Zoltán Pálinkás
Zadaća 1

(1.)

a) $\log_2 n = \Theta(\ln n)$

postoje $c_1 = 1$ i $c_2 = 5$ te $n_0 = 1$ t.d. vrijedi
$$0 \leq \ln n \leq \log_2 n \leq 5 \cdot \ln n, \ \forall n \geq 1$$

b) $3n\sqrt{n} + 6n \ln n - 4n = O(n^2)$

postoje $c = 5$ i $n_0 = 100$ t.d. vrijedi
$$0 \leq 3n\sqrt{n} + 6n\ln n - 4n \leq 5 \cdot n^2, \ \forall n \geq 100$$

c) $3n\sqrt{n} + 6n \ln n - 4n = O(n^3)$

postoje $c = 1$ i $n_0 = 3$ t.d. vrijedi
$$0 \leq 3n\sqrt{n} + 6n\ln n - 4n \leq n^3, \ \forall n \geq 3$$

d) $\sum_{i=0}^{\lfloor \log_4 n \rfloor} 4^i = \Theta(n)$

$\lfloor \log_4 n \rfloor = k$ $\quad$ +1 jer počinjemo od 0

$$\sum_{i=0}^{k} 4^i = \frac{4^{k+1} - 1}{4 - 1} = \frac{4^{k+1} - 1}{3} = \frac{4^{\lfloor \log_4 n \rfloor + 1} - 1}{3}$$

geometrijski niz

$$\frac{4^{\lfloor \log_4 n \rfloor + 1} - 1}{3} \leq c\, n$$

$\cancel{\frac{1}{3} \cdot 4^{\lfloor \log_4 n \rfloor + 1} - \frac{1}{3} \leq c\, n}$

$\cancel{\frac{1}{3} \cdot 4^{\lfloor \log_4 n \rfloor} \cdot 4 - \frac{1}{3} \leq c\, n}$

$\cancel{\log_4 \frac{1}{3} + \log_4 4^{\lfloor \log_4 n \rfloor} + \log_4 4 \leq \log_4 (c\, n + \frac{1}{3})}$

$\cancel{\log_4 \frac{1}{3} + \lfloor \log_4 n \rfloor + 1 \leq \log_4 (c\, n + \frac{1}{3})}$

postoje $c_1 = 1$, $c_2 = n$, $n_0 = 16$ t.d. vrijedi $0 \leq n \leq \sum_{i=0}^{\lfloor \log_4 n \rfloor} 4^i \leq n^2, \ \forall n \geq 16$

(2) $8n^2 + 5n + \Theta(n \log n) = \Theta(n^2)$

Prvo bih pronašao najbrže rastući dio f-je, a to je $8n^2$ i usporedio ga s n^2. Zatim bih pronašao c_1, c_2 i n_0 za koje vrijedi da je

$$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n), \ \forall n \geq n_0$$

Za c_1 mogu uzeti $c_1 = 1$, pošto je $n^2 \leq 8n^2$.

Za c_2 mogu uzeti $c_2 = 100$, a za $n_0 = 1$

tada vrijedi $0 \leq n^2 \leq 8n^2 + 5n + \Theta(n \log n) \leq 100n^2, \ \forall n \geq 1$

$\Theta(n \log n)$ sam mogao ignorirati, jer je n^2 brže rastuća f-ja

(3.) $n! = O(n^n)$

Za dokaz možemo raspisati prvih n članova:

$n! = n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot 3 \cdot 2 \cdot 1$

$n^n = n \cdot n \cdot n \cdot \ldots \cdot n \cdot n \cdot n$

Ako usporedimo gornja dva reda, lako zaključujemo da n^n raste puno brže od $n!$. Ovo vrijedi $\forall \, c \geq 1$ i $n_0 \geq 1$

$$0 \leq n! \leq c\, n^n, \ \forall n \geq n_0$$

(4.)

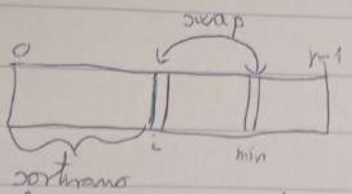
b) Pseudokod:

selection Sort(A, n)   0-indeksirano

for i ← 0 to n-2	c_1	n
{ min ← i	c_2	$n-1$
for j ← i+1 to n-1	c_3	n
if (A[j] < A[min]) min = j	c_4	$\sum_{i=0}^{n-2}(i+1)$
if(i!=min) swap (A[i], A[min])	c_5	$n-1$
}		

$$T(n) = c_1 n + c_2(n-1) + c_3 n + c_4 \cdot \sum_{i=0}^{n-2}(i+1) + c_5(n-1)$$
$$= (c_1 + c_3)n + (c_2 + c_5)(n-1) + \frac{n(n-1)}{2} \cdot c_4$$
$$= \frac{1}{2} n^2 c_4 + (c_1 + c_3 - \frac{1}{2}c_4)n + (c_2 + c_5)(n-1)$$
$$= O(n^2)$$ - u najgorem slučaju

c)



Za dokazivanje algoritma ću se koristiti matematičkom indukcijom

B: $n = 2$ Za bazu ću uzeti listu duljine 2, algoritam će pronaći manji od ta dva elementa te ga postaviti na prvo mjesto

P: $n = k$ Kao pretpostavku uzimam polje od k elemenata i pretpostavljam da je ono sortirano.

K $n = k+1$ Pošto je prema pretpostavci polje s k članova sortirano, preostaje nam samo provjeriti zadnja dva elementa. Pogledajmo pseudokod algoritma:

```
selectionSort (A,n)            0-indeksirano
 for i←k to n-2
{
   min ← i
   for j ← i+1 to n-1
        if (A[j] < A[min]) min = j
 if(i != min) swap (A[i], A[min])
}
```

Algoritam nam može krenuti od k-tog elementa, pošto su prijašnji članovi već sortirani prema pretpostavci, pa nam preostaje usporediti samo zadnja dva člana. Algoritam će pronaći manjeg te ga postaviti na k-to mjesto, a većeg na k+1, ako je potrebno. Tako dolazimo do sortiranog polja

d) Sortiranje 100 nizova od 1000 elemenata je trajalo oko 1,5 ms po nizu, a kada sam povećao broj elemenata na 1000 000 nisam dobio rezultate