P-I-3 Střelec **Vladimír Vávra**

Princip řešení: Na první pohled by úloha mohla svádět k prohledávání do hloubky. Uvážíme-li ovšem, kolik možných zanoření by proběhlo, rychle dojdeme k závěru, že i pro relativně malá vstupní data by program nemohl pokračovat, jelikož by program měl exponenciální složitost.

V takových případech často úloha vede na dynamické programování. Při něm je první krok k nalezení řešení zjistit si, jaký by byl výsledek pro prvních několik nejjednodušších pod-úloh.

Vezměme si tu nejjednodušší. Máme šachovnici písmen RxS, avšak slovo, které máme vybrat je jednopísmenné. Takový výsledek by prostý součet všech výskytů tohoto písmena. Jestliže bychom přidali druhé písmeno, museli bychom při přeskočení na toto písmeno nejprve navštívit předchozí písmeno. Pozice těchto dvou písmen tedy musí být v jedné diagonále. Stejně jako jsme v předchozí pod-úloze nalezli všechna možná první písmena, nyní nalezneme všechna druhá písmena. Nyní sečteme, kolika možnými způsoby je možné se na toto písmeno dostat. Jedná se o součet, kolika možnostmi je možné se dostat na předchozí písmena dostupné po diagonálách k dané pozici. To je nyní vždy jedna, jelikož se jedná o počáteční písmena slova. Toto uděláme pro každý výskyt druhého písmena. Součet těchto výsledků je celkový počet možností, jak poskládat slovo.

Pro další písmeno bychom udělali to samé. Vyhledáme všechny výskyty a spočítáme, kolika způsoby je možné dostat se toto místo z předchozích písmen. Sečteme tedy počty možností, jak se dostat na všechna předchozí písmena dostupná z této pozice. Toto můžeme opakovat pro N písmen.

Popis algoritmu: Datové struktury:

- Pole A: RxS, kde jsou uložena písmena
- pole B: RxS, kde je uložen počet možností, jak se dostat na toto písmeno (na začátku všude 0). Datový typ bude celočíselný o dostatečném rozsahu (BigInteger)

Algoritmus:

1. Najdi všechny výskyty prvního písmena v poli A a ulož do pole B na stejný index 1
2. Opakuj pro všechna zbylá písmena ve slově
   (a) Procházej celé pole A dokud nenajdeš aktuálně hledané písmeno na indexu (i, j)
   (b) Projdi všechny možné pozice na diagonálách. V případě, že se na nějaké z nich nachází předchozí písmeno (x, y), udělej operaci B[i, j] += B[x, y];
   (c) Zpět na krok (a), dokud nebude projité celé pole A
3. Výsledkem je součet čísel v poli B na indexech, kde se v poli A nachází poslední písmena hledaného slova. Ten zmodulujeme číslem $10^9 + 7$ a získáme tak náš výsledek

Paměťová složitost: Potřebujeme dvě pole o velikosti šachovnice, takže: O(2*R*S)

Časová složitost: Celkem musíme uskutečnit celkem N kroků v bloku 2 (bereme jako součást část 1). V každém z těchto N kroků musíme projít celé pole RxS a pokaždé, kdy narazíme na nějaké písmeno, které je aktuální, musíme projít diagonály. Projetí diagonál trvá nejhůře 2*min(R,S). V nejhorším případě budou všechna písmena v nějaký čas aktuální, takže celkově procházení diagonál zabere R*S*2*min(R, S).

Nejhorší časová složitost tedy bude v součtu:
$O(N * R * S + R * S * 2 * min(R, S)) = O(R * S * (N + 2 * min(R, S)))$