JMBAG **IME I PREZIME**

Algoritmi i strukture podataka – međuispit

24. studenoga 2020.

Ispit donosi maksimalno 35 bodova. Ovaj primjerak ispita trebate predati s upisanim imenom i prezimenom te JMBAG-om. Rješenja 2., 4. i 5. zadatka upišite u za to predviđena mjesta na ovom papiru, a rješenja 1. i 3. zadatka napišite na svojim papirima ili unutrašnjosti košuljice.

Zadatak 1. (7 bodova)

Za zadane nizove A i B duljine `m` i `n` napišite rekurzivnu funkciju koja počevši od zadnjeg elementa (tj. elementa s najvećim indeksom) ispisuje vrijednosti elemenata oba niza na nekoj poziciji i dodatno vrijednost većega od ta dva elementa. Ako je jedan niz dulji, tada za elemente kraćeg niza koji nisu definirani ispisati "-". Pretpostaviti da su duljine nizova `m` i `n` ≥ 0. Ako su oba niza prazna (tj. `m` = 0 i `n` = 0), ne treba ništa ispisivati.

Primjeri:

Niz A sadrži elemente 2, 4, 1, 3, a niz B sadrži elemente 4, 4, 8. Ispis treba izgledati ovako:
`3 – 3`
`1 8 8`
`4 4 4`
`2 4 4`
	Niz A sadrži elemente 11, 14, 12, a niz B sadrži elemente -9, -1, 41. Ispis treba izgledati ovako:
	`12 41 41`
	`14 -1 14`
	`11 -9 11`
		Niz A je prazan (tj. `m` = 0), a niz B sadrži elemente 11, 14, 12. Ispis treba izgledati ovako:
		`– 12 12`
		`– 14 14`
		`– 11 11`

Prototip funkcije je:

```
void ispisiVeci(int A[], int m, int B[], int n);
```

Napomena: Nerekurzivno rješenje se neće priznati.

Zadatak 2. (7 bodova) - ISPUNITI NA OVOM PAPIRU

Zadan je razred `Stack<T>` kojim je implementiran stog. Stog ima članske funkcije `push(T data)` i `pop(T &data)` koje omogućavaju stavljanje i skidanje elemenata sa stoga. Potrebno je nadopuniti kôd funkcije `isPalindrome`, koja provjerava je li ulazni znakovni niz palindrom i prema tome vraća istinu ili laž.

```
template <class T> class Stack {
 private:
   struct Atom {
      T data;
      Atom *next;
   };
   Atom *head = nullptr;
 public:
   bool push(T data);
   bool pop(T &data);
};
```

```
bool isPalindrome(string str){
    Stack<char> s;
    char c;
    for (int i=0; i<str.length(); i++) {
        if (i < (str.length()/2)) {

            _____________________________________;
        } else if (i == (str.length()/2) && _____________________________________) {
            // ništa
        } else {
            _____________________________________;
            if (c != str[i])
                _____________________________________________;
        }
    }
    _____________________________________;
}
```

Zadatak 3. (7 bodova)

Zadana je jednostruko povezana lista cijelih brojeva. Lista ima strukturu prikazanu okvirom desno.

```
template <class T> class List {
private:
    struct Atom {
        T data;
        Atom *next;
    };
    Atom *head = nullptr;
public:
    bool insert(T data);//dod. na početak
};
```

Napišite člansku funkciju sljedećeg prototipa:

```
void kopirajParne(List<T> &dstList);
```

koja će sve parne elemente iz liste nad kojom je pozvana prebaciti u listu dstList. Poredak elemenata u dstList **mora biti isti** kao i poredak u originalnoj listi, npr.:

srcList: HEAD -> 5 -> 4 -> 7 -> 1 -> 6 -> 8 -> 2 -> 3 -> NULLPTR
dstList: HEAD -> 4 -> 6 -> 8 -> 2 -> NULLPTR

Zadatak 4. (7 bodova) - ISPUNITI NA OVOM PAPIRU

Za funkcije **f** i **g** odredite, ako je moguće, vrijeme izvođenja u Θ notaciji, a ako nije moguće, odredite vrijeme izvođenja u O i Ω notaciji. Rješenja upišite u pravokutnike ispod zadataka.

```
/* A je polje n cijelih brojeva (n >= 1). Funkcija za
 * sortiranje sortira niz uzlazno i implementira
 * algoritam naveden u imenu funkcije.
 */
void f(int A[], int n) {
   // Funkcija napuniNiz generira elemente niza u
   // vremenu 0(n)

   for (int i = n; i >= 1; i--) {
      napuniNiz(A, n);
      selectionSort(A, n);
      for (j = 0; j < n; j++)
         std::cout << A[j] << " ";
      std::cout << endl;
   }
}
```

Rješenje:

```
int g(int n) { // n >= 0
   int cnt = 0;
   if (n == 0) return 1;
   else {
      cnt += g(n - 1);
      cnt += g(n - 1);
      return cnt;
   }
}
```

Rješenje:

Zadatak 5. (7 bodova) - ISPUNITI NA OVOM PAPIRU

Zadano je polje cijelih brojeva s elementima **8, 2, 6, 4, 1, 3, 5, 7**. Ilustrirajte uzlazno sortiranje polja algoritmom **Poboljšani Bubblesort**. Ispišite polje nakon **<u>svakog koraka vanjske petlje</u>** (indeks vanjske petlje je varijabla i).

Početno polje:	8	2	6	4	1	3	5	7
i = 0								
i = 1								

Rješenja:

1. zadatak (7 bodova)

```
void ispisiVeci(int A[], int m, int B[], int n) {
    if (m >= 1 || n >= 1) {
        if (m > n) { // 1 boda
            cout << A[m - 1] << " " << "-" << " " << A[m - 1] << endl;
            --m;
        }
        else if (n > m) {
            cout << "-" << " " << B[n - 1] << " " << B[n - 1] << endl;
            --n;
        }
        else { // m == n
            int veci = std::max(A[m - 1], B[n - 1]);
            cout << A[m - 1] << " " << B[n - 1] << " " << veci << endl;
            --m; --n;
        }
        ispisiVeci(A, m, B, n);
    }
}
```

2. zadatak (7 bodova)

```
bool isPalindrome(string str){
    Stog<char> s;
    char c;
    for (int i=0; i<str.length(); i++) {
        if (i < (str.length()/2)) {
            s.push(str[i]);
        }
        else if (i == (str.length()/2) && str.length() % 2 == 1) {
            // ništa
        }
        else {
            s.pop(c);
            if (c != str[i])
                return false;
        }
    }
    return true;
}
```

3. zadatak (7 bodova)

```
void kopirajParne(List<T> &dstList) {
    Atom *dstListTail = nullptr;
    for (auto curr = head; curr != nullptr; curr = curr->next) {
        //ako je broj parni
        if (curr->data % 2 == 0) {
            //stvori novi Atom
            Atom* newAtom = new (nothrow) Atom;
            //if (newAtom == nullptr) return;
            newAtom->data = curr->data;
            newAtom->next = nullptr;

            //dva slučaja:
            //1. dstList je prazna
            if (dstList.head == nullptr) {
                dstList.head = dstListTail = newAtom;
            }
            else {
            //2. dstList nije prazna
                dstListTail->next = newAtom;
                dstListTail = dstListTail->next;
            }
        }
    }
}
```

4. zadatak (7 bodova)

a) $T(n) = n \cdot (\Theta(n) + \Theta(n^2) + \Theta(n)) = \mathbf{\Theta(n^3)}$

b) $T(n) = 2 \cdot T(n-1) + \Theta(1) = 2 \cdot (2 \cdot T(n-2) + \Theta(1)) + \Theta(1) = \ldots$
$= (2^{n-1} + \ldots + 2^0) \cdot \Theta(1) = \mathbf{\Theta(2^n)}$

5. zadatak (7 bodova)

Početno polje: 8 2 6 4 1 3 5 7

i = 0 : 2 6 4 1 3 5 7 8

i = 1 : 2 4 1 3 5 6 7 8

i = 2 : 2 1 3 4 5 6 7 8

i = 3 : 1 2 3 4 5 6 7 8

i = 4 : 1 2 3 4 5 6 7 8

Nije bilo promjene za i==4, kraj.