Szymon Acedański — Treść zadania, Opracowanie
Karol Cwalina — Program

Okresy słów

Napisem *nazywamy każdy skończony ciąg małych liter alfabetu angielskiego. W szczególności może to być ciąg pusty. Zapis* $A = BC$ *oznacza, że* A *jest napisem powstałym przez sklejenie napisów* B *i* C *(w tej kolejności). Napis* P *jest* **prefiksem** *napisu* A*, jeżeli istnieje taki napis* B*, że* $A = PB$*. Inaczej mówiąc, prefiksy* A *to początkowe fragmenty* A*. Jeśli dodatkowo* $P \neq A$ *oraz* P *nie jest napisem pustym, to mówimy, że* P *jest* **prefiksem właściwym** A*.*

Napis Q *jest* **okresem** A*, jeśli* Q *jest prefiksem właściwym* A *oraz* A *jest prefiksem (niekoniecznie właściwym) napisu* QQ*. Przykładowo, napisy* `abab` *i* `ababab` *są okresami napisu* `abababa`*.* **Maksymalnym okresem** *napisu* A *nazywamy najdłuższy z jego okresów, lub napis pusty, jeśli* A *nie posiada okresu. Dla przykładu, maksymalnym okresem napisu* `ababab` *jest* `abab`*. Maksymalnym okresem* `abc` *jest napis pusty.*

Zadanie

Napisz program, który:

- *wczyta ze standardowego wejścia długość napisu oraz napis,*
- *wyznaczy sumę długości maksymalnych okresów wszystkich jego prefiksów,*
- *wypisze wynik na standardowe wyjście.*

Wejście

W pierwszym wierszu standardowego wejścia zapisana jest jedna liczba całkowita k *(*$1 \leqslant k \leqslant 1\,000\,000$*) — długość napisu. W kolejnym wierszu znajduje się ciąg dokładnie* k *małych liter alfabetu angielskiego — napis.*

Wyjście

W pierwszym i jedynym wierszu standardowego wyjścia Twój program powinien zapisać jedną liczbę — sumę długości maksymalnych okresów wszystkich prefiksów napisu zadanego na wejściu.

Przykład

Dla pliku wejściowego `okr.in`*:*

```
8
babababa
```

poprawnym wynikiem jest plik wyjściowy `okr.out`*:*

```
24
```

Rozwiązanie

Niniejsze zadanie zawiera kolejny z serii problemów związanych z algorytmami tekstowymi. Na dodatek ograniczenia na dane wejściowe pozwalają domyślać się, że oczekiwane rozwiązanie ma złożoność liniową. Stąd rodzi się podejrzenie, że pomocnym może okazać się algorytm Knutha, Morrisa, Pratta (KMP), który już niejednokrotnie przydawał się w rozwiązaniach problemów wymagających analizy napisów. Jest to słuszne podejrzenie, ale zacznijmy od początku. Spróbujmy najpierw znaleźć szybką metodę na wyznaczenie maksymalnego okresu całego słowa.

Zauważmy, że zdefiniowany w treści zadania *maksymalny okres* rzadko jest tym, co zazwyczaj uważamy za okres słowa. Otóż maksymalny okres, jeśli w ogóle istnieje, ma długość będącą co najmniej połową długości słowa:

$$\underbrace{\text{słowookresowesłowookresowesłowookresowe}}_{\textit{maksymalny okres}}\text{słowookresowe}$$

Rys. 1: Maksymalny okres ma długość będącą co najmniej połową długości słowa

Warto także zaznaczyć, że nie zawsze maksymalny okres jest wielokrotnością standardowo rozumianego, minimalnego okresu:

$$\underbrace{\text{słowookresowesłowookresowesłowookresowesłowookre}}_{\textit{maksymalny okres}}\text{s}$$

Rys. 2: Maksymalny okres nie musi być wielokrotnością minimalnego okresu

W powyższych przykładach maksymalny okres zajmuje większą część słowa. Skoncentrujmy się teraz na pozostałej części. Jest to początkowy fragment „drugiej kopii” maksymalnego okresu — musi być on jednocześnie prefiksem, jak i sufiksem słowa.

Definicja 1 *Prefikso-sufiksem* słowa nazywamy każdy jego prefiks, który jest jednocześnie jego sufiksem.

Łatwo spostrzec, że to, co zostaje po odcięciu maksymalnego okresu to po prostu *minimalny niepusty prefikso-sufiks* słowa. Co więcej, jeżeli długość najkrótszego prefikso-sufiksu słowa jest niewiększa od połowy długości słowa, to po odcięciu tego prefikso-sufiksu zawsze otrzymujemy maksymalny okres słowa.

$$\text{słowookresowesłowookresowesłowookresowe}\underbrace{\text{słowookresowe}}_{\textit{minimalny prefikso-sufiks}}$$

$$\text{słowookresowesłowookresowesłowookresowesłowookre}\underbrace{\text{s}}_{\textit{minimalny prefikso-sufiks}}$$

Rys. 3: Pozostałość po obcięciu maksymalnego okresu

Jak wyznaczyć ten prefikso-sufiks? Możemy próbować skorzystać z funkcji prefiksowej $p(i)$, znanej z algorytmu KMP (jego opis można znaleźć na przykład w książce [13] czy [18]). Funkcja ta, dla danej liczby i, zwraca długość *maksymalnego właściwego prefikso-sufiksu i-*literowego prefiksu całego słowa. Zwróćmy uwagę na te z jej własności, które będą pomocne przy znajdowaniu *minimalnego prefikso-sufiksu*.

1. Każdy prefikso-sufiks słowa jest prefiksem maksymalnego prefikso-sufiksu (ponieważ jest prefiksem całego słowa).

2. Analogicznie, każdy prefikso-sufiks słowa jest sufiksem maksymalnego prefikso-sufiksu.

3. Z tego wniosek, że każdy prefikso-sufiks jest prefikso-sufiksem maksymalnego prefikso-sufiksu i dalej, każdy prefikso-sufiks słowa jest prefikso-sufiksem dowolnego dłuższego prefikso-sufiksu tego słowa.

4. Niech n oznacza długość słowa. Aby znaleźć minimalny prefikso-sufiks, iterujemy funkcję prefiksową p, otrzymując kolejno: $p(n)$ — długość maksymalnego prefikso-sufiksu, $p(p(n))$ — długość krótszego prefikso-sufiksu, $p(p(p(n)))$ — długość jeszcze krótszego, itd. W ten sposób dostajemy w kolejności malejącej długości wszystkich prefikso-sufiksów słowa, więc ostatnia niezerowa wartość jest długością minimalnego niepustego prefikso-sufiksu.

Stosując opisaną metodę otrzymujemy algorytm, który działa w czasie $O(n^2)$.

My jednak chcielibyśmy skonstruować coś szybszego. Przyjrzyjmy się zatem dokładniej funkcji p. Może uda nam się szybko wyznaczyć podobną funkcję, nazwijmy ją s, która dla danego argumentu i zwraca długość *minimalnego* prefikso-sufiksu i-literowego prefiksu słowa wejściowego? Okazuje się to możliwe, jeśli najpierw policzymy funkcję p. Wtedy możemy skorzystać z następującej własności.

$$s(i) = \begin{cases} p(i) & \text{gdy } p(p(i)) = 0 \\ s(p(i)) & \text{w przeciwnym przypadku} \end{cases} \tag{1}$$

W pierwszym przypadku istnieje tylko jeden prefikso-sufiks, więc jest on najkrótszy. W drugim przypadku, gdy słowo ma więcej niż jeden prefikso-sufiks, to szukany najkrótszy prefikso-sufiks jest najkrótszym prefikso-sufiksem najdłuższego prefikso-sufiksu słowa.

Korzystając z funkcji p oraz z powyższej zależności możemy w czasie $O(n)$ obliczyć wartości $s(i)$ dla $i = 1,\ldots,n$. Można łatwo pokazać, że jeżeli dane słowo ma właściwy niepusty prefikso-sufiks dłuższy niż połowa danego słowa, to słowo to posiada również niepusty prefikso-sufiks niedłuższy niż połowa tego słowa. Stąd jeżeli słowo posiada niepusty najkrótszy prefikso-sufiks, to prefiks powstały przez jego odcięcie będzie maksymalnym okresem słowa. Obserwacja ta prowadzi nas do wniosku, jeśli dla pewnego $i = 1,\ldots,n$ zachodzi $s(i) = 0$, to maksymalnym okresem prefiksu długości i jest słowo puste, a w przeciwnym wypadku słowo długości $i - s(i)$. Sumując tak zdefiniowane wartości otrzymujemy ostateczny wynik.

Testy

Do oceny zostało użytych 14 testów, przedstawionych w poniższej tabeli.

Nazwa	n	Opis
okr1.in	10	prosty test poprawnościowy
okr2.in	36	prosty test poprawnościowy
okr3.in	441	$(\mathtt{a}^{k-1}\mathtt{b})^k$ z nieco zmienioną ostatnią ćwiartką
okr4.in	1 024	$(\mathtt{a}^{k-1}\mathtt{b})^k$, zmiany j.w.
okr5.in	501 264	$(\mathtt{a}^{k-1}\mathtt{b})^k$, zmiany j.w.
okr6.in	970 225	$(\mathtt{a}^{k-1}\mathtt{b})^k$, zmiany j.w.
okr7.in	100 387	w^k dla losowego wzorca w, zmiany j.w.
okr8.in	299 809	w^k dla losowego wzorca w, zmiany j.w.
okr9.in	599 976	w^k dla losowego wzorca w, zmiany j.w.
okr10.in	949 721	w^k dla losowego wzorca w, zmiany j.w.
okr11.in	129 962	$(w^k v)^{k^2}$ dla losowych w i v
okr12.in	331 332	$(w^k v)^{k^2}$ dla losowych w i v
okr13.in	614 619	$(w^k v)^{k^2}$ dla losowych w i v
okr14.in	923 387	$(w^k v)^{k^2}$ dla losowych w i v