# Домашно №2 по ДАА на Компютърни Науки

Камен Димитров Младенов, Група 6

9 Април 2023

## Задача 1. Решение

### а) Ляв фрагмент

Цикъла ще извърши точно $m.n = n.n.n = n^3$ итерации. Тоест за $c > 0, f(n) = n^3 + c \asymp n^3$ по дефиниция.

### б) Централен фрагмент

Алгоритъма ще приключи директно ако $n < 10$. Иначе се изпълнява останалата част на алгоритъма.
Условието на цикъла на ред 4 никога няма да е вярно, понеже $6 \not> 8$.

Тялото на цикъла с условие на ред 7 ще се изпълни точно един път: при първо достигане на условието $n - j = n - 6 \geq 10 - 6 = 4 > 1$, на ред 8 $j$ става $n$ и на второто достигане на условието $n - j = n - n = 0 \not> 1$. От това единствено изпълнение имаме рекурентно извикване:

$$g(n-1) + g(n-2) \tag{1}$$

Тялото на цикъла с условие на ред 10 ще се изпълни точно един път: към момента $j = n$ и при първо достигане на условието, $n \geq n$. От ред 11 $j$ става 2 и при второ достигане на условието, $2 = j \not\geq n$ понеже $n \geq 10$. От това единствено изпълнение имаме рекурентно извикване:

$$g(n-j) = g(n-2) \tag{2}$$

От (1) и (2): $g(n) = g(n-1) + 2g(n-2) + c$

Характеристично уравнение: $x^n = x^{n-1} + 2x^{n-2} \quad x^2 - x - 2 = 0 \quad$ с корени $\{-1, 2\}_M$

Нехомогенната част е константа и няма корени. $\quad g(n) = c_1(-1)^n + c_2 2^n \implies g(n) \asymp 2^n$

### в) Десен фрагмент

Алгоритъма ще приключи директно ако $n < 2$. Иначе разглеждаме останалата част на алгоритъма.
От редове 4 и 7 имаме две еднакви рекурентни извиквания:

$$h\left(\frac{n}{3}\right) \tag{3}$$

Ще покажем, че тялото на цикъла с условие на ред 5 ще се изпълни точно

$$\lfloor \lg n \rfloor \text{ на брой пъти.} \tag{4}$$

Ако допуснем, че $n = 2^k$ за някое $k \in \mathbb{N}$, то тогава $i$ последователно ще придобие стойностите $2^1, 2^2, ..., 2^k$, тоест цикъла ще се изпълни $k = \log_2 n \in \mathbb{N}$ на брой пъти.
Ако $n \neq 2^k$ за което и да е $k \in \mathbb{N}$, то тогава съществува $k' \in \mathbb{N}$, такова че $2^{k'} < n < 2^{k'+1}$ и тогава $i$ придобива стойностите $2^1, 2^2, ..., 2^{k'}$. Понеже $k' < \lg n < k' + 1$, то цикъла ще се изпълни $k' = \lfloor \lg n \rfloor \in \mathbb{N}$ на брой пъти.

От (3) и (4): $h(n) = 2\,h\left(\frac{n}{3}\right) + \lfloor \lg n \rfloor$

Тогава за $a = 2, b = 3, f(n) = c \quad$ За $\varepsilon = \frac{1}{2}, f(n) \preceq \frac{n^{\log_3 2}}{\sqrt{n}}, \log_3 2 - \frac{1}{2} > 0 \xrightarrow[\text{Мастър Теоремата}]{\text{1 случай на}} h(n) \asymp n^{\log_3 2}$

## Задача 2. Решение

### а) Инверсен вектор

Инверсния вектор е $(4, 1, 1, 0, 2, 3, 0, 0, 0)$

### б) Алгоритъм

```
1  vector<int> rebuild(vector<int> invvec) {
2      vector<int> result;
3      for (int i = 0; i < result.size(); i++)
4          result[i] = i + 1;
5      for (int i = invvec.size() - 1; i >= 0; i--) {
6          for (int j = i; invvec[i] > 0; j++) {
7              swap(result[j], result[j+1]);
8              invvec[i]--;
9          }
10     }
11     return result;
12 }
```

От дефинициите, инверсният вектор съдържа броя инверсии за всяко $j$ от 1 до $n$. Алгоритъмът `rebuild` използва сортиран вектор (масив) в който създава инверсиите от исканата пермутация. Инверсния вектор се итерира в обратен ред

Нека $f(n)$ да бъде функция по големина на входа, която определя сложността по време на `rebuild`. Циклите с условия на редове 3 и 5 ще се изпълнят $n$ на брой пъти (поотделно). За всяка итерация на цикъла с условие на ред 5, вътрешния цикъл с условие на ред 6 ще се изпълни, в най-лошия случай, точно $n - i - 1$ на брой пъти, колкото е максималния брой елементи, по-големи от $i$-тия (от числото $i + 1$). Тоест:

$$f(n) = n + \sum_{i=0}^{n-1}(n - 1 - i) + c = n + \sum_{i=0}^{n-1}(n - 1) - \sum_{i=0}^{n-1} i + c = n + (n - 1)(n - 1) - \frac{(n-1)n}{2} + c$$
$$= n + n^2 - 2n + 1 - \frac{1}{2}n^2 + \frac{1}{2}n + c = \frac{1}{2}n^2 - \frac{1}{2}n + 1 + c$$

$f(n) \asymp n^2$

## Задача 3. Решение

### а) Алгоритъм

Понеже масивът е двойно индексиран, ще го наричам матрица. Нека преиндексираме входната матрица на алгоритъма: $M[0..(n-1), 0..(m-1)]$, и нека $M.n$ връща $n$ и $M.m$ връща $m$.

```
1  bool isinteresting(matrix<int, int> M) {
2      for (int i = 1; i < M.n; i++) {
3          for (int j = 1; j < M.m; j++) {
4              if (M[i-1, j-1] + M[i, j] > M[i-1,j] + M[i,j-1])
5                  return false;
6          }
7      }
8      return true;
9  }
```

Доказателство за коректност:

· **Лема за вътрешния цикъл**

По отношение на едно изпълнение на външния цикъл, ефектът на вътрешния цикъл е, че $M[i, 0..(m-1)]$ са проверени и не нарушават "интересността" на матрицата.

**Доказателство:** По отношение на едно изпълнение на външния цикъл:

**Инвариант на вътрешния цикъл:** За всяко изпълнение на ред 3, $M[i, 0..(j-1)]$ са проверени и не нарушават интересността.

1. **База:** При първото изпълнение на ред 3, $j = 1$ и $M[i, 0]$ запазва интересността, понеже не съществува валидно $q < 0$ и неравенството се свежда до $M[0, 0] \leq M[0, 0]$.
2. **Поддръжка:** Да допуснем, че твърдението е вярно за някое нефинално достигане на ред 3.

· **Инвариант на външния цикъл**

За всяко достигане на ред 3, подматрицата $M'[0.. \min\{n-1, d\}, 0.. \min\{m-1, d\}]$ е интересна.

Доказателство за сложност по време:

· **Формула**

· **Доказателство**

## б) Доказателство

## в) Обяснение

## г) Алгоритъм