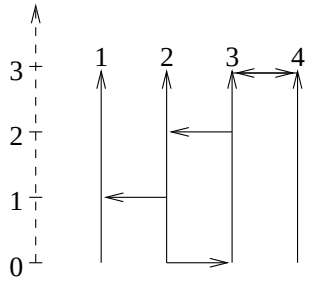
Jakub Radoszewski — Treść zadania, Opracowanie
Michał Brzozowski — Program

OI, Etap III, dzień drugi, 29.03.2007

Egzamin na prawo jazdy

Bajtocki egzamin na prawo jazdy odbywa się na placu, na którym znajduje się n prostych równoległych jednokierunkowych ulic, skierowanych z południa na północ. Każda z ulic ma długość m metrów, wszystkie ulice zaczynają się i kończą na tej samej wysokości. Są one ponumerowane od 1 do n, w kolejności z zachodu na wschód. Na placu znajduje się też p prostopadłych do nich jednokierunkowych ulic, skierowanych ze wschodu na zachód lub z zachodu na wschód i łączących pewne sąsiednie ulice biegnące z południa na północ. Może istnieć dowolnie wiele ulic biegnących ze wschodu na zachód lub z zachodu na wschód, łączących daną parę sąsiednich ulic biegnących z południa na północ. Możliwa jest też sytuacja, w której pewne dwie ulice, jedna biegnąca ze wschodu na zachód, a druga z zachodu na wschód, pokrywają się, tworząc ulicę dwukierunkową.



Przykładowy plac egzaminacyjny ($n = 4$, $m = 3$, $p = 5$).

W trakcie egzaminu egzaminator wybiera ulicę biegnącą z południa na północ, na początku której rozpocznie się egzamin oraz ulicę (również biegnącą z południa na północ), na końcu której egzamin ma się zakończyć. Zadaniem egzaminowanego jest przejechać — oczywiście zgodnie z kierunkami ulic jednokierunkowych — z miejsca, gdzie egzamin się zaczyna, do miejsca, gdzie się kończy.

Żeby uniknąć sytuacji, w której nie istniałaby droga z punktu początkowego egzaminu do punktu końcowego, egzaminatorzy zawsze jako ulicę startową wybierają jedną z takich ulic, z których początku da się dojechać do końca **dowolnej** *innej ulicy biegnącej z południa na północ.*

Praca egzaminatorów jest bardzo monotonna, gdyż ciągle rozpoczynają egzaminy na początku tych samych ulic. Dyrekcja postanowiła wybudować nowy plac, na podstawie istniejących już planów. Obliczono, że funduszy starczy na dodanie nie więcej niż k ulic biegnących ze wschodu na zachód lub z zachodu na wschód. Jednak należy tak dobrać te ulice, by przybyło jak najwięcej potencjalnych punktów początkowych egzaminu (na istniejącym planie mogą, ale nie muszą istnieć ulice będące punktami początkowymi). Dobudowane ulice muszą łączyć pewne pary sąsiednich ulic biegnących z południa na północ.

Zadanie

Napisz program, który:

- *wczyta ze standardowego wejścia opis planu placu egzaminacyjnego oraz liczbę k,*
- *wyznaczy maksymalną liczbę potencjalnych punktów początkowych egzaminu, jakie mogą się pojawić po dodaniu co najwyżej k ulic biegnących ze wschodu na zachód lub z zachodu na wschód,*
- *wypisze wynik na standardowe wyjście.*

Wejście

W pierwszym wierszu wejścia znajdują się cztery liczby całkowite n, m, p oraz k ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m,k \leqslant 100\,000$, $0 \leqslant p \leqslant 100\,000$), pooddzielane pojedynczymi odstępami i oznaczające odpowiednio: liczbę ulic biegnących z południa na północ, długość każdej z tych ulic, liczbę już istniejących ulic biegnących ze wschodu na zachód lub z zachodu na wschód oraz maksymalną liczbę ulic, jakie można dobudować. Ulice biegnące z południa na północ są ponumerowane od 1 do n, w kolejności z zachodu na wschód.

Kolejnych p wierszy zawiera po trzy liczby całkowite n_i, m_i oraz d_i ($1 \leqslant n_i < n$, $0 \leqslant m_i \leqslant m$, $d_i \in \{0,1\}$), pooddzielane pojedynczymi odstępami i opisujące i-tą ulicę biegnącą z zachodu na wschód (dla $d_i = 0$) bądź ze wschodu na zachód (dla $d_i = 1$). Ulica ta łączy ulice biegnące z południa na północ o numerach n_i i $n_i + 1$ oraz łączy się z nimi w punktach odległych o m_i metrów od ich początków.

Wyjście

Pierwszy i jedyny wiersz wyjścia powinien zawierać jedną liczbę całkowitą, równą maksymalnej liczbie nowych punktów początkowych egzaminu, jakie mogą pojawić się na placu po dobudowaniu co najwyżej k ulic biegnących ze wschodu na zachód lub z zachodu na wschód. Dobudowane ulice **nie muszą** *łączyć się z ulicami biegnącymi z południa na północ w punktach oddalonych o całkowitą liczbę metrów od ich początków. Dobudowane ulice biegnące ze wschodu na zachód mogą pokrywać się z ulicami biegnącymi z zachodu na wschód i odwrotnie. W ten sposób ulice jednokierunkowe zmieniają się w dwukierunkowe.*

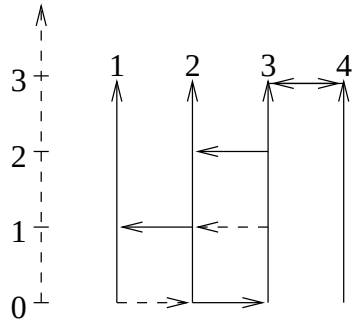
Przykład

Dla danych wejściowych:

```
4 3 5 2
2 0 0
2 2 1
3 3 1
1 1 1
3 3 0
```

poprawnym wynikiem jest:

```
2
```



Nowymi punktami początkowymi egzaminu mogą być, na przykład, początki ulic nr 1 i 3.

Rozwiązanie

Rozwiązanie wzorcowe

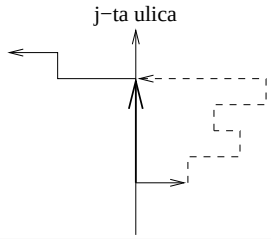
Analiza problemu

Wprowadźmy kilka pojęć, które pozwolą nam czytelniej zaprezentować rozwiązanie. Ulice prowadzące w kierunku południe-północ będziemy nadal nazywać *ulicami*, natomiast ulice prowadzące z zachodu na wschód i ze wschodu na zachód nazwiemy odpowiednio *łącznikami wschodnimi* oraz *łącznikami zachodnimi*. Powiemy, że łącznik jest *na wysokości k*, jeśli łączy punkty dwu sąsiednich ulic położone k jednostek od początku obu ulic. *Trasą* nazwiemy połączenie początku pewnej ulicy i z końcem pewnej ulicy j poprowadzone zgodnie z kierunkami ulic i łączników. Powiemy także, że ulica i-ta *jest połączona* z ulicą j-tą, jeśli istnieje trasa łącząca ulicę i-tą z ulicą j-tą. W takiej sytuacji będziemy również mówić, że z ulicy i-tej *można dojechać* do ulicy j-tej.

Przystąpmy teraz do opisu rozwiązania. Rozpoczniemy od kilku podstawowych spostrzeżeń, które pozwolą nam uprościć problem.

Po pierwsze zauważmy, że egzamin może rozpoczynać się na początku ulicy o numerze i wtedy i tylko wtedy, gdy jest ona połączona z ulicą pierwszą oraz z ulicą n-tą. Jeśli bowiem z ulicy i-tej (dla $i \in \{1,\ldots,n\}$) da się dojechać do ulicy pierwszej, to przerywając podróż odpowiednio wcześniej, możemy zakończyć ją na dowolnej spośród ulic $1,2,\ldots,i-1$; podobnie wygląda sytuacja z ulicami $i+1,\ldots,n-1,n$, które napotykamy, jadąc od ulicy i-tej do n-tej.

Drugie spostrzeżenie pozwala nam uprościć rozważane trasy. Zauważmy, że jadąc z ulicy i-tej do pierwszej, nie trzeba wykorzystywać łączników wschodnich. Gdyby egzaminowany przejechał takim łącznikiem, na przykład poprowadzonym pomiędzy ulicą j-tą a $(j+1)$-szą na wysokości k, to i tak w pewnym momencie musiałby wrócić do ulicy j-tej. Co więcej, znalazłby się wówczas na wysokości $k' > k$, czyli na północ od punktu, w którym poprzednio opuścił ulicę j-tą (gdyż nie ma ulic prowadzących z północy na południe). To oznacza, że mógł przejechać pomiędzy punktami o wysokości k i k' bezpośrednio ulicą j-tą, skracając całą trasę. Analogicznie rozumując, można pokazać, że w drodze z ulicy i-tej do n-tej egzaminowany nie musi wykorzystywać żadnych łączników zachodnich.



Rys. 1: Ilustracja spostrzeżenia drugiego: na drodze z ulicy j-tej do pierwszej nie opłaca się skręcać na wschód.

Poczynione spostrzeżenia pozwalają nam podzielić wyjściowy problem na dwa symetryczne podproblemy. Dla ulicy i ($1 \leqslant i \leqslant n$) wyznaczymy dwie wartości: l_i oraz p_i. Pierwsza z nich, to minimalna liczba łączników zachodnich, które trzeba dobudować, żeby połączyć

ulicę i-tą z ulicą pierwszą. Analogicznie p_i definiujemy jako minimalną liczbę łączników wschodnich, których dobudowanie pozwala przejechać z ulicy i-tej do ulicy n-tej.

Pozostańmy jeszcze chwilę przy ulicy i-tej. Zauważmy, że dobudowując l_i łączników zachodnich, dzięki którym pojawia się możliwość dojechania z ulicy i-tej do pierwszej, tworzymy także połączenie *każdej* spośród ulic $1,\ldots,i-1,i$ z ulicą pierwszą. Faktycznie, zaczynając egzamin na początku dowolnej z tych ulic, można jechać prosto na północ aż do momentu przecięcia trasy łączącej ulicę i-tą z ulicą pierwszą, i w tym właśnie punkcie włączyć się do niej. Podobne spostrzeżenie można także poczynić dla tras prowadzących na wschód. Dalej zauważmy, że $l_1 \leqslant l_2 \leqslant \cdots \leqslant l_n$ — jeżeli dobudowanie l_i (dla $i \in \{2,\ldots,n\}$) łączników zachodnich wystarcza, aby dojechać z ulicy i-tej do pierwszej, to z pewnością tyle samo łączników wystarczy, by dojechać tam z ulicy $(i-1)$-szej. Podobnie możemy uzasadnić, że $p_1 \geqslant p_2 \geqslant \cdots \geqslant p_n$.

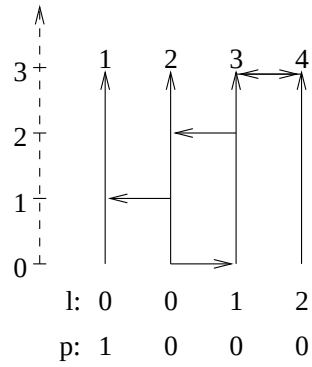
Ograniczenia zadania pozwalają nam dobudować jedynie k łączników. Ustalmy, że najwyżej L z nich to będą łączniki zachodnie. Niech x będzie największym numerem ulicy, dla której $l_x \leqslant L$ — budując co najwyżej $l_x \leqslant L$ łączników zachodnich, można połączyć ulice $1,\ldots,x$ z ulicą pierwszą. Co więcej, nie istnieje sposób, by dobudowując tylko L łączników, stworzyć połączenie ulicy $(x+1)$-szej (i kolejnych) z pierwszą! Po zbudowaniu l_x łączników zachodnich mamy jeszcze do wykorzystania $P = k - l_x$ łączników wschodnich. Zdefiniujmy dla P, analogicznie jak dla L, indeks $y \in \{1,\ldots,n\}$ jako najmniejszy indeks, dla którego $p_y \leqslant P$. Po dobudowaniu wszystkich k łączników łączących ulicę x-tą z pierwszą oraz ulicę y-tą z n-tą, uzyskujemy możliwość dojazdu do ulicy pierwszej z ulic $1,\ldots,x$ oraz do ulicy n-tej z ulic $y,\ldots,n$. To oznacza, że z ulic $y,\ldots,x$ (i tylko tych) da się dojechać zarówno do ulicy pierwszej, jak i n-tej. Początki tych ulic to wszystkie możliwe punkty startowe egzaminu. Ponieważ naszym zadaniem jest wyznaczenie liczby *nowych* punktów startowych — takich, które powstały dopiero po rozbudowie placu — zatem trzeba jeszcze rozpoznać, które spośród znalezionych punktów były dobrymi punktami startowymi także przed rozbudową. Jest to jednak bardzo proste: początek ulicy i-tej był dobrym punktem startowym egzaminu przed rozbudową wtedy i tylko wtedy, gdy $l_i = p_i = 0$.

Znamy już ogólną ideę rozwiązania. Teraz trzeba uzupełnić szczegóły — wyznaczyć wartości l_i, p_i, L, P oraz związane z nimi indeksy x i y. Zajmijmy się najpierw parametrami L i P. W ich przypadku nie będziemy stosować żadnych wyszukanych metod — po prostu przetestujemy wszystkie możliwe wartości parametru L od 1 do k. Zauważmy, że mając wyznaczone wszystkie wartości $l_1,\ldots,l_n$ oraz $p_1,\ldots,p_n$, indeks x możemy odnaleźć w czasie $O(\log n)$, stosując proste wyszukiwanie binarne wartości L w ciągu $l_1,\ldots,l_n$. Wartość l_x pozwala nam określić wartość $P = k - l_x$, a kolejne wyszukiwanie binarne w ciągu uporządkowanym $p_1,\ldots,p_n$ pozwala odnaleźć indeks y — także w czasie $O(\log n)$. Ponieważ $L \in \{0,\ldots,k\}$, to otrzymujemy złożoność czasową tej fazy $O(k \log n)$, czyli akceptowalną z punktu widzenia ograniczeń z zadania.

Zanim przejdziemy do wyznaczania wartości l_i oraz p_i, przedstawimy działanie opisanego algorytmu na przykładzie z treści zadania. Na rysunku 2 podane zostały wartości l_i oraz p_i dla $i = 1,2,3,4$.

Przeanalizujmy wszystkie możliwe wartości parametru L i odpowiadające im wartości parametrów P, x oraz y (pamiętając o tym, że w przykładzie $k = 2$):

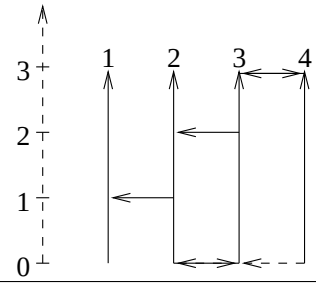
- $L = 0$, stąd $x = 2$, więc $l_x = 0$ i dalej $P = 2 - 0 = 2$ oraz $y = 1$; to daje łącznie 2 możliwe punkty startowe.



Rys. 2: Wartości l_i oraz p_i dla przykładu z treści zadania.

- $L = 1$, stąd $x = 3$, więc $l_x = 1$ i dalej $P = 2 - 1 = 1$ oraz $y = 1$; to daje łącznie 3 możliwe punkty startowe.
- $L = 2$, stąd $x = 4$, więc $l_x = 2$ i dalej $P = 2 - 2 = 0$ oraz $y = 2$; to znów daje łącznie 3 możliwe punkty startowe.

Ponieważ na początku mamy 1 punkt startowy (początek ulicy 2), to w najlepszym przypadku udaje się nam utworzyć 2 nowe punkty startowe egzaminu. Przykład optymalnego sposobu dobudowania łączników dla przypadku $L = P = 1$ jest zilustrowany na rysunku w treści zadania, natomiast dla przypadku $L = 2$, $P = 0$ — na rysunku 3.



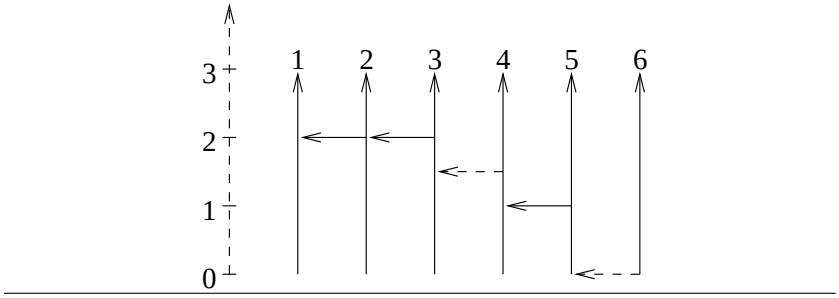
Rys. 3: Przykład optymalnej rozbudowy placu przez dobudowanie dwu łączników zachodnich (jeden z nich pokrywa się z już istniejącym łącznikiem wschodnim).

Na koniec warto wspomnieć, że rozważany etap rozwiązania można wykonać jeszcze szybciej — w czasie $O(n+k)$. Wystarczy zauważyć, że w miarę wzrostu parametru L, indeks x nie może maleć, gdyż ciąg l_i jest niemalejący. Podobnie, w miarę zmniejszania się wartości parametru P, indeks y nie może maleć, gdyż ciąg p_i jest nierosnący. Rozpoczynając obliczenia od $L = 0$, można poszukiwanie x rozpocząć od $x = 1$ i sprawdzać kolejne wartości, dopóki nie otrzymamy $l_x \leqslant L < l_{x+1}$. Znaleziona wartość l_x pozwala nam wyznaczyć $P = k - l_x$. Przystępujemy teraz do poszukiwania y — zaczynamy od $y = 1$ i zwiększamy tę wartość, dopóki nie otrzymamy $p_{y-1} > P \geqslant p_y$. Dostajemy czwórkę parametrów: $L = 0$, P, x i y, dla której wyznaczamy rozwiązanie. Przechodząc do $L = 1$, poszukiwanie x i y kontynuujemy od poprzednich wartości tych parametrów. Tak samo

postępujemy po kolejnych zwiększeniach L. W ten sposób w trakcie działania algorytmu każdy z parametrów x oraz y zostanie zwiększony o jeden łącznie $O(n)$ razy, co oznacza, że omawiana faza rozwiązania działa w zapowiadanej złożoności czasowej $O(k+n)$. Opisane usprawnienie, ze względu na czas konieczny do wykonania pozostałych faz, nie powoduje niestety istotnego wzrostu efektywności rozwiązania.

Wyznaczanie l_i oraz p_i

Skupimy się na ulicy i-tej i wyznaczeniu wartości l_i, dla pewnego $1 \leqslant i \leqslant n$ — obliczenie p_i wykonamy analogicznie. Zauważmy, że optymalna trasa łącząca tę ulicę z ulicą pierwszą zawiera dokładnie $i-1$ łączników zachodnich. Jeżeli l_i spośród tych łączników to nowo dobudowane, to $i-1-l_i$ łączników musiało być na placu przed rozbudową. „Stare" łączniki tworzą swoisty „ciąg niemalejący", który formalnie nazwiemy *ciągiem zachodnio-północnym* — jest to ciąg łączników zachodnich uporządkowanych tak, że każdy kolejny jest położony na zachód od poprzedniego i na nie mniejszej wysokości niż poprzedni. Łatwo spostrzec, że dowolny ciąg zachodnio-północny złożony z c łączników można rozbudować, dodając $i-c-1$ łączników zachodnich, tworząc trasę z ulicy i-tej do pierwszej. Trasę taką stworzymy najmniejszym kosztem, jeśli rozbudujemy najdłuższy ciąg zachodnio-północny, rozpoczynający się od tej ulicy — l_i wynosi wówczas $i-c-1$, gdzie c jest długością tego ciągu.

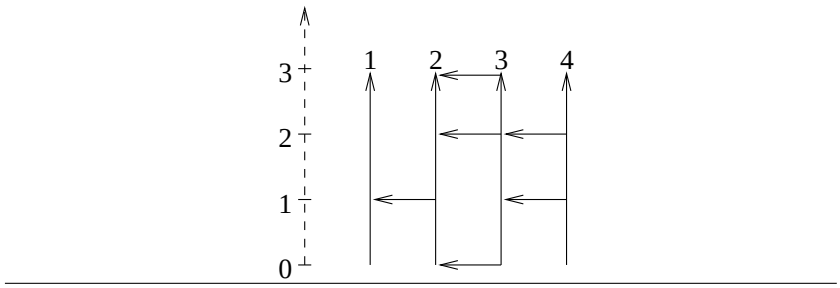


Rys. 4: Ciąg zachodnio-północny. Na rysunku $i=6$, liczba dobudowanych łączników to $l_i=2$, a najdłuższy ciąg zachodnio-północny ma długość $i-l_i+1=3$.

Pozostaje już tylko wyznaczyć długość najdłuższego ciągu zachodnio-północnego dla rozważanej ulicy. Spróbujmy przekształcić ten problem do poszukiwania najdłuższego podciągu niemalejącego w ciągu liczbowym. Rozważmy ciąg liczb, jaki otrzymamy, wypisując wysokości wszystkich łączników zachodnich, poczynając od łączników wychodzących z ulicy i-tej i poruszając się w kierunku zachodnim. Łączniki wychodzące z tej samej ulicy wypisujemy w kolejności malejących wysokości, czyli w kierunku z północy na południe. Rysunek 5 obrazuje ten sposób konstrukcji ciągu dla przykładowego placu egzaminacyjnego.

Podciągi niemalejące skonstruowanego ciągu liczb odpowiadają wzajemnie jednoznacznie ciągom zachodnio-północnym — kolejnym liczbom w podciągu odpowiadają bowiem łączniki położone na północ (gdyż podciąg jest niemalejący) i *ściśle* na zachód od poprzednich (ostrą nierówność uzyskujemy dzięki uporządkowaniu łączników wychodzących z tej samej ulicy w kolejności od północy na południe).

Istnieje wiele algorytmów wyznaczania długości najdłuższego podciągu niemalejącego zadanego ciągu — działają one w ogólnym przypadku w czasie $O(n \log n)$ dla ciągu długości



Rys. 5: Ciągiem liczbowym, skonstruowanym dla powyższego przykładu placu egzaminacyjnego ($i = 4$) jest: $2,1,3,2,0,1$.

n. Stąd wyznaczanie tych wartości dla każdego indeksu $i \in \{1,\ldots,n\}$ *osobno*, skończyłoby się przekroczeniem dopuszczalnych limitów czasu. Zauważmy jednak, że dowolny podciąg niemalejący danego ciągu jest podciągiem nierosnącym ciągu odwróconego. To z pozoru banalne spostrzeżenie pozwala efektywnie rozwiązać problem:

- dla kolejnych wartości indeksu $i = 1,2,\ldots,n$ możemy stopniowo przedłużać ciąg, dołączając do niego łączniki zachodnie wychodzące z ulicy i-tej oraz
- możemy dynamicznie aktualizować długość najdłuższego podciągu nierosnącego wydłużonego prefiksu, otrzymując wartość l_i.

Najdłuższy podciąg nierosnący

W niniejszym rozdziale skupimy się na ostatnio postawionym problemie i nie będziemy się już odwoływać do pojęć z treści zadania: zakładamy, że mamy ciąg liczbowy $a_1,\ldots,a_n$ i poszukujemy długości najdłuższych podciągów nierosnących każdego prefiksu tego ciągu — *prefiksem* długości k ciągu $a_1,\ldots,a_n$ nazywamy ciąg $a_1,\ldots,a_k$, który będziemy oznaczać $a[1..k]$.

Kluczowe informacje będziemy zapisywać w tablicy $t[1..n]$, aktualizując jej zawartość w miarę analizowania kolejnych elementów ciągu a, czyli rozważania coraz dłuższych prefiksów $a[1..k]$. Element $t[i]$ zdefiniujemy jako wartość *największego* elementu w ciągu $a[1..k]$, który jest ostatnim elementem pewnego podciągu nierosnącego długości i w prefiksie $a[1..k]$ (czyli spośród wszystkich podciągów nierosnących długości i ciągu $a[1..k]$ wybieramy ten, który kończy się największym elementem i element ten ustalamy jako wartość $t[i]$). Jeżeli taki element nie istnieje (gdyż nie ma żadnego podciągu o długości i w prefiksie $a[1..k]$), to przyjmujemy, że $t[i] = -\infty$. Przy takiej definicji tablicy t widzimy, że po każdym kroku jej konstrukcji (rozważeniu $a[1..k]$) długość najdłuższego podciągu nierosnącego $a[1..k]$ będzie po prostu równa największemu indeksowi i, dla którego $t[i] > -\infty$.

Zanim opiszemy algorytm obliczania tablicy t, przeanalizujmy przykład, który pozwoli nam zaobserwować jej kilka interesujących własności. Rozważmy mianowicie ciąg $a = 7,4,3,5,5,1,6,1$ długości 8 i zastanówmy się, jak powinna wyglądać tablica t, zbudowana dla kolejnych jego prefiksów:

- Wyjściowa tablica (dla pustego prefiksu ciągu a) wygląda następująco:

-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞

- W prefiksie $a[1..1]$ mamy już jeden podciągu rosnący (7) o długości jeden, którego ostatni element jest równy 7. Tablica t po pierwszym kroku powinna więc mieć postać:

7	-∞	-∞	-∞	-∞	-∞	-∞	-∞

- Wydłużamy prefiks do $a[1..2]$. Dołożony element $a_2 = 4$ nie jest lepszym niż poprzedni ciągiem jednoelementowym, ale pozwala przedłużyć ciąg (7) do $(7,4)$ i ustalić $t[2] = 4$:

7	4	-∞	-∞	-∞	-∞	-∞	-∞

- Wydłużamy prefiks o kolejny element $a_3 = 3$. Jest on, podobnie jak poprzedni, mniejszy od wszystkich dotychczas rozważonych:

7	4	3	-∞	-∞	-∞	-∞	-∞

- Kolejny element to $a_4 = 5$. Zauważmy, że za jego pomocą nie można poprawić ani $t[1]$ (gdyż $a_4 < t[1]$), ani $t[4]$ (największym elementem, jakim może się kończyć 3-elementowy ciąg nierosnący, jest $t[3] = 3$, czyli nie można tego ciągu wydłużyć za pomocą 5), ani $t[3]$ (z tego samego powodu, co poprzednio). Ponieważ jednak $t[1] \geqslant a_4$ oraz $a_4 > t[2]$, to widzimy, że w prefiksie istnieje podciąg $(7,5)$ i jest on lepszy niż poprzedni ciąg dwuelementowy $(7,4)$, gdyż kończy się większą wartością — możemy poprawić wartość $t[2]$:

7	5	3	-∞	-∞	-∞	-∞	-∞

- Wydłużamy prefiks do $a[1..5]$ i ponownie natrafiamy na element równy 5. Wykorzystując go, możemy tym razem poprawić $t[3]$, jako że w poprzednim kroku udało nam się zwiększyć $t[2]$ — mamy w ten sposób podciąg $(7,5,5)$; zauważmy, że żadnych innych pozycji w tablicy t nie możemy w tym kroku poprawić:

7	5	5	-∞	-∞	-∞	-∞	-∞

- Następny element $a_6 = 1$, co pozwala wreszcie zmodyfikować wartość $t[4]$, gdyż pojawił się podciąg $(7,5,5,1)$. Pozostałe elementy t z oczywistych względów pozostają bez zmian:

7	5	5	1	-∞	-∞	-∞	-∞

- Dochodzi kolejny element $a_7 = 6$. Tak dużego elementu nie można dołączyć do znalezionych dotychczas ciągów o niemalejących o 2, 3 oraz 4 elementach — możliwe natomiast jest dołączenie go do ciągu (7) i taki ciąg jest lepszy niż najlepszy dotychczas ciąg dwuelementowy $(7,5)$, gdyż kończy się większym elementem; to pozwala (już po raz trzeci) zmodyfikować $t[2]$:

7	6	5	1	-∞	-∞	-∞	-∞

- Został nam jeszcze ostatni element ciągu $a_8 = 1$. Jest on nie większy od elementów $t[1], t[2], t[3], t[4]$, więc nie nadaje się do poprawienia tych wartości; można jednak dołączyć go do znalezionego wcześniej ciągu czteroelementowego kończącego się elementem 1. Stąd ostateczna postać tablicy t to:

7	6	5	1	1	-∞	-∞	-∞

Zawartość tablicy t wyznaczona dla kolejnych prefiksów ciągu a pozwala stwierdzić, że długości najdłuższych podciągów nierosnących w tych prefiksach, to: 1, 2, 3, 3, 3, 4, 4, 5. W pierwszej chwili można by przypuszczać, że znaleźliśmy nie tylko długość, ale także sam

ciąg — najdłuższy podciąg nierosnący całego ciągu. Nie jest to prawdą! Zapisany w tablicy t ciąg $7,6,5,1,1$ *nie jest podciągiem* ciągu a! Jako proste, ale wartościowe, ćwiczenie polecamy Czytelnikowi zastanowienie się nad tym zjawiskiem.

Jesteśmy już gotowi, by sformułować i udowodnić kilka własności tablicy t, które prowadzą do prostego i efektywnego algorytmu jej konstrukcji:

1. Elementy zapisane w tablicy t po każdym kroku algorytmu są ustawione w kolejności nierosnącej. Aby się o tym przekonać, wystarczy zauważyć, że jeśli w rozważanym prefiksie $a[1..k]$ mamy podciąg długości i zakończony elementem $t[i]$, to w $a[1..k]$ mamy także podciąg niemalejący długości $i-1$ zakończony $t[i]$ — wystarczy z poprzedniego ciągu usunąć pierwszy element. Stąd $t[i-1] \geqslant t[i]$, o ile $t[i] > -\infty$. Jeżeli zaś $t[i] = -\infty$, to nierówność $t[i-1] \geqslant t[i]$ oczywiście zachodzi.

2. W każdym kroku algorytmu zmianie ulega *dokładnie jeden* element tablicy t. Aby to pokazać, rozważmy krok algorytmu, w którym rozszerzamy prefiks o a_k. Z poprzedniego spostrzeżenia wiemy, że istnieje taki indeks w tablicy t, powiedzmy j, że elementy $t[1],\ldots,t[j]$ są nie mniejsze od a_k, a elementy $t[j+1],\ldots,t[n]$ są mniejsze od a_k:

   - wiadomo, że żadnego spośród elementów $t[1],\ldots,t[j]$ nie można poprawić za pomocą a_k, ponieważ mamy już podciągi odpowiedniej długości, kończące się elementami nie mniejszymi (a więc lepszymi) niż a_k;
   - wartość $t[j]$ mówi nam, że w prefiksie $a[1..k-1]$ jest podciąg nierosnący długości j zakończony elementem $t[j]$; ponieważ $t[j] \geqslant a_i$, więc na końcu tego ciągu możemy dołączyć element a_k, otrzymując podciąg długości $j+1$ zakończony elementem $a_k > t[j+1]$; to oznacza, że możemy powiększyć wartość $t[j+1]$;
   - pozostałe elementy $t[j+1],\ldots,t[n]$ są mniejsze od a_k — to oznacza, że elementu a_k nie da się dołączyć na koniec żadnego podciągu długości większej niż j, czyli elementów $t[j+2],t[j+3],\ldots,t[n]$ nie można poprawić za pomocą a_k.

3. Ostatnie spostrzeżenie będzie prostym wnioskiem z dwóch poprzednich: skoro dodanie elementu a_k powoduje zmianę w dokładnie jednym miejscu tablicy t i miejsce to potrafimy dokładnie określić — jest to pierwszy element tablicy t mniejszy od a_k — to do wyznaczenia jego pozycji można użyć wyszukiwania binarnego. To pozwala wykonać jeden krok algorytmu w złożoności czasowej $O(\log n)$, a cały algorytm aktualizacji t — w złożoności czasowej $O(n \log n)$.

Przedstawmy teraz pseudokod opisanego algorytmu, w którym w każdym z n kroków (dla $i = 1,\ldots,n$) wypisujemy długość najdłuższego podciągu nierosnącego prefiksu $a[1..k]$ ciągu a:

```
1: for j := 1 to n do
2:     t[j] := −∞;
3: len := 0; { Długość najdłuższego podciągu nierosnącego. }
4: for k := 1 to n do
5: begin
6:     { Wyszukiwanie binarne elementu tablicy t, który należy zmienić. }
7:     { Złożoność czasowa O(log n). }
```

```
 8:   d := 1; g := len + 1; { dolna i górna granica wyszukiwania }
 9:   while d < g do
10:   begin
11:     s := ⌊(d+g)/2⌋;
12:     if t[s] ⩾ a_k then
13:       d := s + 1;
14:     else
15:       g := s;
16:   end
17:   { d (= g) jest szukaną pozycją w tablicy t. }
18:   t[d] := a_k;
19:   if d > len then
20:     { Musi zachodzić d = len + 1, czyli wynik się powiększył. }
21:     len := d;
22:   Wypisz_Długość_Podciągu(len);
23: end
```

Najbardziej skomplikowaną częścią powyższego algorytmu jest wyszukiwanie binarne pierwszego elementu tablicy t, który jest mniejszy od a_k (wiersze 8–16). Taki element musi zawsze istnieć, gdyż przed każdym krokiem algorytmu zachodzi $a_k > t[len+1] = -\infty$.

Podsumowanie

Uporządkujmy dotychczasowe rozważania i przypomnijmy w skrócie kolejne kroki rozwiązania wzorcowego:

- Dzielimy wszystkie łączniki na zachodnie (Z) i wschodnie (W). Dla pierwszej z tych grup wykonujemy wszystkie opisane niżej kroki. Natomiast łączniki wschodnie odbijamy symetrycznie względem dowolnej ulicy (czyli prostej pionowej), co odpowiada odwróceniu numeracji ulic, od których rozpoczynają się łączniki. Złożoność czasowa tego kroku to $O(p)$.

- Sortujemy łączniki ze zbioru Z w porządku malejącym numerów ulic, od których się rozpoczynają. Łączniki wychodzące z tej samej ulicy sortujemy w kolejności z północy na południe. Złożoność czasowa tego kroku to $O(p \log p)$, jeżeli zastosujemy jeden z efektywnych algorytmów sortowania, np. sortowanie przez scalanie.

- Tworzymy ciąg liczbowy złożony z wysokości wszystkich łączników ze zbioru Z we wspomnianej kolejności. Następnie odwracamy otrzymany ciąg i, za pomocą opisanego algorytmu o złożoności czasowej $O(p \log p)$, dla każdego prefiksu ciągu odwróconego znajdujemy długość najdłuższego podciągu nierosnącego. Na tej podstawie wyznaczamy wartości l_i. Całkowita złożoność czasowa tego kroku to $O(p \log p + n)$.

- Wykonując analogiczne do powyższych obliczenia dla zbioru W, otrzymujemy, w takim samym czasie, wartości p_i.

- Obliczywszy wartości l_i oraz p_i, przeglądamy w $k+1$ fazach wszystkie możliwe wartości parametru L od 0 do k, obliczając dla każdej z nich pozostałe parametry: x, P oraz y.

- Końcowy wynik wyznaczamy jako maksimum po wszystkich fazach z wartości $x-y+1-q_{x,y}$, gdzie $x-y+1$ odpowiada liczbie punktów startowych, które możemy otrzymać po rozbudowie placu, a $q_{x,y}$ jest liczbą starych punktów startowych wśród nich (czyli liczbą indeksów z przedziału $[y,x]$, dla których $l_i = p_i = 0$). Ze względu na wykorzystanie wyszukiwania binarnego, złożoność czasowa tego podpunktu to $O(k \log n)$. Wspomnieliśmy także o szybszym sposobie realizacji tej fazy, w którym przy wyznaczaniu kolejnych wartości parametrów x oraz y korzystamy z wiedzy o ich poprzednich wartościach i nigdy ich nie zmniejszamy. W ten sposób można zredukować złożoność czasową tej fazy do $O(n+k)$.

Całkowita złożoność czasowa powyższego rozwiązania to $O(p \log p + n + k)$. Zostało ono zaimplementowane w plikach `egz.cpp`, `egz2.pas` oraz `egz3.c`.

Inne rozwiązania

Wszystkie przewidziane przez Jury Olimpiady poprawne rozwiązania alternatywne, podobnie zresztą jak wszystkie rozwiązania zawodników, które uzyskały dodatnią punktację, opierają się w mniejszym lub większym stopniu na tych samych pomysłach, co rozwiązanie wzorcowe. Występuje w nich faza wyznaczania wartości analogicznych do l_i i p_i oraz faza, w której na ich podstawie jest obliczany wynik. Każdą z opisanych faz można zaimplementować mniej efektywnie niż zostało to uczynione w rozwiązaniu wzorcowym. Jeżeli chodzi o pierwszą fazę, to istnieje cała gama algorytmów wyznaczania długości najdłuższego podciągu nierosnącego w kwadratowej złożoności czasowej (w naszym przypadku jest to $O(p^2)$), jak choćby naiwna implementacja opisanego algorytmu wyznaczania tablicy t. Również drugą fazę można zrealizować nieoptymalnie, przeglądając wszystkie możliwe początki i końce przedziałów potencjalnych punktów startowych i dla każdej takiej pary (x,y) sprawdzając, czy $p_x + l_y \leqslant k$. Taki sposób implementacji drugiej fazy ma koszt czasowy $O(n^2)$.

Testy

Zadanie było sprawdzane na 10 zestawach danych wejściowych. W poniższej tabelce n oznacza liczbę ulic w teście, m — długości ulic, p — liczbę już wybudowanych łączników zachodnich i wschodnich, a k — maksymalną liczbę łączników, które można dobudować. Wreszcie przez w oznaczono wynik dla testu, to znaczy maksymalną liczbę nowych punktów startowych, jakie można uzyskać, dobudowując co najwyżej k łączników. We wszystkich testach, w których $p > 0$, istniejące łączniki zachodnie i wschodnie zostały rozmieszczone na placu w sposób losowy.

Nazwa	n	m	p	k	w	Opis
egz1a.in	4	3	5	2	2	prosty test poprawnościowy
egz1b.in	100	100	200	100	30	większy test poprawnościowy
egz2a.in	10	1	0	8	0	prosty test bez łączników
egz2b.in	10	1	0	9	1	prosty test bez łączników
egz2c.in	10	10	10	10	9	prosty test, w którym wszystkie parametry wejściowe są równe 10
egz2d.in	100000	1	0	99999	1	duży test bez łączników
egz2e.in	100	50	150	100	27	większy test losowy
egz3a.in	100	100	300	100	42	prosty test poprawnościowy
egz3b.in	100	100	400	100	46	prosty test poprawnościowy
egz3c.in	100	100	500	100	50	prosty test poprawnościowy
egz4a.in	500	1000	500	300	0	średniej wielkości test z wynikiem zerowym
egz4b.in	500	1000	100	800	323	średniej wielkości test z wynikiem niezerowym
egz5.in	1000	50000	100000	1000	740	średniej wielkości test
egz6.in	10000	50000	50000	15000	5536	duży test
egz7.in	20000	50000	20000	30000	10353	duży test
egz8a.in	50000	100000	100000	18000	0	duży test, wynik zerowy
egz8b.in	50000	100000	100000	50000	637	duży test, wynik niezerowy
egz9.in	100000	10000	100000	99999	622	duży test
egz10.in	100000	80000	100000	100000	618	test (prawie) maksymalnej wielkości

Rozwiązania wolniejsze przechodziły na zawodach pewne spośród testów 1-5, w zależności od tego, które fazy rozwiązania wzorcowego były zaimplementowane nieoptymalnie.