1A – Od deski do deski

autor zadania: Mateusz Radecki

Treść

Mówimy, że ciąg jest dobry, jeśli możemy sprawić, że cały zniknie, wykonując dowolną liczbę następujących ruchów: wybieramy dwa równe elementy ciągu znajdujące się na różnych pozycjach i usuwamy je wraz ze wszystkimi elementami pomiędzy nimi. Następnie sklejamy ze sobą pozostały prefiks i sufiks ciągu.

Mamy dane liczby n oraz m. Należy policzyć (modulo duża liczba pierwsza) liczbę dobrych ciągów n-elementowych o wartościach z przedziału $[1, m]$.

Rozwiązanie

Zaznaczmy w oryginalnym ciągu pary pozycji, które odpowiadają parom wybieranych elementów. Spójrzmy na dwie pary (ℓ_1, r_1) oraz (ℓ_2, r_2). Oczywiście wszystkie wartości ℓ_1, r_1, ℓ_2 i r_2 muszą być parami różne. Załóżmy bez straty ogólności, że $\ell_1 < \ell_2$. Nie może zachodzić $\ell_2 < r_1$ i $r_1 < r_2$, gdyż wtedy usuwając najpierw pierwszy przedział, usunęlibyśmy ℓ_2, a usuwając najpierw drugi przedział, usunęlibyśmy r_1. Jeśli zachodzi $r_2 < r_1$, to przedział $[\ell_2, r_2]$ musieliśmy usunąć przed $[\ell_1, r_1]$. Łatwo zauważyć, że niepotrzebnie w ogóle usuwaliśmy przedział $[\ell_2, r_2]$, gdyż i tak cały zniknąłby przy usuwaniu przedziału $[\ell_1, r_1]$.

Możemy zatem wywnioskować, że ciąg jest dobry wtedy i tylko wtedy, gdy możemy podzielić go na przedziały długości co najmniej 2 takie, że każdy przedział zaczyna oraz kończy się tą samą wartością. Policzymy dobre ciągi za pomocą programowania dynamicznego. Niech $DP[i][j][l]$ oznacza liczbę takich ciągów i-elementowych, dla których istnieje dokładnie j wartości z przedziału $[1, m]$, które po dołożeniu na koniec ciągu zamieniały by go w dobry ciąg. Dodatkowo, jeśli $l = 1$, to zliczamy jedynie dobre ciągi, a jeśli $l = 0$, to zliczamy jedynie ciągi, które nie są dobre.

Programowanie dynamiczne inicjalizujemy jako $DP[0][0][1] = 1$. Przejdźmy się po i od 0 do $n - 1$ włącznie i wypychajmy wartości w przód.

Jeśli $l = 0$ i dołożymy wartość, która nie zamienia ciągu w dobry ciąg, to j nie ulega zmianie.

Wykonujemy zatem: $DP[i + 1][j][0] \mathrel{+}= dp[i][j][0] \cdot (m - j)$.

Jeśli zaś dołożymy wartość, która uczyni ciąg dobrym, to wartość j również się nie zmieni, ale ciąg będzie dobry.

Wykonujemy zatem: $DP[i+1][j][1] \mathrel{+}= DP[i][j][0] \cdot j$.

Jeśli $l = 1$ i dołożymy wartość, która zamienia ciąg w dobry ciąg, to wartość j również się nie zmienia.

Wykonujemy zatem: $DP[i+1][j][1] \mathrel{+}= DP[i][j][1] \cdot j$.

Jeśli zaś dołożymy wartość, która nie zmienia ciągu w dobry ciąg, to zauważmy, że po każdym następnym wystąpieniu tej wartości będziemy w stanie usunąć cały ciąg (gdyż będziemy mogli usunąć prefiks długości i oraz w jednym ruchu wszystko to co jest po nim).

Wykonujemy zatem: $DP[i+1][j+1][0] \mathrel{+}= DP[i][j][1] \cdot (m-j)$.

Warto zauważyć, że wartość parametru j nigdy nie przekroczy wartości n. Całe programowanie dynamiczne ma zatem złożoność $\mathcal{O}(n^2)$.