Wojciech Rytter
Treść zadania, Opracowanie

Jan Kanty Milczek
Program

Dostępna pamięć: 64 MB. *OI, etap I, 6.10–3.11.2014*

Kwadraty

W tym zadaniu rozważamy rozkłady dodatnich liczb całkowitych na sumy **różnych** *kwadratów dodatnich liczb całkowitych (dalej będziemy je nazywać w skrócie rozkładami). Na przykład, liczba 30 ma dwa rozkłady:* $1^2 + 2^2 + 5^2 = 1^2 + 2^2 + 3^2 + 4^2 = 30$*, natomiast dla liczby 8 nie istnieje żaden rozkład.*

Interesuje nas odpowiedź na pytanie, jak duża musi być największa liczba w rozkładzie danej liczby n. Innymi słowy, chcemy wyznaczyć wartość $k(n)$ *będącą minimum z największych liczb występujących we wszystkich rozkładach liczby n. Dla uproszczenia przyjmijmy, że jeśli liczby n nie da się rozłożyć, to* $k(n) = \infty$*. Dla przykładu,* $k(1) = 1$, $k(8) = \infty$, $k(30) = 4$, $k(378) = 12$, $k(380) = 10$.

Liczbą **przerośniętą** *nazwiemy taką liczbę x, dla której istnieje liczba* $y > x$ *taka, że* $k(y) < k(x)$*. Z poprzedniego przykładu widzimy więc, że 378 jest liczbą przerośniętą.*

Dla zadanej liczby n oblicz $k(n)$ *oraz liczbę liczb przerośniętych mniejszych lub równych n.*

Wejście

W pierwszym i jedynym wierszu standardowego wejścia znajduje się jedna liczba całkowita n ($1 \leqslant n \leqslant 10^{18}$)*. W testach wartych 45% punktów zachodzi dodatkowy warunek* $n \leqslant 50\,000\,000$*, w podzbiorze tych testów wartym 30% punktów warunek* $n \leqslant 1\,000\,000$*, a w podzbiorze tych testów wartym 20% punktów warunek* $n \leqslant 1000$*.*

Wyjście

Twój program powinien wypisać na standardowe wyjście dwie liczby całkowite oddzielone pojedynczym odstępem: pierwsza z nich to $k(n)$*, a druga to liczba liczb przerośniętych z przedziału od 1 do n. Jeśli* $k(n) = \infty$*, to zamiast pierwszej liczby należy wypisać znak* `-` *(minus).*

Przykład

Dla danych wejściowych:
```
30
```
poprawnym wynikiem jest:
```
4 15
```

natomiast dla danych wejściowych:
```
8
```
poprawnym wynikiem jest:
```
- 5
```

Rozwiązanie

W zadaniu tym trzeba było trochę poeksperymentować komputerowo na małych liczbach, żeby zauważyć pewne własności. Dlatego też zadanie pojawiło się na I etapie, by zawodnicy mieli czas na takie eksperymenty. Na początek trzeba zauważyć, że wartości $k(n)$ jesteśmy w stanie wyznaczyć algorytmem plecakowym (programowanie dynamiczne). Do zbioru przedmiotów dodajemy po kolei przedmioty o rozmiarach $1^2, 2^2, \ldots$ i dla każdej liczby m zapisujemy najmniejszy przedmiot, po dodaniu którego jest możliwe uzyskanie liczby m jako sumy rozmiarów różnych przedmiotów dodanych do tej pory. Jeżeli chcemy wyznaczyć wartości $k(1), \ldots, k(n)$, to dodawanie przedmiotów powinniśmy przerwać w momencie, w którym kolejny dodawany przedmiot miałby rozmiar większy od n. W tym momencie uzyskaliśmy rozwiązanie siłowe, które co prawda na zawodach dostawało mało punktów (zgodnie z treścią zadania – 20 punktów), jednak przyda nam się w rozwiązaniu wzorcowym.

Aby zbadać, jak zachowuje się funkcja k, możemy uruchomić nasze rozwiązanie siłowe wyznaczające jej początkowe wartości np. dla $n \leqslant 1000$. Na początku zastanówmy się nad bardziej podstawowym pytaniem: dla jakich liczb n zachodzi $k(n) = \infty$? Przyglądając się wynikom naszego rozwiązania siłowego, możemy zauważyć, że największa znaleziona liczba bez żadnego przedstawienia w postaci sumy różnych kwadratów liczb całkowitych dodatnich to 128, a wszystkie liczby z przedziału $[129, 1000]$ posiadają jakieś przedstawienie. To może nam nasunąć podejrzenie, że nie istnieją liczby większe od 128, dla których $k(n) = \infty$. Spróbujemy zatem udowodnić następujący fakt:

Fakt 1. *Jeżeli $n \geqslant 129$, to $k(n) \neq \infty$.*

Zanim jednak zabierzemy się do jego dowodu, zastanówmy się jeszcze chwilę i popatrzmy na wyniki wyprodukowane przez rozwiązanie siłowe. Zdefiniujmy następujące funkcje:

$$\sigma(m) \;=\; 1^2 + 2^2 + 3^2 + \ldots + m^2, \quad ind(n) \;=\; \min\{\, m \;:\; \sigma(m) \;\geqslant\; n \,\}.$$

Jest jasne, że $k(n) \geqslant ind(n)$, innymi słowy $\sigma(k(n)) \geqslant n$ (ponieważ suma pewnych spośród liczb $1^2, 2^2, \ldots, k(n)^2$ nie może przekraczać sumy wszystkich tych liczb). Jeżeli popatrzymy na wyniki naszego rozwiązania siłowego, dojdziemy do wniosku, że dla odpowiednio dużych liczb (dajmy na to, większych od 400) często zachodzi $k(n) = ind(n)$, a dla pozostałych liczb zachodzi $k(n) = ind(n) + 1$. Udowodnimy następujący fakt:

Fakt 2. *Jeżeli $m \geqslant 11$ oraz $n \in [129, \sigma(m)]$, to $k(n) \leqslant m + 1$.*

Zauważmy, że z tak sformułowanego stwierdzenia wynika w sposób natychmiastowy prawdziwość poprzedniego faktu.

Dowód: Dowód przeprowadzimy przez indukcję. Dla $m = 11$ teza jest prawdziwa na mocy wyników uzyskanych przez nasze rozwiązanie siłowe.

Niech teraz $m > 11$ oraz $n \in [129, \sigma(m)]$. Będziemy chcieli udowodnić tezę indukcyjną dla m, wiedząc, że jest prawdziwa dla $m - 1$. Rozpatrzmy dwa przypadki:

Przypadek 1: $n \leqslant 128 + (m+1)^2$**.** W tym przypadku mamy $128 \leqslant \sigma(m-3)$ (jako że już $\sigma(8) = 204$) oraz $(m+1)^2 \leqslant (m-2)^2 + (m-1)^2$ (co zachodzi dla $m \geqslant 8$). Po zsumowaniu stronami tych dwóch nierówności otrzymujemy $128 + (m+1)^2 \leqslant \sigma(m-1)$, zatem $n \in [129, \sigma(m-1)]$. Na mocy założenia indukcyjnego $k(n) \leqslant m \leqslant m + 1$, więc w tym przypadku teza istotnie zachodzi.

Przypadek 2: $n \geqslant 129+(m+1)^2$**.** Wiemy, że $n-(m+1)^2 \in [129, \sigma(m)-(m+1)^2]$, zatem tym bardziej $n-(m+1)^2 \in [129, \sigma(m-1)]$. Na mocy założenia indukcyjnego $k(n-(m+1)^2) \leqslant m$, zatem $n-(m+1)^2 = a_1^2+a_2^2+\ldots+a_l^2$, dla pewnych $1 \leqslant a_1 < a_2 < \ldots < a_l \leqslant m$. Stąd wynika, że $n = a_1^2+a_2^2+\ldots+a_l^2+(m+1)^2$, co jest przedstawieniem spełniającym warunki zadania. Ostatecznie uzyskujemy $k(n) \leqslant m+1$, co dowodzi tezy indukcyjnej. ∎

Wiemy już całkiem dużo na temat wartości funkcji k; dla *małych* liczb (nazwijmy tak liczby całkowite równe co najwyżej $\sigma(11) = 506$) mamy je *explicite* wyznaczone przez nasze rozwiązanie siłowe, a dla *dużych* n (nazwijmy tak liczby większe od $\sigma(11)$) zachodzi $k(n) \in \{ind(n), ind(n)+1\}$.

Do poznania wszystkich wartości funkcji k pozostaje nam jeszcze jedynie stwierdzić, dla których *dużych* liczb zachodzi $k(n) = ind(n)$, a dla których $k(n) = ind(n)+1$. Ustalmy zatem pewną liczbę n, dla której $k(n) = ind(n)$. Wtedy $n = a_1^2+a_2^2+\ldots+a_l^2$, gdzie $1 \leqslant a_1 < a_2 < \ldots < a_l = ind(n)$. Niech teraz liczby $b_1, b_2, \ldots, b_p$ będą to wszystkie liczby całkowite z przedziału $[1, ind(n)]$ różne od liczb $a_1, a_2, \ldots, a_l$. Skoro liczby $a_1, a_2, \ldots, a_l$ oraz $b_1, b_2, \ldots, b_p$ to sumarycznie wszystkie liczby od 1 do $ind(n)$, to

$$\sigma(ind(n)) = a_1^2+\ldots+a_l^2+b_1^2+\ldots+b_p^2 = n+b_1^2+\ldots+b_p^2,$$

zatem $\sigma(ind(n))-n = b_1^2+\ldots+b_p^2$, czyli $k(\sigma(ind(n))-n) < \infty$. Stąd wniosek, że jeżeli dla jakiejś *dużej* liczby n zachodzi $k(\sigma(ind(n))-n) = \infty$, to wtedy $k(n) = ind(n)+1$.

Sprawdźmy natomiast, co się dzieje, jeżeli $k(\sigma(ind(n))-n) < \infty$. Jako że $n > \sigma(ind(n)-1)$, to

$$\sigma(ind(n))-n < ind(n)^2 < \sigma(ind(n)-1).$$

Stąd na mocy faktu 2 otrzymujemy, że $k(\sigma(ind(n))-n) \leqslant ind(n)$, co oznacza, że $\sigma(ind(n))-n = b_1^2+\ldots+b_p^2$ dla pewnych $1 \leqslant b_1 < \ldots < b_p \leqslant ind(n)$. Jeżeli zatem za liczby $a_1, \ldots, a_l$ weźmiemy wszystkie liczby od 1 do $ind(n)$ różne od $b_1, \ldots, b_p$, to wtedy

$$a_1^2+\ldots+a_l^2 = \sigma(ind(n))-(b_1^2+\ldots+b_p^2) = \sigma(ind(n))-(\sigma(ind(n))-n) = n.$$

Zatem w istocie n ma przedstawienie w postaci sumy kwadratów różnych liczb całkowitych nie większych niż $ind(n)$, czyli $k(n) = ind(n)$, co dopełnia charakteryzacji wartości funkcji k. Podsumowując:

Fakt 3 (Metoda obliczania $k(n)$).
Jeżeli n jest małą liczbą, to $k(n)$ jest wyznaczone przez algorytm siłowy. Jeżeli n jest dużą liczbą i $\sigma(ind(n))-n > 128$, to $k(n) = ind(n)$. Jeżeli n jest dużą liczbą i $\sigma(ind(n))-n \leqslant 128$, to $k(n) = ind(n)$, jeżeli $k(\sigma(ind(n))-n) \neq \infty$, a $k(n) = ind(n)+1$ w przeciwnym przypadku.

Przejdźmy teraz do drugiej części zadania, czyli do zliczania liczb *przerośniętych*. Udowodnimy następujące stwierdzenie:

Fakt 4. *Liczba n* **nie** *jest przerośnięta wtedy i tylko wtedy, gdy $k(n) = ind(n)$.*

Dowód: Jeżeli $k(n) = ind(n)$ oraz $p > n$, to $k(p) \geqslant ind(p) \geqslant ind(n) = k(n)$. Tak więc n w istocie nie jest liczbą przerośniętą.

Mamy także $\sigma(ind(n)) \geqslant n$, zatem jeżeli $k(n) > ind(n)$, to $k(\sigma(ind(n))) = ind(n) < k(n)$. W tym przypadku n rzeczywiście jest liczbą przerośniętą, co dowodzi postawionego faktu. ■

Na przykład liczba 522 jest przerośnięta, ponieważ w tym przypadku mamy: $ind(522) = 12$, $\sigma(11) = 506 < 522 \leqslant \sigma(12) = 650$, ale $k(522) = 13$.

Dla $m \geqslant 12$ zdefiniujmy teraz

$$\Delta_m = (\sigma(m-1), \sigma(m)],\ L_m = (\sigma(m-1), \sigma(m)-129],\ R_m = (\sigma(m)-129, \sigma(m)].$$

Już wiemy, że jeżeli $n \in L_m$ oraz n jest *duże*, to n nie jest przerośnięte. Ponadto dla każdego $m \geqslant 12$ w przedziale R_m jest tyle samo liczb przerośniętych, gdyż jeżeli $n \in R_m$, to to, czy n jest przerośnięte, zależy jedynie od tego, czy $k(\sigma(m) - n) = \infty$. Konkretnie, w każdym takim przedziale znajduje się tyle liczb przerośniętych, ile istnieje takich m, że $k(m) = \infty$, czyli dokładnie 31 (wszystkie takie liczby znamy, gdyż wiemy, że są one równe co najwyżej 128).

Stąd wynika już prosty algorytm na zliczanie liczb przerośniętych równych co najwyżej n. Jeżeli n jest *małe*, to wynik odczytujemy z rozwiązania siłowego. W przeciwnym przypadku przedział $[1, n]$ dzielimy na liczby *małe*, czyli $[1, \sigma(11)]$, w którym znamy liczbę liczb przerośniętych (wynosi ona 175), oraz na przedział $[\sigma(11)+1, n]$, który to z kolei dzielimy na przedziały $\Delta_{12} \cup \ldots \cup \Delta_{ind(n)-1} \cup S$, gdzie $S = (\sigma(ind(n)-1), n]$. W każdym z przedziałów od Δ_{12} do $\Delta_{ind(n)-1}$ znajduje się dokładnie 31 liczb przerośniętych. Jeżeli $n \leqslant \sigma(m)-129$, to $S \subseteq L_{ind(n)}$, zatem w takim przypadku w S nie ma żadnych liczb przerośniętych. Jeżeli jednak $n \geqslant \sigma(m)-128$, to $S = L_{ind(n)} \cup (S \cap R_{ind(n)})$ i w S znajduje się tyle liczb przerośniętych, co w $S \cap R_{ind(n)}$. Przedział ten zawiera co najwyżej 129 liczb, zatem możemy przejrzeć wszystkie jego elementy i dla każdego z nich stwierdzić, korzystając z naszej charakteryzacji (fakty 3 i 4), czy jest on liczbą przerośniętą.

Takim oto sposobem otrzymaliśmy algorytm, który na początku musi wykonać pewien *preprocessing*, aby wyznaczyć wartości funkcji k dla *małych* argumentów, potem wyznaczyć $ind(n)$ i na koniec ewentualnie przeiterować po przedziale o długości co najwyżej 129, stwierdzając, które liczby z niego są przerośnięte. Czas działania naszego algorytmu jest zatem stały, nie licząc czasu wyznaczenia $ind(n)$. Jako że $\sigma(m) = \Theta(m^3)$, to $ind(n) = \Theta(\sqrt[3]{n})$, zatem iterowanie po kolejnych wartościach $\sigma(1), \sigma(2), \ldots$ da nam algorytm działający w czasie $O(\sqrt[3]{n})$. To w zupełności wystarcza, aby zmieścić się w limicie czasowym. Możemy też skorzystać ze znanego wzoru $\sigma(m) = \frac{m\cdot(m+1)\cdot(2m+1)}{6}$ i wyszukać wartość $ind(n)$ binarnie, co pozwoli nam otrzymać algorytm działający w czasie $O(\log n)$.