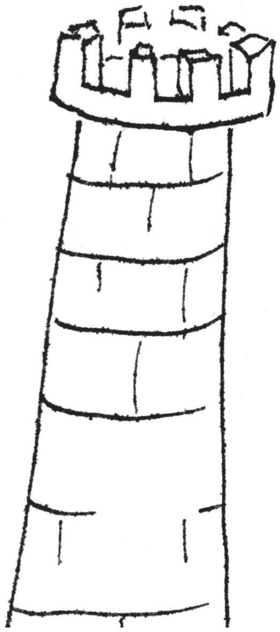
Informatyczny kącik olimpijski (37): Ogromna wieża

Tym razem przyjrzymy się zadaniu *Ogromna wieża* (ang. *A huge tower*) pochodzącemu z Olimpiady Informatycznej Europy Środkowej 2010 (CEOI 2010).

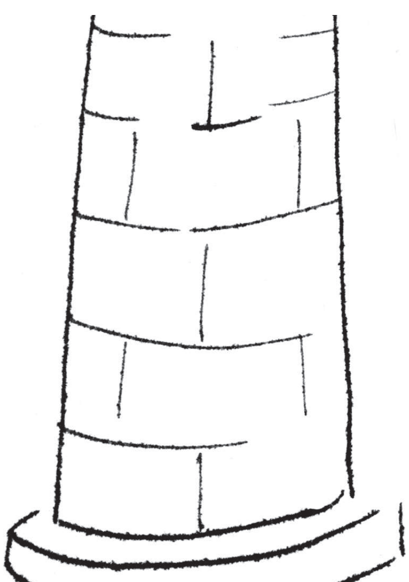
Mamy do dyspozycji n kamiennych bloków różnych rozmiarów. Mamy obliczyć, na ile sposobów można zbudować z tych bloków wieżę, jeżeli należy użyć ich wszystkich, a jeden blok można położyć na drugim tylko wtedy, gdy rozmiar górnego jest nie większy niż rozmiar dolnego powiększony o ustaloną liczbę dodatnią d. Wynik należy zwrócić modulo jakaś nieduża liczba całkowita.

Oznaczmy przez $r_1, r_2, \ldots, r_n$ rozmiary poszczególnych bloków, a wieżę zbudowaną z nich zgodnie z warunkami zadania nazwijmy *poprawną*. Pierwsze rozwiązanie, które przedstawimy, jest najmniej efektywne, ale nie wymaga żadnych spostrzeżeń. Wystarczy użyć programowania dynamicznego do stopniowego obliczania wartości $t(S, a)$, które oznaczają odpowiedź na pytanie „na ile sposobów można zbudować poprawną wieżę z bloków należących do zbioru $S \subseteq \{1, 2, \ldots, n\}$, tak aby blok o numerze a znalazł się na dole?". Naturalnie, zakładamy, że $a \in S$. Jeśli dostępny jest tylko jeden blok, wieżę można zbudować na dokładnie jeden sposób. Jeśli zaś mamy zbiór S co najmniej dwóch bloków, a blok a ma znaleźć się na dole, to należy na wszystkie sposoby wybrać blok, który ma się znaleźć bezpośrednio na nim. Jeśli ten przedostatni blok oznaczymy przez b, to pod warunkiem, że $r_b \leqslant r_a + d$, znaleźliśmy $t(S \setminus \{a\}, b)$ sposobów na zbudowanie naszej wieży. Sumujemy takie wartości po wszystkich możliwościach wyboru $b \in S$ i otrzymujemy $t(S, a)$. W ten sposób, szukając odpowiedzi w kolejności od najmniej licznych zbiorów S, każdą z nich znajdujemy w czasie proporcjonalnym do rozmiaru S, co oznacza, że cały problem zostanie rozwiązany po wykonaniu $O(2^n n^2)$ operacji.

Aby znaleźć bardziej efektywne rozwiązanie, warto inaczej pomyśleć o budowaniu wieży. Zamiast zastanawiać się, który z bloków położymy na spód albo wierzch, rozpatrujmy bloki kolejno, od najmniejszego do największego, ale za to dopuśćmy umieszczanie kolejnego bloku gdzieś w środku już zbudowanej konstrukcji. Załóżmy, że rozmiary bloków r_i są uporządkowane niemalejąco, i rozważmy dowolną poprawną wieżę zbudowaną z bloków o numerach od 1 do $k-1$. Powiedzmy, że w tej wieży blok a leży pod blokiem b i pomiędzy nie chcemy teraz włożyć blok k. Zastanówmy się, kiedy nowa konstrukcja jest poprawna. Poza fragmentem zawierającym bloki a, b i k, nowa wieża jest poprawna wtedy i tylko wtedy, gdy dotychczasowa była. Ponadto, musi być $r_k \leqslant r_a + d$ oraz $r_b \leqslant r_k + d$. Pamiętajmy jednak, że $r_b \leqslant r_k$, więc druga nierówność zawsze zachodzi. To znaczy, że jeżeli mamy poprawną wieżę zbudowaną z bloków $1, \ldots, k-1$ oraz $r_k \leqslant r_a + d$, to wstawiając blok k między a oraz b, otrzymamy poprawną wieżę dla bloków $1, \ldots, k$. Z drugiej strony, jeśli mamy poprawną wieżę dla bloków $1, \ldots, k$ i blok k znajduje się w niej między blokami a oraz b, to $r_k \leqslant r_a + d$. Ponieważ jednak $r_b \leqslant r_k$, więc mamy $r_b \leqslant r_a + d$ i rozważana wieża po usunięciu bloku k staje się poprawną wieżą dla bloków $1, \ldots, k-1$.



Czytelnik Dociekliwy zauważy zapewne, że blok k możemy chcieć położyć na spód lub wierzch wieży, a nie tylko pomiędzy dwa bloki. Czy zmienia to przedstawione rozumowanie? Jak poradzić sobie z takim przypadkiem?



Z powyższego rozumowania wyciągamy wniosek, że wieża ze wstawionym pomiędzy bloki a i b blokiem k (nie mniejszym od wszystkich bloków wieży) jest poprawna wtedy i tylko wtedy, gdy była poprawna bez niego oraz gdy można go położyć na bloku a. Bardzo ważne jest spostrzeżenie, że w tym właśnie wniosku nie używamy tak naprawdę rozmiarów bloków innych niż a oraz k.

Z takim spostrzeżeniem możemy śmiało konstruować algorytm rozwiązujący nasz problem. Oznaczmy przez a_k liczbę tych bloków spośród $1, \ldots, k-1$, na które można położyć blok k. Każdy blok można też położyć na samym dole wieży. Widzimy, że używając tylko pierwszego bloku, poprawną wieżę można zbudować na $a_1 + 1 = 1$ sposób. Używając pierwszych dwóch, na $(a_1+1)(a_2+1)$ sposobów. I tak dalej: aby utworzyć poprawną wieżę z k pierwszych bloków, należy jakoś zbudować wieżę z pierwszych $k-1$ bloków, a następnie na jeden z $a_k + 1$ sposobów ustawić blok k. Ostatecznym wynikiem jest więc $(a_1+1)(a_2+1) \cdot \ldots \cdot (a_n+1)$. Zauważmy jeszcze, że wartości a_i możemy szybko obliczyć, korzystając z (niemalejącego) ciągu r_i. Używamy do tego dwóch indeksów, x oraz y, przy czym x przebiega kolejno wartości od 1 do n, a y ma być zawsze najmniejszym takim indeksem, że $r_y + d \geqslant r_x$. W takim przypadku mamy, dla każdego kolejno rozpatrywanego x, równość $a_x = x - y$. Dodajmy, że oba wskaźniki w trakcie działania tego algorytmu jedynie wzrastają, co zapewnia, że czas obliczenia ciągu a_i jest liniowy. Całkowity koszt czasowy jest więc uzależniony od sortowania rozmiarów bloków i wynosi $O(n \log n)$.

Tomasz KULCZYŃSKI