Примерни решения на домашно № 2

Задача 1: Даден е ориентиран граф $G = (V, E)$. Някои от неговите ребра са *лоши*, а останалите ребра са *добри*. Дадени са и два върха s и t от G. Предложете колкото е възможно по-ефикасен алгоритъм, който или намира път от s до t с не повече от четири лоши ребра, ако такъв съществува, или връща индикация, че такъв път не съществува, в противен случай. Допуснете, че във време $O(1)$ може да се получи информация дали дадено ребро е добро или лошо; примерно, има функция IsGood, такава че за всяко ребро e, IsGood(e) връща във време $O(1)$ или стойност ДА, ако реброто e е добро, или стойност НЕ, ако реброто e е лошо.

Решение: Едно решение е да направим графа тегловен, като тегловната функция дава тегло 0 на добрите ребра и тегло 1 на лошите ребра, след което пускаме алгоритъма на Dijkstra за най-къс път от s до t (алгоритъмът престава да итерира след итерацията, в която намира най-къс път до t). Ако намереният най-къс път от s до t е с тегло ≤ 4, връщаме него, в противен случай най-къс път с не повече от 4 лоши ребра няма и връщаме индикация, че такъв път не съществува.

Ако не искаме да ползваме нулеви тегла, може да даваме тегло 1 на добро ребро и тегло $m + 1$ на лошо ребро. При това положение, намереният най-къс ползва не повече от 4 лоши ребра тстк неговото тегло е $< 5m + 5$.

Задача 2: Дефинираме, че *диаметърът* на масив от положителни числа $A[1\,..\,n]$ е $\mathrm{diam}(A) = \max\{|A[i] - A[j]| \mid 1 \le i \le j \le n\}$. *Разбиване на* A *на* k *подмасива*, където $k \le n$, е редица от масиви $\Pi = (B_1, B_2, \ldots, B_{k-1}, B_k)$, такива че

$$
\begin{aligned}
B_1 &= A[i_1\,..\,i_2 - 1] \\
B_2 &= A[i_2\,..\,i_3 - 1] \\
&\ldots \\
B_{k-1} &= A[i_{k-1}\,..\,i_k - 1] \\
B_k &= A[i_k\,..\,n]
\end{aligned}
$$

където $i_1 = 1$ и $i_1 < i_2 < \cdots < i_{k-1} < i_k \le n$. Диаметърът на разбиването Π е $\mathrm{diam}(\Pi) = \max\{\mathrm{diam}(B_i) \mid 1 \le i \le k\}$.

Предложете ефикасен алгоритъм, който при даден сортиран масив $A[1\,..\,n]$ от положителни числа и число $k \le n$ намира разбиване на A на k подмасива с минимален диаметър. Достатъчно е алгоритъмът да намира оптималната цена, а не самото решение.

Решение: Нека $D[i, \ell]$ е минималният диаметър на разбиване на $A[1\,..\,i]$ на точно ℓ подмасива, за $0 \le i \le n$ и $0 \le \ell \le k$. Търсеният отговор е $D[n, k]$. Ще конструираме решение по схемата **Динамично Програмиране**. Ключов факт за обосновка на рекурсивната декомпозиция е, че диаметърът на сортиран масив $A[p\,..\,q]$ е разликата $A[q] - A[p]$.

Базата на рекурсивната декомпозиция е $i = 0$ или $\ell = 0$. Очевидно

$$
\begin{aligned}
&D[0, 0] = 0 \\
&D[i, 0] = \infty \text{ за } i > 0 \\
&D[0, \ell] = \infty \text{ за } \ell > 0
\end{aligned}
$$

За $i, \ell > 0$, в сила е

$$D[i, \ell] = \min\{\max\{D[j, \ell - 1], A[i] - A[j + 1]\} \mid 0 \le j \le i - 1\}$$

Аргументацията за коректност е следната. Мислим за диаметъра на разбиване на $A[1\,..\,i]$ на ℓ подмасива. Последният масив в такова разбиване е точно един от тези:

$A[1\,..\,i]$ с диаметър $A[i] - A[1]$
$A[2\,..\,i]$ с диаметър $A[i] - A[2]$
$A[3\,..\,i]$ с диаметър $A[i] - A[3]$
$\ldots$
$A[i\,..\,i]$ с диаметър $A[i] - A[i]$

Във всеки от тези случаи, j е индексът на елемента, който в A е вляво от най-левия елемент на съответния последен масив. Тогава последният масив е $A[j{+}1\,..\,i]$, където $j \in \{0, \ldots, i - 1\}$. За всяко такова j, диаметърът на цялото разбиване (на $A[1, \ldots, i]$ на ℓ подмасива) е по-голямото от диаметъра на $A[j + 1\,..\,i]$, който е $A[i] - A[j + 1]$, и диаметъра на оптимално разбиване на $A[1, \ldots, j]$ на точно $\ell - 1$ подмасива. Вземаме минимума по j, понеже задачата е минимизационна.