој бици. Остало му је забележено само који од зордова су на почетку (пре свих битака) били пуни а који празни, и још може уочити да су сада, после свега, сви зордови истог стања (тј. или су сви пуни, или су сви празни). Власник дућана је јако љут и прети да ће му запленити неке ствари из Командног центра, те вас Зордон моли за помоћ: одредите колико је зордова у којој бици било допуњавано.

Улаз. (Улазни подаци се налазе у датотеци `rendzeri.in`) У првом реду улазне датотеке налази се природан, непаран број n ($1 \leq n < 10^6$), број зордова које Ренџери поседују. У наредном реду налази се низ знакова '`+`' и '`-`', дужине n, где је i-ти знак једнак '`+`' ако је i-ти зорд био пун пре доласка зликоваца, а '`-`' ако је био празан.

Излаз. (Излазне податке уписати у датотеку `rendzeri.out`) У i-том реду излазне датотеке треба уписати колико се зордова допуњавало током i-те битке. Ако се задатак може решити на више начина, исписати било који.

Пример 1.

rendzeri.in	rendzeri.out
5	1
-+---	2
	0
	4
	0

Објашњење. У питању је првобитна постава Ренџера: Зак, Кимберли, Били, Трини и Џејсон. На слици је приказано како су се они борили: бледом нијансом приказани су празни зордови, а јаким тоном пуни. Видимо да су на крају сви зордови истог стања (у овом случају празни), што се поклапа с условом задатка. Такође можемо видети да у i-тој бици учествује тачно i зордова (подсећамо: неки се боре и тиме троше енергију, а остатак од тих i се допуњује), као што је речено.

Пример 2.

rendzeri.in

```
7
+++++++
```

rendzeri.out

```
0
0
0
3
4
0
7
```

Проблем 2. Стабло

Одредити број минималних покривајућих стабала датог повезаног тежинског графа у коме се ни једна тежина гране не понавља више од 4 пута.

Улаз. (Улазни подаци се налазе у датотеци `stabla.in`) У првом реду улаза су записани бројеви N и M $(1 \leq N, M \leq 5 \times 10^4)$, број чворова и број грана датог графа. У сваком од наредних M редова записана су три цела броја A, B и W $(1 \leq A, B \leq N,\ 1 \leq W \leq 2^{30})$, који означавају да између чворова A и B постоји грана тежине W.

Излаз. (Излазне податке уписати у датотеку `stabla.out`) У излазну датотеку исписати остатак траженог броја минималних покривајућих стабала датог графа који се добија при дељењу са 1000003.

Пример 1.

stabla.in	stabla.out
3 4 1 2 6 1 2 6 2 3 6 3 1 6 3 3 8	5

Све гране су исте тежине, између чворова 1 и 2 постоје две гране, а постоји и грана којој су оба краја у чвору 3.

Пример 2.

stabla.in	stabla.out
6 8 1 2 1 2 3 2 3 4 3 4 5 3 5 6 2 6 1 1 2 6 1 3 5 3	6

Проблем 3. Долина

Планинарски савез јужне Србије је одлучио да сагради нову атракцију на Старој Планини. По први пут, свет ће видети скијашку долину, стазу која иде и низбрдо и узбрдо. Они верују да ће скијаши достићи довољну брзину при спусту, како би се попели у другом делу стазе (под условом да почетна висина није мања од крајње). Да би уживање било што веће, потребно је направити најдужу такву стазу.

Да би поједноставили проблем, Савез преставља терен матрицом $M \times N$ поља, а висина сваког поља је позната – преузета из Географског Института Србије. Скијашка долина је онда низ суседних поља, која се не понављају, тако да висине поља прво само опадају дуж низа, а потом, почев од неког поља, висине само расту до краја низа. Два поља су суседна ако деле заједничку страницу. Дужина скијашке долине је број поља у том низу.

Математички прецизније то можемо исказати на следећи начин. Терен је матрица $M \times N$ поља, где (i, j) представља поље у i-том реду и j-тој колони, а $h(i, j)$ је његова висина. Скијашка долина је низ $(x_1, y_1), (x_2, y_2), ..., (x_l, y_l)$, такав да:

- за свако $i(1 \leq i \leq l-1)$, важи $x_i = x_{i+1}$ или $y_i = y_{i+1}$ (суседи),
- ако је $i \neq j(1 \leq i, j \leq l)$, важи $x_i \neq x_j$ или $y_i \neq y_j$ (нема понављања), и
- постоји $k(1 \leq k \leq l)$, тако да важи

$$h(x_1, y_1) > h(x_2, y_2) > ... > h(x_{k-1}, y_{k-1}) > h(x_k, y_k) < h(x_{k+1}, y_{k+1}) < ... < h(x_l, y_l)$$

  (доле па горе).

Дужина ове скијашке долине је l.

Планинарски Савез нема пуно искуства у програмирању, па те је замолио да напишеш програм који ће наћи најдужу скијашку долину задатог терена. Ако постоји више скијашких долина максималне дужине, можеш одабрати било коју.

Улаз. (Улазни подаци се налазе у датотеци `dolina.in`) Први ред улаза садржи целе бројеве M и N, респективно, раздвојене празнином ($1 \leq M, N \leq 70$). У сваком од наредних M редова налази се N бројева, тако да j-ти број у i-том реду представља $h(i, j)$. Висине поља су различити цели бројеви у опсегу $[-10^6, 10^6]$. У свакој линији, бројеви су раздвојени празнином.

Излаз. (Излазне податке уписати у датотеку `dolina.out`) У први ред излаза потребно је уписати број l_{max}, највећу дужину неке скијашке долине. У следећих l_{max} линија дајте опис било које скијашке долине те дужине, тако да се у i-том реду налазе два цела броја x_i и y_i раздвојена празнином, и (x_i, y_i) представља i-то поље долине.

Пример 1.

`dolina.in`
```
3 4
2 6 7 16
1 4 3 20
9 8 17 12
```

`dolina.out`
```
9
3 1
3 2
2 2
2 1
1 1
1 2
1 3
1 4
2 4
```

Пример 2.

dolina.in	dolina.out
3 3	3
1 9 2	2 1
8 3 7	2 2
4 6 5	2 3

Проблем 4. Санке

Учитељица Данка је одлучила да своје ђаке одведе на санкање. Међутим, пошто је једино она била расположена за такву авантуру, и ђаци осталих разреда су одлучили да се придруже. И тако је она са N ученика отишла на оближњу планину. Када су дошли, она је замолила сваког ђака да јој каже одакле ће почети да се санка, и у ком правцу ће се спуштати. Како је Данка веома паметна учитељица, она је на основу рељефа планине и те две информације закључила и колико брзо ће ићи санке сваког ученика. Пошто веома дуго ради као учитељица, има довољно искуства да све ђаке може да контролише чак и ако се не помера (тј. све време стоји на само једном месту). Данка слабо види, а безбедност деце је на првом месту, па је одлучила да понесе наочаре. Али не било какве наочаре, већ специјалне наочаре на којима може да се намести колико далеко се види с њима. Да би била сигурна да је све у реду, одлучила је да подеси даљину наочара тако да бар у једном моменту види бар K ученика. Пошто сте Ви њен најбољи ученик, замолила вас је да јој израчунате колика је најмања могућа даљина на коју треба да подеси наочаре тако да то важи.

Улаз. (Улазни подаци се налазе у датотеци `sanke.in`) У првом реду датотеке се налазе два броја x и y ($0 \leq x, y \leq 10^4$) и они представљају координате уцитељице. У другом реду се налазе бројеви n ($5 \leq n \leq 1000$) и k ($1 \leq k \leq n$). У сваком од наредних n редова се налазе 4 броја : прва два броја $xs[i]$, $ys[i]$ ($0 \leq xs[i], ys[i] \leq 10^4$) означавају почетну позицију ђака, а друга два броја $xk[i]$, $yk[i]$ ($0 \leq xk[i], yk[i] \leq 10^4$) место на коме ће се ђак наћи после 1 секунде спуштања.

Напомена: сви ђаци се спуштају праволинијски, и ниједан ђак неће "прегазити" учитељицу.

Излаз. (Излазне податке уписати у датотеку `sanke.out`) У излазну датотеку треба исписати тражену најмању могућу даљину на 5 децимала.

Пример 1.

sanke.in	sanke.out
6 7	5.12050
5 4	
1 4 1 11	
4 12 10 10	
4 9 4 7	
2 4 10 4	
3 1 9 1	

Пример 2.

sanke.in	sanke.out
7 7	5.46418
5 3	
2 1 2 7	
4 1 4 5	
6 1 6 10	
8 1 8 3	
10 1 10 5	

Проблем 5. Задатак Светиљке

Дуж једног пута који је прав налази се Томина кућа. Дуж тог пута налази се и n светиљки, свака са

- одређеном координатом,
- дометом осветљавања
- потрошњом у једниници времена.

Томина кућа има координату 0. Координата светиљке означава позицију светиљке у односу на Томину кућу. Уколико је координата светиљке неки број x тада се

- уколико је x број већи или једнак 0 светиљка налази x метара десно од Томине куће
- уколико је x мање од 0 светиљка налази x метара лево од Томине куће

Домет осветљавања неке светиљка је број који означава колико метара лево и десно од те светиљке је пут осветљен помоћу ње. Уколико је домет неке светиљке неко x а њена координата неко y пут је од кординате $y - x$ до $y + x$ осветљен помоћу те светиљке.

Потрошња у јединици времена неке светиљке је број који означава колико џула енергије у јединици времена троши та светиљка када је упаљена.

На почетку су све светиљке угашене. Тома жели, тако што ће упалити неке светиљке, да осветли пут у континуитету од координате A до координате B ($A < B$) тако да потрошња енергије у јединици времена буде минимална.

(Тома жели само да све тачке пута почев од тачке са координатом A до тачке са координатом B буду осветљене, не занима га да ли ће још неки делови пута који су изван овог интервала бити осветљени)

Улаз. (Улазни подаци се налазе у датотеци `svetiljke.in`) У првом реду текстуалне датотеке налазе се редом бројеви n, A и B (n је природан број мањи или једнак 10^5 док су A и B цели бројеви који су већи или једнаки -2×10^9 и мањи или једнаки 2×10^9 и $A < B$) У сваком од следећих n редова налазе се по три цела броја који описују једну светиљку. Ти бројеви су редом координата светиљке (цео број већи или једнак -10^9 и мањи или једнак 10^9), домет осветљавања (природан број мањи или једнак 10^9) и потрошња у јединици времена (природан број мањи или једнак 20000). (У реду чији је редни број $i + 1$ налази се опис i-те светиљке)

Излаз. (Излазне податке уписати у датотеку `svetiljke.out`) У излазној датотеци треба уписати број који означава минималну потрошњу енергије у јединици времена која се мора потрошити да би се осветлио део пута који Тома жели уколико је то могуће. Уколико то није могуће уписати -1.

У 50% тест примера n је мање или једнако 5000.

Пример 1.

svetiljke.in	svetiljke.out
6 -10 20	5
-8 4 2	
25 4 7	
3 10 1	
0 15 4	
17 3 5	
16 4 2	

Могуће је осветлити део пута од координате -10 до координате 20. Минимална потрошња енергије у јединици времена која се мора имати да би се до извело је 5 и она се добија паљењем 1. , 3. и 6. светиљке. (1. светиљка осветљава интервал од -12 до -4 , 3. светиљка осветљава интервал од -7 до 13 а 6. светиљка осветљава интервал од 12 од 20)

Пример 2.

svetiljke.in	svetiljke.out

`3 4 20`
`8 5 10`
`8 4 7`
`19 4 10`
	`-1`

Проблем 6. Телепорт

Услед превеликог загађења ваздуха, људи су морали да напусте Земљу и сада живе у свемирским станицама. Наведене цивилне станице су нумерисане бројевима $1, 2, ..., m$. Ради одбране од ванземаљаца направљено је још n војних станица. Између сваке војне и сваке цивилне станице могуће је телепортовање у тачно једном смеру (ако је могуће телепортовање из станице A у станицу B није могуће из станице B у A). Из војне станице или није могуће телепортовање ни у једну цивилну станицу или је могуће телепортовање у низ узастопних цивилних станица (нпр. из војне станице могуће је телепортовање у цивилне станице $k, k+1, ..., p$ где је $1 \leq k \leq p \leq m$). Не може се телепортовати од једне цивилне станице до друге цивилне, као и од једне војне до друге војне.

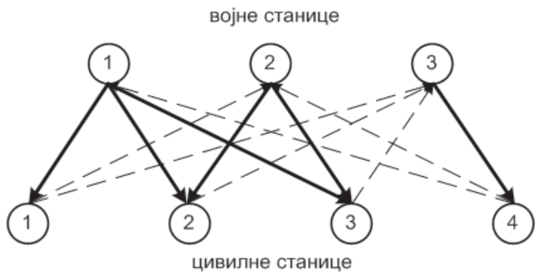
Међутим, ванземаљци су решили да освоје планету Земљу. Генерал Ђурић је открио да ванземаљци планирају да нападну неку станицу X, али не зна тачно коју. Он је проценио да ако војсци треба више од 3 телепортовања до нападнуте станице X она ће сигурно пасти у руке ванземаљаца. Зато је решио да нађе све војне станице са којих може да стигне да одбрани све остале станице и тамо распореди војску. Како је остало мало времена - замолио је вас младе програмере да му помогнете.

Улаз. (Улазни подаци се налазе у датотеци `teleport.in`) У првој линији улазне датотеке се налазе два броја n и m ($1 \leq n, m \leq 5 \times 10^5$) који редом представљају број војних и број цивилних станица. Затим следи n линија које описују везе између станица: у i-тој ($i = 1, 3, ..., n$) линији се налазе два цела броја $a[i]$ и $b[i]$ ($0 \leq a[i] \leq b[i] \leq m$) који представљају крајње цивилне станице у које се може телепортовати из i-те војне станице (из војне станице могуће је телепортовање у цивилне станице $a[i], a[i]+1, ..., b[i]$). Уколико је је $a[i] = b[i] = 0$ тада се из дате војне станице директно не може стићи ни у једну цивилну станицу.

Излаз. (Излазне податке уписати у датотеку `teleport.out`) У првом реду излазне датотеке се налази број војних станица које тражи генерал Ђурић, а у другом листа тих војних станица раздвојених једним размаком.

Пример 1.

teleport.in	teleport.out
3 4	2
1 3	1 3
2 3	
4 4	



Објашњење. Војна станица 2 не задовољава Ђурићеву процену, јер најкраћи путеви до цивилне станице 1 и војне станице 1 захтевају више од 3 телепортовања.

Пример 2.

teleport.in	teleport.out
4 4 1 4 1 3 0 0 2 4	1 1