Liczby niepotęgowe (rozwiązanie)

Autor zadania: **Karol Pokorski**
Opracowanie: **Kajetan Ramsza, Dominik Wawszczak**
Opis rozwiązania: **Bartosz Kostka**

Olimpiada Informatyczna Juniorów

Zamiast znajdować N-tą liczbę niepotęgową, skupimy się na odpowiedzi na pytanie, ile jest liczb niepotęgowych w przedziale od 1 do X dla danego X. Potrafiąc odpowiedzieć na takie pytanie, możemy zastosować wyszukiwanie binarne[1] do rozwiązania oryginalnego problemu w następujący sposób. Dla punktu pośrodku rozważanego przedziału w wyszukiwaniu binarnym M wystarczy zapytać, ile jest liczb niepotęgowych mniejszych lub równych M. Jeśli jest ich co najmniej N, wiemy, że odpowiedź jest mniejsza lub równa M. W przeciwnym wypadku odpowiedź jest większa niż M.

Skupmy się zatem na pytaniu, jak policzyć liczbę liczb niepotęgowych w przedziale od 1 do X. Możemy spróbować wygenerować wszystkie liczby niepotęgowe, jednak dość szybko zauważymy, że jest ich na tyle dużo, iż nie zmieścimy się ani w czasie, ani w pamięci, aby je wszystkie wygenerować lub zapisać. Odwrotnie, liczb potęgowych jest również sporo (choć zauważalnie mniej). Zastanówmy się, ile ich dokładnie jest. Od 1 do 10^{18} będziemy mieli z grubsza $\sqrt{10^{18}} = 10^9$ kwadratów, $\sqrt[3]{10^{18}} = 10^6$ sześcianów, $\sqrt[4]{10^{18}} \approx 3162277$ liczb w czwartej potędze, itd. Zwróćmy uwagę, że z powyższych rozważań wynika, iż odpowiedź może być większa niż 10^{18}.

Widzimy zatem, że gdybyśmy nie musieli się przejmować kwadratami, powinniśmy być w stanie wygenerować i zapisać w pamięci wszystkie liczby potęgowe o wykładniku większym niż dwa. Wprowadźmy zatem następujące pojęcia:

- mała liczba potęgowa – liczba, która jest kwadratem liczby całkowitej większej niż 1, np. $4 = 2^2$ lub $49 = 7^2$;
- duża liczba potęgowa – liczba, która jest potęgą liczby całkowitej większej niż 1, z wykładnikiem większym niż 3, niebędąca małą liczbą potęgową (tj. kwadratem), np. $27 = 3^3$ lub $32 = 2^5$. Dużą liczbą potęgową nie będzie natomiast $64 = 2^6$, ponieważ 64 jest także równe 8^2.

Definiując te zbiory w powyższy sposób, gwarantujemy, że nie mają one wspólnych elementów, czyli nie policzymy żadnych liczb potęgowych wielokrotnie. Aby policzyć zatem liczbę liczb potęgowych od 1 do X, wystarczy policzyć osobno liczbę małych liczb potęgowych i dużych liczb potęgowych i zsumować te wyniki.

Policzenie małych liczb potęgowych jest prostsze. Liczbę kwadratów od 1 do X możemy łatwo policzyć, korzystając z funkcji pierwiastka kwadratowego (np. od 1 do 50 mamy $\lfloor\sqrt{50}\rfloor = 7$ kwadratów: $1, 4, 9, 16, 25, 36, 49$). Oznaczenie $\lfloor a \rfloor$ oznacza tutaj podłogę liczby, czyli największą liczbę całkowitą nie większą niż a, lub prościej mówiąc: a zaokrąglone w dół. Pamiętając, że 1 nie jest małą liczbą potęgową (zgodnie z definicją), wystarczy policzyć $\lfloor\sqrt{X}\rfloor - 1$, aby otrzymać liczbę małych liczb potęgowych od 1 do X.

Skupmy się teraz na policzeniu dużych liczb potęgowych. Zastosujemy tutaj technikę *preprocessingu*, czyli wykonamy pewne obliczenia przed właściwym programem, które później wykorzystamy. Nasz preprocessing będzie polegał na wygenerowaniu wszystkich dużych liczb potęgowych. Aby to zrobić, przeiterujemy się po wszystkich liczbach całkowitych od 2 do odpowiednio dużej liczby (zastanowimy się za moment, co to właściwie oznacza) i dodamy wszystkie potęgi tej liczby X, rozpoczynając od trzeciej potęgi: X^3, X^4, X^5, ..., ponownie, do odpowiednio dużej liczby. Musimy tutaj sprawdzić, czy żadna z tych potęg nie jest przypadkiem kwadratem (aby przypadkowo nie dodać małej liczby potęgowej). Możemy to zrobić, sprawdzając, czy pierwiastek kwadratowy z danej potęgi jest liczbą całkowitą.

Zastanówmy się teraz, jak wybrać odpowiednio dużą liczbę do powyższych rozważań. Można zauważyć, że musi ona być równa co najmniej maksymalnemu wynikowi. Nie powinna być też znacznie większa od tego wyniku, abyśmy nie wykonywali niepotrzebnych obliczeń. Najpierw zastanówmy się, jaki może być najwyższy wykładnik, jaki może być użyty. Oczywiście będzie on użyty przy najmniejszej liczbie całkowitej, czyli 2. Sprawdzając, że $2^{60} > 1.15 \cdot 10^{18} > 10^{18}$, widzimy, że największy wykładnik nie przekroczy 60. Co więcej, wiedząc, że kwadratów będzie *z grubsza* niewiele więcej niż 10^9, jeśli pomnożymy te dwie liczby, tj. $60 \cdot 10^9$, i dodamy do maksymalnej liczby, jaką możemy dostać na wejściu (10^{18}), otrzymamy dobre oszacowanie na maksymalną odpowiedź, tj. $10^{18} + 6 \cdot 10^{10}$. Rozważanie to miało na celu pokazanie, jak można oszacować tę wartość. Można też było eksperymentalnie wyznaczyć tę liczbę. Możemy tutaj wspomnieć, że wynosi ona $1\,000\,000\,001\,001\,003\,331$.

Mając teraz wygenerowane wszystkie duże liczby potęgowe, możemy je posortować. Teraz w tak posortowanej tablicy lub liście, aby powiedzieć, ile jest dużych liczb potęgowych mniejszych lub równych X, wystarczy ponownie użyć algorytmu wyszukiwania binarnego.

[1] https://pl.wikipedia.org/wiki/Wyszukiwanie_binarne

Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2022/2023, 2023/2024, 2024/2025".

Podsumujmy teraz całe rozwiązanie. Na początku programu generujemy wszystkie duże liczby potęgowe. Następnie wczytujemy kolejne zapytania. Dla każdego zapytania posiłkujemy się wyszukiwaniem binarnym, aby znaleźć N-tą liczbę potęgową. Aby to zrobić, w zapytaniu wyszukiwania binarnego musimy policzyć liczbę liczb potęgowych (i odjąć od liczby wszystkich liczb, aby otrzymać liczbę liczb niepotęgowych). Aby to zrobić, sprawdzamy, ile jest dużych liczb potęgowych (ponownie korzystając z algorytmu wyszukiwania binarnego) i małych liczb potęgowych (używając funkcji pierwiastka kwadratowego).

Nie jest to jedyne rozwiązanie tego zadania, jednak zgodnie z przewidywaniami Jury było to jedno z najczęściej implementowanych rozwiązań. Dla zainteresowanych zachęcamy do zapoznania się z zasadą włączeń i wyłączeń[2] i zastanowienia się, jakie rozwiązanie może powstać na jej podstawie.

lic.cpp

```cpp
#include "bits/stdc++.h"

using namespace std;

const long long LIMIT = 1'000'000'060'000'000'000LL;

bool czy_jest_kwadratem(long long X) {
  long long pierwiastek = floor(sqrtl(X));
  return pierwiastek * pierwiastek == X;
}

// Liczby potęgowe podzielimy na:
// małe liczby potęgowe (z wykładnikiem równym 2),
// duże liczby potęgowe (z wykładnikiem większym niż 2).

vector<long long> duze_potegowe;

// Generujemy duże liczby potegowe (z wykładnikiem większym niż 2).
void wygeneruj_duze_potegowe() {
  set<long long> potegowe;
  long long baza = 2;
  while (true) {
    long long aktualna_potega = baza * baza * baza;
    if (aktualna_potega > LIMIT)
      break;
    while (aktualna_potega <= LIMIT) {
      // Chcemy mieć jedynie duże liczby potęgowe, więc
      // sprawdzamy czy przez przypadek nie jest kwadratem
      // (tj.  małą liczbą potęgową).
      if (!czy_jest_kwadratem(aktualna_potega)) {
        potegowe.insert(aktualna_potega);
      }
      if (LIMIT / baza < aktualna_potega)
        break; // Unikamy przekroczenia zakresu long long.
      aktualna_potega *= baza;
    }
    baza++;
  }
  duze_potegowe = vector<long long>(potegowe.begin(), potegowe.end());
}

// Liczymy ile jest małych liczb potęgowych mniejszych od X.
long long ile_mniejszych_malych(long long X) { return floor(sqrtl(X - 1)) - 1; }

// Zwraca którą liczbą potęgową jest X.
```

[2] https://pl.wikipedia.org/wiki/Zasada_wÅĆÄĚczeÅĎ_i_wyÅĆÄĚczeÅĎ

Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2022/2023, 2023/2024, 2024/2025".

```cpp
long long ktora_potegowa(long long X) {
  long long mniejszych_malych = ile_mniejszych_malych(X);
  long long mniejszych_duzych =
      lower_bound(duze_potegowe.begin(), duze_potegowe.end(), X) -
      duze_potegowe.begin();
  return X - mniejszych_malych - mniejszych_duzych;
}

// Używamy wyszukiwania binarnego do znalezienia odpowiedzi.
long long wyszukiwanie_binarne(long long N) {
  long long a = 1, b = LIMIT;
  while (b - a > 1) {
    long long mid = (a + b) / 2;
    // Pytamy, którą liczbą niepotęgową jest mid.
    if (ktora_potegowa(mid) <= N) {
      a = mid;
    } else {
      b = mid;
    }
  }
  return a;
}

int main() {
  wygeneruj_duze_potegowe();

  int T;
  cin >> T;
  while (T--) {
    long long N;
    cin >> N;
    cout << wyszukiwanie_binarne(N) << "\n";
  }

  return 0;
}
```

lic.py

```python
import bisect
import math

LIMIT = int(10**18 + 6 * 10**10)

def czy_jest_kwadratem(X):
    pierwiastek = math.isqrt(X)
    return pierwiastek * pierwiastek == X

# Liczby potęgowe podzielimy na:
# małe liczby potęgowe (z wykładnikiem równym 2),
# duże liczby potęgowe (z wykładnikiem większym niż 2).

# Generujemy duże liczby potegowe (z wykładnikiem większym niż 2).
def wygeneruj_duze_potegowe():
    potegowe = []
    baza = 2
    while True:
        aktualna_potega = baza ** 3
        if aktualna_potega > LIMIT:
            break
```


Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2022/2023, 2023/2024, 2024/2025".

```
        while aktualna_potega <= LIMIT:
            # Chcemy mieć jedynie duże liczby potęgowe, więc
            # sprawdzamy czy przez przypadek nie jest kwadratem
            # (tj.  małą liczbą potęgową).
            if not czy_jest_kwadratem(aktualna_potega):
                potegowe.append(aktualna_potega)
            aktualna_potega *= baza
        baza += 1
    return sorted(list(set(potegowe)))

# Liczymy ile jest małych liczb potęgowych mniejszych od X.
def ile_mniejszych_malych(X):
    return math.isqrt(X-1) - 1

# Zwraca którą liczbą potęgową jest X.
def ktora_potegowa(X, duze_potegowe):
    mniejszych_malych = ile_mniejszych_malych(X)
    mniejszych_duzych = bisect.bisect_left(duze_potegowe, X)
    return X - mniejszych_malych - mniejszych_duzych

# Używamy wyszukiwania binarnego do znalezienia odpowiedzi.
def wyszukiwanie_binarne(N, duze_potegowe):
    a, b = 1, LIMIT
    while b-a > 1:
        mid = (a+b) // 2
        # Pytamy, którą liczbą niepotęgową jest mid.
        if ktora_potegowa(mid, duze_potegowe) <= N:
            a = mid
        else:
            b = mid
    return a

def main():
    duze_potegowe = wygeneruj_duze_potegowe()
    T = int(input())
    for _ in range(T):
        N = int(input())
        print(wyszukiwanie_binarne(N, duze_potegowe))

if __name__ == "__main__":
    main()
```


Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2022/2023, 2023/2024, 2024/2025".