Omówienie zadania *Symbol Newtona*

Bartosz Chomiński

1 Zadanie

Symbol Newtona $\binom{N}{K}$ (czytaj: N po K) oznacza liczbę kombinacji K-elementowych zbioru N-elementowego. Można go obliczyć na przykład ze wzoru:

$$\binom{N}{K} = \frac{N!}{K! \cdot (N-K)!},$$

gdzie symbol wykrzyknika oznacza silnię liczby czyli iloczyn kolejnych dodatnich liczb naturalnych do tej liczby. Na przykład: $5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$.

W tym zadaniu sprawdzimy po prostu czy potrafisz liczyć symbol Newtona. Dla bardzo dużych wartości N oraz K. Powodzenia. Dla ułatwienia, wystarczy że podasz resztę z dzielenia wyniku przez 10^9+7.

Napisz program, który wczyta wartości N oraz K, wyznaczy wartość $\binom{N}{K} \bmod (10^9+7)$ i wypisze wynik na standardowe wyjście.

2 Rozwiązanie za wolne

W wyrażeniu podanym w treści zadania są dwa „problemy”: dzielenie oraz silnia, które możemy, a nawet powinniśmy, rozważyć osobno.

2.1 Dzielenie

Dzielenie reszt z dzielenia jest zwodnicze, ponieważ dzielenie jest jedynym spośród czterech podstawowych działań arytmetycznych, które nie jest *rozdzielne względem modulo*, a więc nie ma własności

$$(a \bowtie b) \bmod p = ((a \bmod p) \bowtie (b \bmod p)) \bmod p,$$

gdzie za $\bowtie$ możemy wpisać $+$, $-$ lub $\times$.

Rozwiązaniem tego problemu jest skorzystanie z definicji dzielenia:

Dzielenie to mnożenie przez odwrotność.

Definicję i efektywne sposoby wyznaczenia odwrotności modularnej Dociekliwy Czytelnik może znaleźć w notatce *Odwrotność modularna*.

2.2 Silnia

Resztę z dzielenia silni przez dowolną liczbę można obliczyć prostą pętlą typu `for`, domnażając kolejne liczby naturalne i modulując odpowiednio często.

2.3 Podsumowanie

Uzyskaliśmy rozwiązanie wykonujące co najmniej tyle „kroków” ile wynosi liczba N. Kroki te pochodzą z fazy domnażania kolejnych liczb do wyniku podczas liczenia silni. To jednak zbyt wolno jak na limit czasu wynoszący $0.5\,\mathrm{s}$ i potrzebujemy usprawnienia. Etap dzielenia zajmuje czas stały względem N i K, bo jest to pojedyncze wyznaczenie odwrotności modulo 10^9+7 i kilka mnożeń, zatem powinniśmy się skupić na przyspieszeniu fazy wyznaczania silni.

3 Rozwiązanie wzorcowe

3.1 Potęga informacji

Zauważmy, że zasadniczym problemem przy liczeniu $N!$ jest odległość do ostatniej wartości silni, którą znamy. W rozwiązaniu podanym wyżej, jeżeli mielibyśmy policzyć $1\,234!$, to musielibyśmy wykonać $1\,234$ mnożenia, by osiągnąć wynik.

Wyobraźmy sobie, że wiemy ile wynosi reszta z dzielenia $1\,000!$ przez 10^9+7. Wtedy, by obliczyć $1\,234!$ nie musimy już zaczynać od $1, 2, \ldots$, a możemy wystartować od $1\,000!$ i domnażać $1\,001, 1\,002, 1\,003, \ldots$, wykonując w ten sposób tylko 234 mnożenia zamiast $1\,234$ mnożeń.

By istotnie przyspieszyć proces szukania silni, warto znać odpowiednie reszty z dzielenia wielu silni, przykładowo $1\,000\,000!, 2\,000\,000!, 3\,000\,000!, \ldots$ w równych odstępach aż do $1\,000\,000\,000!$ (dalej będą to dla nas „kamienie milowe”). W ten sposób najdłuższy ciąg domnażania skróci się z miliarda do miliona.

Przejdźmy teraz do wdrożenia tego planu w życie.

3.2 Wskazówki implementacyjne

By zaoszczędzić czas wykonania programu, powinniśmy policzyć kamienie milowe poza czasem mierzonym przez sprawdzaczkę, na przykład jako osobny program na stacji zawodniczej. To obliczenie powinno potrwać kilka do kilkunastu sekund, a jego wynik warto sformatować tak, by wynik tegoż osobnego programu był kodem wpisania wartości do tablicy w języku C++.

Wynik programu liczącego kamienie milowe może zaczynać się tak:

```
t[0] = 1;
t[1] = 641102369;
t[2] = 578095319;
```

Limit wielkości kodu źródłowego w popularnych sprawdzaczkach (SIO, Solve) wynosi 100 kB, a więc możemy wysłać aż 100 000 znaków jako nasze rozwiązanie. Wiedząc to, można się przekonać, że nasz plan zadziała – cały kod potrzebny na wpisanie wartości kamieni milowych do tablicy powinien zmieścić się w 20 kB, ponieważ mamy 1 000 wierszy, każdy po co najwyżej 20 znaków, czyli 20 bajtów.