Bartosz Walczak
Treść zadania, Opracowanie

Marcin Michalski
Program

OI, Etap III, dzień pierwszy, 29.03.2006

Estetyczny tekst

Rozważmy dowolny tekst złożony z n słów ponumerowanych od 1 do n. Dowolny podział tego tekstu na k wierszy reprezentujemy za pomocą takiego ciągu liczb $(a_1, a_2, \ldots, a_{k-1})$, że słowa o numerach od 1 do a_1 znajdują się w pierwszym wierszu, słowa o numerach od $a_1 + 1$ do a_2 znajdują się w drugim wierszu itd., a słowa o numerach od $a_{k-1} + 1$ do n znajdują się w ostatnim, k-tym wierszu.

Każde słowo ma określoną długość (wyrażoną liczbą znaków). Długość słowa o numerze x oznaczamy przez $\mathit{length}(x)$. Ponadto każde dwa sąsiednie słowa w wierszu są oddzielone odstępem szerokości jednego znaku. Długością wiersza nazywamy sumę długości wszystkich słów w tym wierszu powiększoną o liczbę odstępów między nimi. Długość wiersza o numerze w oznaczamy przez $\mathit{line}(w)$. Oznacza to, że jeżeli w wierszu o numerze w znajdują się słowa o numerach od i do j włącznie, to długość tego wiersza wynosi:

$$\mathit{line}(w) = \mathit{length}(i) + \mathit{length}(i+1) + \ldots + \mathit{length}(j) + (j - i)$$

Dla przykładu, rozważmy tekst złożony z 4 słów o długościach kolejno 4, 3, 2 i 5 oraz jego podział $(1, 3)$ na 3 wiersze. Wówczas długość pierwszego wiersza wynosi 4, drugiego — 6, a trzeciego — 5:

```
XXXX     (1. wiersz)
XXX XX   (2. wiersz)
XXXXX    (3. wiersz)
```

Współczynnikiem estetyczności podziału danego tekstu na k wierszy nazywamy liczbę wyrażoną wzorem:

$$|\mathit{line}(1) - \mathit{line}(2)| + |\mathit{line}(2) - \mathit{line}(3)| + \ldots + |\mathit{line}(k-1) - \mathit{line}(k)|$$

W szczególności, jeżeli podział zajmuje tylko jeden wiersz, jego współczynnik estetyczności jest równy 0.

Im mniejszy jest współczynnik estetyczności, tym bardziej estetyczny jest dany podział. Rozpatrujemy tylko takie podziały, w których długość żadnego wiersza nie przekracza pewnej stałej liczby m. Spośród wszystkich takich podziałów danego tekstu na dowolną liczbę wierszy poszukujemy podziału najbardziej estetycznego, czyli o minimalnym współczynniku estetyczności. W podanym powyżej przykładzie współczynnik estetyczności podziału jest równy 3 i jest to minimalna wartość współczynnika estetyczności dla $m = 6$ lub $m = 7$.

Zadanie

Napisz program, który:

- *Wczyta ze standardowego wejścia liczby m i n oraz długości kolejnych słów.*
- *Wyznaczy minimalny współczynnik estetyczności dla tych podziałów, w których długość żadnego wiersza nie przekracza m.*
- *Wypisze wynik na standardowe wyjście.*

Wejście

Pierwszy wiersz standardowego wejścia zawiera liczby całkowite m *i* n*,* $1 \leqslant m \leqslant 1\,000\,000$*,* $1 \leqslant n \leqslant 2\,000$*, oddzielone pojedynczym odstępem. Drugi i ostatni wiersz wejścia zawiera* n *liczb całkowitych będących długościami kolejnych słów,* $1 \leqslant \mathsf{length}(i) \leqslant m$ *dla* $i = 1, 2, \ldots, n$*, pooddzielanych pojedynczymi odstępami.*

Wyjście

Pierwszy i jedyny wiersz standardowego wyjścia powinien zawierać jedną liczbę całkowitą: minimalny współczynnik estetyczności dla tych podziałów, w których długość żadnego wiersza nie przekracza m*.*

Przykład

Dla danych wejściowych:

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

poprawnym wynikiem jest:

```
3
```

Dla danych wejściowych:

```
4 2
1 2
```

poprawnym wynikiem jest:

```
0
```

Rozwiązanie

Rozpocznijmy od pewnych uproszeń zadania, które ułatwią prezentację rozwiązania. Po pierwsze zauważmy, że możemy zapomnieć o odstępach pomiędzy wyrazami w wierszu, jeśli zwiększymy o 1 długość każdego słowa oraz ograniczenie długości wiersza m. Wówczas długość wiersza możemy liczyć po prostu jako sumę długości występujących w nim wyrazów.

Wprowadźmy także następujące terminy: *minimalnym podziałem* danego tekstu będziemy nazywali minimalny współczynnik estetyczności podziału tego tekstu na wiersze, przy czym minimum bierzemy albo po wszystkich podziałach, albo po pewnych szczególnych podziałach (będzie to wynikało z kontekstu). Ponadto *prefiksem tekstu* będziemy nazywać tekst złożony z początkowych wyrazów tekstu, czyli ze słów o numerach $1, \ldots, k$ dla pewnego $1 \leqslant k \leqslant n$.

Proste rozwiązanie dynamiczne

Możemy teraz przystąpić do poszukiwania efektywnego rozwiązania problemu. Naturalną metodą, która przychodzi na myśl w przypadku zadań optymalizacyjnych podobnych do naszego, jest programowanie dynamiczne. Postępując zgodnie z tą techniką postaramy się wyznaczyć minimalne podziały wszystkich prefiksów danego tekstu, otrzymując ostatecznie minimalny podział całości.

Jednak aby wyznaczyć minimalny podział określonego prefiksu, nie wystarczy obliczyć wcześniej jedynie minimalne podziały dla wszystkich krótszych prefiksów. Dla każdego z nich musimy także uwzględnić różne możliwe długości ostatniego wiersza, by w zależności od tej wartości ustalić współczynnik estetyczności po dołożeniu kolejnego wiersza.

Niech więc $suma_x^y$ oznacza sumę długości słów o numerach $x, \ldots, y$, a $wynik_x^y$ niech oznacza minimalny podział prefiksu złożonego ze słów $1, \ldots, y$, w którym ostatni wiersz składa się ze słów o numerach $x, \ldots, y$. Jeżeli $suma_x^y \leqslant m$, to wartość $wynik_x^y$ możemy obliczyć następująco:

$$wynik_x^y = \begin{cases} 0 & \text{dla } y \geqslant x = 1, \\ \min\{wynik_z^{x-1} + |suma_x^y - suma_z^{x-1}| \ : \ 1 \leqslant z < x\} & \text{dla } y \geqslant x \geqslant 2 \end{cases} \tag{1}$$

W przypadku, gdy $suma_x^y > m$, przyjmujemy $wynik_x^y = \infty$.
Wszystkie wartości $suma_x^y$ możemy obliczyć w czasie $\Theta(n^2)$:

for $x := 1$ **to** n **do**
begin
  $suma_x^x := length(x)$;
  for $y := x+1$ **to** n **do**
    $suma_x^y := suma_x^{y-1} + length(y)$;
end

Następnie korzystając ze wzoru (1) możemy zapisać prosty algorytm, pozwalający wyznaczyć końcowy wynik $\min\{wynik_x^n \ : \ 1 \leqslant x \leqslant n\}$ w czasie $\Theta(n^3)$. Jednakże przy podanym w treści zadania ograniczeniu na n taka złożoność nie jest satysfakcjonująca.

Przyspieszenie obliczeń

Okazuje się, że algorytm można przyspieszyć, jeśli zmienimy nieco sposób obliczania wartości *wynik*. W tym celu przekształcimy wzór (1) wyodrębniając przypadki, gdy ostatni wiersz jest krótszy od poprzedniego:

$$\begin{aligned} wynik_x^y = \min\{&\min\{wynik_z^{x-1} + suma_z^{x-1} - suma_x^y \ : \ suma_z^{x-1} > suma_x^y\}, \\ &\min\{wynik_z^{x-1} - suma_z^{x-1} + suma_x^y \ : \ suma_z^{x-1} \leqslant suma_x^y\}\} = \\ = \min\{&\min\{wynik_z^{x-1} + suma_z^{x-1} \ : \ suma_z^{x-1} > suma_x^y\} - suma_x^y, \\ &\min\{wynik_z^{x-1} - suma_z^{x-1} \ : \ suma_z^{x-1} \leqslant suma_x^y\} + suma_x^y\} \end{aligned}$$

Następnie rozważmy ustalone wartości x i y. Niech q będzie maksymalną liczbą w przedziale $1 \leqslant q < x$, dla której $suma_q^{x-1} > suma_x^y$. Jeżeli takie q nie istnieje, to przyjmujemy $q = 0$.

Wówczas nierówność $suma_z^{x-1} > suma_x^y$ zachodzi wtedy i tylko wtedy, gdy $z \leqslant q$, jako że wartość $suma_a^b$ maleje wraz ze wzrostem wartości a. Ponadto wprowadźmy oznaczenia:

$$mniejsze_x^y = \min\{wynik_a^y + suma_a^y : a \leqslant x\}$$
$$wieksze_x^y = \min\{wynik_a^y - suma_a^y : a \geqslant x\}$$

Przyjmijmy przy tym $mniejsze_0^y = \infty$ oraz $wieksze_{y+1}^y = \infty$. Z poczynionych obserwacji wynika, że:

$$\min\{wynik_z^{x-1} + suma_z^{x-1} : suma_z^{x-1} > suma_x^y\} = \min\{wynik_z^{x-1} + suma_z^{x-1} : z \leqslant q\} = mniejsze_q^{x-1}$$
$$\min\{wynik_z^{x-1} - suma_z^{x-1} : suma_z^{x-1} \leqslant suma_x^y\} = \min\{wynik_z^{x-1} - suma_z^{x-1} : z > q\} = wieksze_{q+1}^{x-1}$$

Stąd ostatecznie wzór (1) sprowadza się do następującej postaci:

$$wynik_x^y = \min\{mniejsze_q^{x-1} - suma_x^y, wieksze_{q+1}^{x-1} + suma_x^y\} \tag{2}$$

Możemy więc zapisać schemat rozwiązania wzorcowego:

1: **for** $y := 1$ **to** n **do**
2: **begin**
3:    **for** $x := 1$ **to** y **do** oblicz $wynik_x^y$
4:    **for** $x := 1$ **to** y **do** oblicz $mniejsze_x^y$
5:    **for** $x := y$ **downto** 1 **do** oblicz $wieksze_x^y$
6: **end**

Jeżeli dla danych x i y znamy wartość q, to dzięki formule (2) możemy obliczyć $wynik_x^y$ w linii 3 w czasie stałym dla każdego x. Wartości $mniejsze_x^y$ i $wieksze_x^y$ dla każdego x również łatwo obliczyć w czasie stałym — wystarczy wykorzystać $mniejsze_{x-1}^y$ do obliczenia $mniejsze_x^y$ oraz $wieksze_{x+1}^y$ do obliczenia $wieksze_x^y$ (stąd odwrócona kolejność pętli w linii 5):

$$mniejsze_x^y = \min(mniejsze_{x-1}^y, wynik_x^y + suma_x^y)$$

$$wieksze_x^y = \min(wieksze_{x+1}^y, wynik_x^y - suma_x^y)$$

Pozostaje zastanowić się, w jaki sposób wyznaczyć indeks q. Można zastosować wyszukiwanie binarne wykorzystując przy tym monotoniczność funkcji *suma*. Wówczas wyznaczenie każdej wartości q, a tym samym obliczenie wartości $wynik_x^y$, wymaga czasu $\Theta(\log n)$, więc cały algorytm działa w złożoności czasowej $\Theta(n^2 \log n)$. Można jednak postąpić sprytniej. Zauważmy, że jeżeli przy ustalonym y zwiększamy x (tak jak w linii 3), to zmniejsza się $suma_x^y$, więc wartość q może się tylko zwiększyć lub pozostać niezmieniona. Tak więc wykonując obliczenia w linii 3, możemy dla kolejnych wartości x poszukiwać wartości q, począwszy od indeksu q znalezionego w poprzedniej iteracji pętli. W ten sposób wyznaczymy wszystkie wartości q dla danego y w łącznym czasie liniowym. Zatem każdy krok głównej pętli wykonamy w czasie liniowym, czyli otrzymamy algorytm o złożoności $\Theta(n^2)$.

Jako podsumowanie opisu algorytmu zamieszczamy jego pseudokod:

```
for y := 0 to n do
   mniejsze^y_0 := ∞;
for y := 1 to n do
   wieksze^y_{y+1} := ∞;
for y := 1 to n do
begin
   { tablica wynik }
   q := 0;
   for x := 1 to y do
   begin
      if suma^y_x > m then
      begin
         wynik^y_x := ∞;
         continue;
      end
      if x = 1 then
      begin
         wynik^y_x := 0;
         continue;
      end
      while (q + 1 ⩽ x − 1) and (suma^{x−1}_{q+1} > suma^y_x) do
         q := q + 1;
      wynik^y_x := min(mniejsze^{x−1}_q − suma^y_x, wieksze^{x−1}_{q+1} + suma^y_x);
   end
   { tablica mniejsze }
   for x := 1 to y do
      mniejsze^y_x := min(mniejsze^y_{x−1}, wynik^y_x + suma^y_x);
   { tablica wieksze }
   for x := y downto 1 do
      wieksze^y_x := min(wieksze^y_{x+1}, wynik^y_x − suma^y_x);
end
```

Implementacja rozwiązania wzorcowego znajduje się w plikach `est.cpp` i `est0.pas` (rozwiązanie działające w czasie $\Theta(n^2)$) oraz w `ests1.cpp` i `ests4.pas` (rozwiązanie działające w czasie $\Theta(n^2 \log n)$).

Testy

Poniżej znajduje się opis zestawu danych testowych, na których były oceniane rozwiązania zawodników. Pary testów 8a i 8b, 9a i 9b, 10a i 10b oraz 11a i 11b zostały pogrupowane. Rozwiązania o złożoności $\Theta(n^2)$ lub $\Theta(n^2 \log n)$ przechodziły wszystkie testy. Rozwiązania o złożoności $\Theta(n^3)$ nie przechodziły grup testów 5, 6, 8, 9, 10, 11. Niepoprawne rozwiązania, które polegały na zachłannym umieszczaniu w wierszach maksymalnej liczby słów, nie przechodziły prawie żadnego z testów.

Nazwa	m	n	Opis
est1.in	10	4	mały, poprawnościowy
est2.in	11	8	mały, poprawnościowy
est3.in	99	3	mały, poprawnościowy
est4.in	1 000	80	losowy
est5.in	3 000	1 000	zawierający dużo jednoliterowych słów i na końcu kilka długich losowych
est6.in	650	1 303	zawierający dużo jednoliterowych słów i na końcu 3 dłuższe
est7.in	1 000	1 400	losowy; zawierający na początku na zmianę jedno- i 999-literowe słowa, których nie można połączyć w wiersze
est8a.in	100	2 000	losowy; zawierający na zmianę krótkie i długie słowa — niektóre pary można połączyć w wiersze, inne nie
est8b.in	100 000	1 754	zawierający na początku i na końcu długie słowo, a w środku dużo krótkich losowych
est9a.in	1 000 000	2 000	losowy; zawierający krótkie słowa, które się mieszczą w jednym wierszu
est9b.in	1 000	2 000	jak wyżej, ale słowa się nie mieszczą w jednym wierszu
est10a.in	1 000 000	2 000	losowy
est10b.in	999 752	2 000	losowy; zawierający w środku 500 długich słów, a na początku i na końcu po 750 krótkich
est11a.in	13	7	mały, poprawnościowy
est11b.in	999 999	2 000	zawierający wiele krótkich słów poprzedzielanych długimi, odpowiedź 0