# OBLIKOVANJE I ANALIZA ALGORITAMA — 1. kolokvij

## 8. 5. 2008.

1. (10) Između ponuđenih odgovora

$$\Theta(1), \quad \Theta(\lg\lg n), \quad \Theta(\lg n), \quad \Theta(n), \quad \Theta(n\lg n), \quad \Theta(n^2), \quad \Theta(2^n), \quad \Theta(n!),$$

nađite točan red veličine za broj koliko puta se izvršava naredba `x = x + 1` u svakom od sljedećih dijelova programa

(a)
```
i = n;
while (i >= 1) {
  for j = 1 to n
    x = x + 1;
  i = i / 2;
}
```

(b)
```
i = 2;
while (i < n) {
  i = i * i;
  x = x + 1;
}
```

2. (10) Zadana je rekurzivna relacija

$$T(n) = 4\,T(n/3) + f(n), \quad f(n) = n,$$

uz početni uvjet $T(1) = d > 0$. Nađite uvjetno asimptotsko ponašanje relacijom $\Theta$ za rješenje $T(n)$, ako je $n$ potencija od 3. Da li se dobiveno rješenje može proširiti tako da asimptotsko ponašanje vrijedi bezuvjetno, za svaki dovoljno veliki $n \in \mathbf{N}$, ako u rekurziji piše $\lfloor n/3 \rfloor$, umjesto $n/3$?

3. (20) Profesor Tražić predlaže sljedeći rekurzivni algoritam za binarno traženje zadane vrijednosti *key* u uzlazno sortiranom komadu polja $a[i], \ldots, a[j]$. Algoritam treba vratiti indeks $k$ takav da je $a[k] = key$, ako vrijednost *key* postoji u tom komadu polja. U protivnom, ako ne nađe traženu vrijednost, algoritam treba vratiti $-1$.

```
int bsearch(a, i, j, key) {
  if (i > j) return -1;
  k = (i + j) / 2;
  if (key == a[k]) return k;
  if (key < a[k])
    return bsearch(a, i, k, key);
  else
    return bsearch(a, k + 1, j, key);
}
```

Radi li ova verzija algoritma korektno, tj. nalazi li zadanu vrijednost ako ona postoji u zadanom komadu polja, i vraća li $-1$ ako je nema?

Ako ovaj algoritam radi korektno, kolika je njegova vremenska složenost u najgorem slučaju? Ako ne radi korektno, pokažite to primjerom i objasnite što se tada događa. Kako treba popraviti algoritam i kolika je onda vremenska složenost u najgorem slučaju?

**OKRENITE!**

4. (15) Napišite algoritam koji na ulazu dobiva polje $a$ od $n$ realnih brojeva, sortirano u uzlaznom (nepadajućem) poretku, i realnu vrijednost $key$. Algoritam treba vratiti `TRUE` ako postoje dva različita indeksa $i$, $j$ takva da je $a[i]+a[j] = key$. U protivnom, treba vratiti `FALSE`. Smijete koristiti cjelobrojni prikaz logičkih vrijednosti `TRUE` $= 1$, `FALSE` $= 0$ (kao u C-u).

Red veličine vremenske složenosti algoritma mora biti $O(n \log n)$. Analizirajte složenost vašeg algoritma i pokažite da ona zadovoljava ovaj uvjet.

Napomena: Broj bodova ovisi o složenosti algoritma. Složenost $O(n \log n)$ vrijedi najviše 15 bodova. **Bonus**: složenost $O(n)$ vrijedi 5 bodova više!

**REZULTATI:** četvrtak, 15. 5. 2008. u 10 sati. Saša Singer