

13. Prepostavka je da postoje funkcije za operacije nad stogom skinii i dodaj sa sljedećim prototipima:

```
int skinii (int *stavka, int stog[], int *vrh)
```

```
int dodaj (int stavka, int stog[], int n, int *vrh)
```

Što će ispisati sljedeći program?

```
#include <stdio.h>

#define MAXSTOG 100

int main() {
    int stog[MAXSTOG],vrh=-1;
    int a=1,b=2,c=3;
    dodaj(a,stog,MAXSTOG,&vrh);
    dodaj(b,stog,MAXSTOG,&vrh);
    dodaj(c,stog,MAXSTOG,&vrh);
    skinii(&a,stog,&vrh);
    skinii(&c,stog,&vrh);
    skinii(&b,stog,&vrh);
    printf("%d %d %d",a,b,c);
}
```

a) 3 1 2

b) 1 2 3

c) 2 3 1

d) 1 3 2

e) 2 1 3

17. Na stog se pohranjuju samo cijeli brojevi. Prototip funkcije za skidanje cijelog broja sa stoga je (funkcija vraća 0 ili 1, ovisno o tome da li se zapis uspio skinuti s vrha stoga):

- a) int skini(int stavka, int stog[], int *vrhStog);
- b) int skini(int *stavka, int stog[], int *vrhStog);
- c) int skini(float stavka, float stog[], int vrhStog);
- d) void *skini(int *stavka, int stog[], int n, int *vrhStog);
- e) int *skini(int *stavka, int stog[], int vrhStog);

18. Koja od sljedećih nizova naredbi u pseudokodu će zamijeniti vrijednost varijabli A i B pomoću stoga:

- a) stavi(A); skini(B); stavi(B); skini(A);
- b) stavi(B); skini(A);
- c) stavi(A); skini(B);
- d) stavi(A); stavi(Pom); stavi(B); stavi(Pom); skini(A); stavi(Pom); skini(B);
- e) stavi(A); stavi(B); skini(A); skini(B);

Koja od sljedećih tvrdnji nije istinita ?

- a) Funkcija func2 je složenosti $O(n \log 2n)$
- b) Funkcija func1 će se brže izvršiti

36. Ako imamo cjelobrojni stog i funkciju uzmi koja uzima element sa stoga i ima sljedeći prototip (funkcija vraća 1 ako je uspješno skinula element, a 0 ako nije):

```
int uzmi(int STOG[], int n, int *vrh, int *stavka);
```

kako bi se napisala funkcija koja računa broj elemenata na stogu

a) int br_elem(int STOG[], int n, int *vrh) {

```
    int elem, br=0;  
  
    while( uzmi(STOG, n, vrh, elem) != 0 ) br++;  
  
    return br;  
}
```

b) int br_elem(int STOG[], int n, int *vrh) {

```
    int elem, br=0;  
  
    while( uzmi(STOG, n, vrh, &elem) != 0 ) br++;  
  
    return br;  
}
```

c) int br_elem(int STOG[], int n, int *vrh) {

```
    int elem, br=0;  
  
    while( uzmi(STOG, n, *vrh, &elem) != 0 ) br++;  
  
    return br;  
}
```

d) int br_elem(int STOG[], int n, int *vrh) {

```

int elem, br=0;

while( uzmi(&STOG, n, vrh, &elem) != 0 ) br++;

return br;

}

e) int br_elem( int STOG[], int n, int *vrh) {

int elem, br=0;

while( uzmi(&STOG[0], n, vrh, elem) != 0 ) br++;

return br;

}

```

37. Složenost funkcije

```

int dodaj (zapis stavka, zapis stog[], int n, int *vrh) {

if (*vrh >= n-1) return 0;

(*vrh)++;

stog [*vrh] = stavka;

return 1;

}

```

je:

- a) složenost ovisi o veličini zapisa stavke, pa se ne može jednoznačno odrediti
- b) $O(\log n)$
- c) $O(\log_2 n)$
- d) $O(n)$
- e) $O(1)$

39. Koja od sljedećih tvrdnji vezanih uz stog je istinita?

- a) Interni stog računala koristi se samo pri deklaraciji globalnih varijabli C programa
- b) pojedina operacija dodaj (push) i brisi (pop) zahtijeva jednako vremena bez obzira na broj pohranjenih podataka
- c) stog je programska struktura u koju se dodaju i brišu elementi po načelu FIFO (First In First Out)
- d) za programsku realizaciju stoga moguće je isključivo koristiti stog koji je definiran kao polje
- e) na stog se mogu pohraniti isključivo cijelobrojni (int) podaci

41. Kakav je sadržaj stoga nakon izvođenja funkcije funkcija, ako je stog prije poziva prazan?

Funkcije za operacije nad stogom skinu i dodaj vraćaju 1 ako su obavile traženu zadaću, a 0 ako nisu, te imaju sljedeće prototipe:

```
int skini (int *stavka, int stog[], int *vrh)  
int dodaj (int stavka, int stog[], int n, int *vrh)
```

```
#include <stdio.h>  
  
#define MAXSTOG 100  
  
void funkcija() {  
  
    int stog[MAXSTOG], pomStog[MAXSTOG];  
  
    int i, vrh = -1, pomVrh = -1;  
  
    while (skini(&i, stog, &vrh)) {  
  
        if (i>=0) dodaj(i, pomStog, MAXSTOG, &pomVrh);  
  
    }  
  
    while (skini(&i, pomStog, &pomVrh)) {  
  
        if (i<0){
```

```
dodaj(i, stog, MAXSTOG, &vrh);  
}  
}  
}
```

a) Stog sadrži samo elemente ≤ 0

b) Stog je prazan

c) Sadržaj stoga je nepoznat

d) Sadržaj stoga je nepromijenjen

e) Stog sadrži samo elemente > 0

75. Ako push stavlja na stog i vraca 1 za uspjesno stavljanje a 0 za neuspjesno,

te pop skida sa stoga i vraca skinuti element za uspjesno skidanje ili -1 za neuspjesno odredi sto ce se nalaziti ne stogu.

```
push(5);  
push(push(pop()));
```

RJESENJE: 51

77. Prepostavka je da postoje funkcije za operacije nad stogom skinji dodaj sa sljedećim prototipima:

```
int skinji (int *stavka, int stog[], int *vrh)
```

```
int dodaj (int stavka, int stog[], int n, int *vrh)
```

78. Što će ispisati sljedeći program?

```
#include <stdio.h>

#define MAXSTOG 100

int main() {
    int stog[MAXSTOG],vrh=-1;
    int a=1,b=2,c=3;
    dodaj(a,stog,MAXSTOG,&vrh);
    dodaj(b,stog,MAXSTOG,&vrh);
    dodaj(c,stog,MAXSTOG,&vrh);

    skini( &a ,stog,&vrh);
    skini( &c ,stog,&vrh);
    skini( &b ,stog,&vrh);

    printf("%d %d %d",a,b,c);
}

*a) 3 1 2
b) 1 2 3
c) 2 3 1
d) 1 3 2
e) 2 1 3
```

87. Ako imamo cjelobrojni stog i funkciju uzmi koja uzima element sa stoga i ima sljedeći prototip (funkcija vraća 1 ako je uspješno skinula element, a 0 ako nije):

```
int uzmi(int STOG[], int n, int *vrh, int *stavka);
```

kako bi se napisala funkcija koja računa broj elemenata na stogu

```
**a) int br_elem( int STOG[], int n, int *vrh) {  
    int elem, br=0;  
    while( uzmi(STOG, n, vrh, elem) != 0 )  
        br++;  
    return br;  
}  
  
b) int br_elem( int STOG[], int n, int *vrh) {  
    int elem, br=0;  
    while( uzmi(STOG, n, vrh, &elem) != 0 )  
        br++;  
    return br;  
}  
  
c) int br_elem( int STOG[], int n, int *vrh) {  
    int elem, br=0;  
    while( uzmi(STOG, n, *vrh, &elem) != 0 )  
        br++;  
    return br;  
}  
  
d) int br_elem( int STOG[], int n, int *vrh) {  
    int elem, br=0;
```

```

while( uzmi(&STOG, n, vrh, &elem) != 0 )

br++;

return br;

}

e) int br_elem( int STOG[], int n, int *vrh) {

int elem, br=0;

while( uzmi(&STOG[0], n, vrh, elem) != 0 )

br++;

return br;

}

```

88. Složenost funkcije

```

int dodaj (zapis stavka, zapis stog[], int n, int *vrh) {

if (*vrh >= n-1) return 0;

(*vrh)++;

stog [*vrh] = stavka;

return 1;

}

```

je:

- a) složenost ovisi o veličini zapisa stavke, pa se ne može jednoznačno odrediti
- b) $O(\log n)$
- c) $O(\log_2 n)$
- d) $O(n)$
- *e) $O(1)$

90. Koja od sljedećih tvrdnji vezanih uz stog je istinita?

- a) interni stog računala koristi se samo pri deklaraciji globalnih varijabli C programa
- *b) pojedina operacija dodaj (push) i brisi (pop) zahtijeva jednako vremena bez obzira na broj pohranjenih podataka
- c) stog je programska struktura u koju se dodaju i brišu elementi po načelu FIFO (First In First Out)
- d) za programsku realizaciju stoga moguće je isključivo koristiti stog koji je definiran kao polje
- e) na stog se mogu pohraniti isključivo cijelobrojni (int) podaci

92. Kakav je sadržaj stoga nakon izvođenja funkcije funkcija, ako je stog prije poziva prazan?

Funkcije za operacije nad stogom skinu i dodaj vraćaju 1 ako su obavile traženu zadaću,

a 0 ako nisu, te imaju sljedeće prototipe:

```
int skini (int *stavka, int stog[], int *vrh)
```

```
int dodaj (int stavka, int stog[], int n, int *vrh)
```

```
#include <stdio.h>
```

```
#define MAXSTOG 100
```

```
void funkcija() {
```

```
    int stog[MAXSTOG],
```

```
    pomStog[MAXSTOG];
```

```
    int i, vrh = -1, pomVrh = -1;
```

```
    while (skini(&i, stog, &vrh)) {
```

```
        if (i>=0) dodaj(i, pomStog, MAXSTOG, &pomVrh);
```

```
}
```

```
while (skini(&i, pomStog, &pomVrh)) {  
    if (i<0){  
        dodaj(i, stog, MAXSTOG, &vrh);  
    } }  
}
```

a) Stog sadrži samo elemente ≤ 0

*b) Stog je prazan

c) Sadržaj stoga je nepoznat

d) Sadržaj stoga je nepromijenjen

e) Stog sadrži samo elemente > 0

95.

```
int stavi (int polje[], int *vrh, int n, int element);
```

```
int skinii (int polje[], int *vrh, int element);
```

vraćaju 1 za uspješno inače 0.

Što će se ispisati sljedećim:

```
void func(){  
    int i;  
    int polje[10];  
    int vrh;  
    for(i=0;i<10;i++){  
        stavi(polje, &vrh, 5, i);  
    }  
    while (skini(polje,&vrh,&i)){  
        printf("%d",i);  
    }  
}
```

}

Rj: e) 4 3 2 1 0

99. Funkcija stavljanja na stog (int(push(int element))) vraća

1 za uspjeh, 0-neuspjeh, funkcija skidanja (int(pop())) vraća

vrijednost elementa s vrha ili -1 ako je stog prazan.

Što će biti?

pop(push(push(pop()))) (stog je bio prazan)

Rj: b)-1

102. Ako imamo cijelobrojni stog i funkciju koja uzima element sa stoga:

int uzmi(int stog[], int n, int *vrh, int *stavka);

koja vraća 1 za uspjeh inače 0.

Kako bi se napisala funkcija koja vraća broj el. na stogu?

Rj:

```
int br_elem(int stog[], int n, int *vrh){  
    int elem_br=0;  
    while(uzmi(stog, n, vrh, &elem)!=-1) br ++;  
}
```

```
return br;
```

103.

Na stog se pohranjuju samo cijeli brojevi. Prototip funkcije za stavljanje cijelog broja na stog je (funkcija vraća 0 ili 1 ovisno o tome da li se zapis uspio pohraniti na vrh stoga):

- a)int dodaj(int stavka, int stog[], int n, int VrhStog);
- b)int dodaj(float stavka, float stog[], int n, int VrhStog);
- c)void dodaj(int stavka, int stog[], int n, int *VrhStog);
- *d)int dodaj(int stavka, int stog[], int n, int *VrhStog);
- e)int *dodaj(int *stavka, int stog[], int n, int VrhStog);

107.

Za stog realiziran cjelobrojnim poljem postoje funkcije push i pull koje stavljuju, odnosno uzimaju element sa stoga. Ukoliko je vrh stoga na lijevoj strani, što će se nalaziti na stogu nakon izvršavanja sljedećeg programskog odsječka (na početku je stog prazan):

```
for(i=1;i<=10;i++)  
    push(i);  
  
for(j=1;j<=5;j++)  
    pull();
```

*a) 6 7 8 9 10

- b) 1 2 3 4 5
- c) 1
- d) 1 2 3 4 5 6 7 8 9 10
- e) neće biti više elemenata na stogu

112. Ako funkcija stavljanja na stog vraša 1 u služaju uspjeha a 0 u služaju neuspjeha i ima prototip

```
int push (int element);
```

a funkcija skidanja sa stoga vraša vrijednost element s vrha ili -1 ako je stog prazan i ima prototip

```
int pop ();
```

Uto Še biti na stogu nakon obavljanja sljedeših naredbi, uz prepostavku da je stog bio prazan i da stog raste s lijeva na desno:

```
push(push(pop()));  
pop();
```

- a) Stog Še biti prazan.
- b) -1 0
- c) 1
- *d) -1
- e) 1 -1

114. Stog je struktura za koju vrijedi:

- *a) Da bi pristupili elementu s dna stoga, potrebno je sve ostale skinuti.
- b) Zadnji element koji smo stavili na stog zadnjega Šemo i skinuti
- c) Omogušava direktni pristup svakom upisanom elementu
- d) FIFO
- e) Ništa od navedenog

116. Ako imamo cjelobrojni stog i funkciju uzmi koja uzima element sa stoga i ima sljedeći prototip (funkcija vraća 1 ako je uspješno skinula element, a 0 ako nije):

```
int uzmi(int STOG[], int n, int *vrh, int *stavka);
```

kako bi se napisala funkcija koja računa broj elemenata na stogu

```
a) int br_elem( int STOG[], int n, int *vrh) {
    int elem, br=0;
    while( uzmi(&STOG, n, vrh, &elem) != 0 ) br++;
    return br;
}

*b) int br_elem( int STOG[], int n, int *vrh) {
    int elem, br=0;
    while( uzmi(STOG, n, vrh, &elem) != 0 ) br++;
    return br;
}

c) int br_elem( int STOG[], int n, int *vrh) {
    int elem, br=0;
    while( uzmi(&STOG[0], n, vrh, elem) != 0 ) br++;
    return br;
}
```

```

}

d) int br_elem( int STOG[], int n, int *vrh) {

    int elem, br=0;

    while( uzmi(STOG, n, *vrh, &elem) != 0 ) br++;

    return br;

}

e) int br_elem( int STOG[], int n, int *vrh) {

    int elem, br=0;

    while( uzmi(STOG, n, vrh, elem) != 0 ) br++;

    return br;

}

```

119. Ako push stavlja na stog i vraca 1 za uspjesno stavljanje a 0 za neuspjesno, te pop skida sa stoga i vraca skinuti element za uspjesno skidanje ili -1 za neuspjesno odredi sto ce se nalaziti ne stogu.

```

push(5);

push(push(pop()));

```

RJESENJE: 51

120. Ako push stavlja na stog i vraca 1 za uspjesno stavljanje te 0 za neuspjesno, a pull skida sa stoga i vraca vrijednost skinutog elementa te se uzima da ne postoji slucaj da pull ne uspije skinuti sa stoga odredi sto ce se nalaziti na stogu.

```

push(push(push(5))+pull());

```

122. Prepostavka je da postoje funkcije za operacije nad stogom skini i dodaj sa sljedećim prototipima:

```
int skini (int *stavka, int stog[], int *vrh)  
int dodaj (int stavka, int stog[], int n, int *vrh)
```

Što će ispisati sljedeći program?

```
#include <stdio.h>  
  
#define MAXSTOG 100  
  
int main() {  
  
    int stog[MAXSTOG], vrh=-1;  
  
    int a=1, b=2, c=3;  
  
    dodaj(a, stog, MAXSTOG, &vrh);  
  
    dodaj(b, stog, MAXSTOG, &vrh);  
  
    dodaj(c, stog, MAXSTOG, &vrh);  
  
    skini(&a, stog, &vrh);  
  
    skini(&c, stog, &vrh);  
  
    skini(&b, stog, &vrh);  
  
    printf("%d %d %d", a, b, c);}  
  
*a) 3 1 2  
b) 1 2 3  
c) 2 3 1  
d) 1 3 2  
e) 2 1 3
```

126. Na stog se pohranjuju samo cijeli brojevi. Prototip funkcije za skidanje cijelog broja sa stoga je

(funkcija vraća 0 ili 1, ovisno o tome da li se zapis uspio skinuti s vrha stoga):

- a) int skini(int stavka, int stog[], int *vrhStog);
- *b) int skini(int *stavka, int stog[], int *vrhStog);
- c) int skini(float stavka, float stog[], int vrhStog);
- d) void *skini(int *stavka, int stog[], int n, int *vrhStog);
- e) int *skini(int *stavka, int stog[], int vrhStog);

127. Koja od sljedećih nizova naredbi u pseudokodu će zamijeniti vrijednost varijabli A i B pomoću stoga:

- a) stavi(A); skini(B); stavi(B); skini(A);
- b) stavi(B); skini(A);
- c) stavi(A); skini(B);
- d) stavi(A); stavi(Pom); stavi(B); stavi(Pom); skini(A); stavi(Pom);skini(B);
- *e) stavi(A); stavi(B); skini(A); skini(B);

1. Prepostavka je da postoje funkcije za operacije nad stogom skini i dodaj sa sljedećim prototipima:

```
int skini (int *stavka, int stog[], int *vrh)
int dodaj (int stavka, int stog[], int n, int *vrh)
```

Što će ispisati sljedeći program?

```
#include <stdio.h>
#define MAXSTOG 100
int main() {
    int stog[MAXSTOG], vrh=-1;
    int a=1, b=2, c=3;
    dodaj (a, stog, MAXSTOG, &vrh);
    dodaj (b, stog, MAXSTOG, &vrh);
    dodaj (c, stog, MAXSTOG, &vrh);
```

```

    skini(&a, stog, &vrh);
    skini(&c, stog, &vrh);
    skini(&b, stog, &vrh);
    printf("%d %d %d", a, b, c);
}
a) 3 1 2
b) 1 2 3
c) 2 3 1
d) 1 3 2
e) 2 1 3

```

Rješenje: A

2. Na stog se pohranjuju samo cijeli brojevi. Prototip funkcije za skidanje cijelog broja sa stoga je (funkcija vraća 0 ili 1, ovisno o tome da li se zapis uspio skinuti s vrha stoga):
 - a) int skini(int stavka, int stog[], int *vrhStog);
 - b) int skini(int *stavka, int stog[], int *vrhStog);
 - c) int skini(float stavka, float stog[], int vrhStog);
 - d) void *skini(int *stavka, int stog[], int n, int *vrhStog);
 - e) int *skini(int *stavka, int stog[], int vrhStog);

Rješenje: B

3. Koja od sljedećih nizova naredbi u pseudokodu će zamijeniti vrijednost varijabli A i B pomoću stoga:
 - a) stavi(A); skini(B); stavi(B); skini(A);
 - b) stavi(B); skini(A);
 - c) stavi(A); skini(B);
 - d) stavi(A); stavi(Pom); stavi(B); stavi(Pom); skini(A);

stavi(Pom); skini(B);
 - e) stavi(A); stavi(B); skini(A); skini(B);

Rješenje: E

4. Ako push stavlja na stog i vraća 1 za uspješno stavljanje a 0 za neuspješno, te pop skida sa stoga i vraća skinuti element za uspješno skidanje ili -1 za neuspješno odredi sto će se nalaziti ne stogu.

```

push(5);
push(push(pop()));

```

5. Prepostavka je da postoje funkcije za operacije nad stogom skini i dodaj sa sljedećim prototipima:

```
int skini (int *stavka, int stog[], int *vrh)  
int dodaj (int stavka, int stog[], int n, int *vrh)
```

Što će ispisati sljedeći program?

```
#include <stdio.h>  
  
#define MAXSTOG 100  
  
int main() {  
  
    int stog[MAXSTOG], vrh=-1;  
  
    int a=1, b=2, c=3;  
  
    dodaj (a, stog, MAXSTOG, &vrh);  
    dodaj (b, stog, MAXSTOG, &vrh);  
    dodaj (c, stog, MAXSTOG, &vrh);  
  
    skini( &a ,stog,&vrh);  
    skini( &c ,stog,&vrh);  
    skini( &b ,stog,&vrh);  
  
    printf("%d %d %d", a,b,c);  
}
```

a) 3 1 2

b) 1 2 3

c) 2 3 1

d) 1 3 2

e) 2 1 3

Rješenje: A

6. Složenost funkcije

```
int dodaj (zapis stavka, zapis stog[], int n, int *vrh) {  
    if (*vrh >= n-1) return 0;  
    (*vrh)++;  
    stog [*vrh] = stavka;  
    return 1;  
}
```

a) složenost ovisi o veličini zapisa stavke, pa se ne može jednoznačno odrediti

b) $O(\log n)$

c) $O(\log_2 n)$

d) $O(n)$

Rješenje: E

e) $O(1)$

7. Prepostavka je da postoje funkcije za operacije nad stogom skini i dodaj sa sljedećim prototipima:

```
int skini (int *stavka, int stog[], int *vrh)  
int dodaj (int stavka, int stog[], int n, int *vrh)
```

Što će ispisati sljedeći program?

```
#include <stdio.h>  
#define MAXSTOG 100  
int main() {  
    int stog[MAXSTOG], vrh=-1;  
    int a=1, b=2, c=3;  
    dodaj (a, stog, MAXSTOG, &vrh);  
    dodaj (b, stog, MAXSTOG, &vrh);  
    dodaj (c, stog, MAXSTOG, &vrh);  
    skini (&a, stog, &vrh);  
    skini (&c, stog, &vrh);  
    skini (&b, stog, &vrh);  
    printf ("%d %d %d", a, b, c);  
}
```

a) 3 1 2

b) 1 2 3

c) 2 3 1

d) 1 3 2

e) 2 1 3

Rješenje: A

b)

8. Na stog se pohranjuju samo cijeli brojevi. Prototip funkcije za skidanje cijelog broja sa stoga je (funkcija vraća 0 ili 1, ovisno o tome da li se zapis uspio skinuti s vrha stoga):
- a) int skini(int stavka, int stog[], int *vrhStog);
 - b) int skini(int *stavka, int stog[], int *vrhStog);
 - c) int skini(float stavka, float stog[], int vrhStog);
 - d) void *skini(int *stavka, int stog[], int n, int *vrhStog);
 - e) int *skini(int *stavka, int stog[], int vrhStog);

Rješenje: B

9. Koja od sljedećih nizova naredbi u pseudokodu će zamijeniti vrijednost varijabli A i B pomoću stoga:
- a) stavi(A); skini(B); stavi(B); skini(A);
 - b) stavi(B); skini(A);
 - c) stavi(A); skini(B);
 - d) stavi(A); stavi(Pom); stavi(B); stavi(Pom); skini(A);
stavi(Pom); skini(B);
 - e) stavi(A); stavi(B); skini(A); skini(B);

Rješenje: E

10. Ako push stavlja na stog i vraća 1 za uspješno stavljanje a 0 za neuspješno, te pop skida sa stoga i vraća skinuti element za uspješno skidanje ili -1 za neuspješno odredi sto će se nalaziti ne stogu.

```
push(5);  
  
push(push(pop()));
```

11. Prepostavka je da postoje funkcije za operacije nad stogom skini i dodaj sa sljedećim prototipima:

```
int skini (int *stavka, int stog[], int *vrh)  
  
int dodaj (int stavka, int stog[], int n, int *vrh)
```

Što će ispisati sljedeći program?

```
#include <stdio.h>  
  
#define MAXSTOG 100  
  
int main() {  
  
    int stog[MAXSTOG], vrh=-1;  
  
    int a=1, b=2, c=3;
```

```

dodaj (a,stog,MAXSTOG,&vrh) ;
dodaj (b,stog,MAXSTOG,&vrh) ;
dodaj (c,stog,MAXSTOG,&vrh) ;

skinii( &a ,stog,&vrh);
skinii( &c ,stog,&vrh);
skinii( &b ,stog,&vrh);

printf("%d %d %d",a,b,c);

}

a) 3 1 2
b) 1 2 3
c) 2 3 1
d) 1 3 2
e) 2 1 3

```

Rješenje: A

12. Složenost funkcije

```

int dodaj (zapis stavka, zapis stog[], int n, int *vrh) {
    if (*vrh >= n-1) return 0;
    (*vrh)++;
    stog [*vrh] = stavka;
    return 1;
}

```

- a) složenost ovisi o veličini zapisa stavke, pa se ne može jednoznačno odrediti
- b) $O(\log n)$
- c) $O(\log_2 n)$
- d) $O(n)$
- e) $O(1)$

Rješenje: E

13. Koja od sljedećih tvrdnji vezanih uz stog je istinita?

- a) interni stog računala koristi se samo pri deklaraciji globalnih varijabli C programa
- b) pojedina operacija dodaj (push) i brisi (pop) zahtijeva jednako vremena bez obzira na broj pohranjenih podataka
- c) stog je programska struktura u koju se dodaju i brišu elementi po načelu FIFO (First In First Out)
- d) za programsku realizaciju stoga moguće je isključivo koristiti stog koji je definiran kao polje
- e) na stog se mogu pohraniti isključivo cjelobrojni (`int`) podaci

Rješenje: B

14. Kakav je sadržaj stoga nakon izvođenja funkcije funkcija, ako je stog prije poziva prazan? Funkcije za operacije nad stogom skinu i dodaj vraćaju 1 ako su obavile traženu zadaću, a 0 ako nisu, te imaju sljedeće prototipe:

```
int skini (int *stavka, int stog[], int *vrh)

int dodaj (int stavka, int stog[], int n, int *vrh)
```

```
#include <stdio.h>

#define MAXSTOG 100

void funkcija() {

    int stog[MAXSTOG],
        pomStog[MAXSTOG];
    int i, vrh = -1, pomVrh = -1;

    while (skini(&i, stog, &vrh)) {

        if (i>=0) dodaj(i, pomStog, MAXSTOG, &pomVrh);

    }

    while (skini(&i, pomStog, &pomVrh)) {

        if (i<0) {

            dodaj(i, stog, MAXSTOG, &vrh);

        }

    }

}
```

- a) Stog sadrži samo elemente ≤ 0

- b) Stog je prazan
- c) Sadržaj stoga je nepoznat
- d) Sadržaj stoga je nepromijenjen
- e) Stog sadrži samo elemente > 0

Rješenje: B

15. Funkcija stavljanja na stog (`int (push (int element))`) vraća 1 za uspjeh, 0-neuspjeh, funkcija skidanja (`int (pop ())`) vraća vrijednost elementa s vrha ili -1 ako je stog prazan. Što će biti?

`pop (push (push (pop ())))` (stog je bio prazan)

Rj: b)-1

16. Ako imamo cjelobrojni stog i funkciju koja uzima element sa stoga:

```
int uzmi(int stog[], int n, int *vrh, int *stavka);
```

koja vraća 1 za uspjeh inače 0.

Kako bi se napisala funkcija koja vraća broj el. na stogu?

Rj:

```
int br_elem(int stog[], int n, int *vrh) {
    int elem_br=0;
    while(uzmi(stog, n, vrh, &elem) !=0) br++;
}
return br;
```

17. Na stog se pohranjuju samo cijeli brojevi. Prototip funkcije za stavljanje cijelog broja na stog je (funkcija vraća 0 ili 1 ovisno o tome da li se zapis uspio pohraniti na vrh stoga):

- a) `int dodaj (int stavka, int stog[], int n, int VrhStog);`
- b) `int dodaj (float stavka, float stog[], int n, int VrhStog);`
- c) `void dodaj (int stavka, int stog[], int n, int *VrhStog);`
- d) `int dodaj (int stavka, int stog[], int n, int *VrhStog);`
- e) `int *dodaj (int *stavka, int stog[], int n, int VrhStog);`

Rješenje: D

18. Za stog realiziran cijelobrojnim poljem postoje funkcije `push` i `pull` koje stavljaju, odnosno uzimaju element sa stoga. Ukoliko je vrh stoga na lijevoj strani, što će se nalaziti na stogu nakon izvršavanja sljedećeg programskog odsječka (na početku je stog prazan):

```
for (i=1; i<=10; i++)
```

```
    push(i);
```

```
for (j=1; j<=5; j++)
```

```
    pull();
```

- a) 6 7 8 9 10
- b) 1 2 3 4 5
- c) 1
- d) 1 2 3 4 5 6 7 8 9 10
- e) neće biti više elemenata na stogu

Rješenje: A

19. Stog je struktura za koju vrijedi:

- a) Da bi pristupili elementu s dna stoga, potrebno je sve ostale skinuti.
- b) Zadnji element koji smo stavili na stog zadnjega ćemo i skinuti
- c) Omogućava direktni pristup svakom upisanom elementu
- d) FIFO
- e) Ništa od navedenog

Rješenje: A

20. Ako imamo cijelobrojni stog i funkciju uzmi koja uzima element sa stoga i ima sljedeći prototip (funkcija vraća 1 ako je uspješno skinula element, a 0 ako nije):

```
int uzmi(int STOG[], int n, int *vrh, int *stavka);
```

kako bi se napisala funkcija koja računa broj elemenata na stogu

```
a) int br_elem( int STOG[], int n, int *vrh) {  
    int elem, br=0;  
    while( uzmi(&STOG, n, vrh, &elem) != 0 ) br++;  
    return br;  
}
```

```

b) int br_elem( int STOG[], int n, int *vrh)  {

    int elem, br=0;

    while( uzmi(STOG, n, vrh, &elem) != 0 ) br++;

    return br;

}

c) int br_elem( int STOG[], int n, int *vrh)  {

    int elem, br=0;

    while( uzmi(&STOG[0], n, vrh, elem) != 0 ) br++;

    return br;

}

d) int br_elem( int STOG[], int n, int *vrh)  {

    int elem, br=0;

    while( uzmi(STOG, n, *vrh, &elem) != 0 ) br++;

    return br;

}

e) int br_elem( int STOG[], int n, int *vrh)  {

    int elem, br=0;

    while( uzmi(STOG, n, vrh, elem) != 0 ) br++;

    return br;

}

```

Rješenje: B (ili možda E)

21. Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha a 0 u slučaju neuspjeha i ima prototip
- ```
int push (int element);
```

a funkcija skidanja sa stoga vraća vrijednost element s vrha ili -1 ako je stog prazan i ima prototip

```
int pop ();
```

Što će biti na stogu nakon obavljanja sljedećih naredbi, uz prepostavku da je stog bio prazan i da stog raste s lijeva na desno:

```
push(push(pop()));
```

```
pop();
```

- a) Stog će biti prazan.
- b) -1 0
- c) 1
- d) -1
- e) 1 -1

Rješenje: D

22. Ako `push` stavlja na stog i vraća 1 za uspješno stavljanje te 0 za neuspješno, a `pull` skida sa stoga i vraća vrijednost skinutog elementa te se uzima da ne postoji slučaj da `pull` ne uspije skinuti sa stoga odredi što će se nalaziti na stogu.

```
push(push(push(5))+pull());
```

1. Najefikasniji algoritam stvaranja gomile od  $n$  elemenata za najgori slučaj ima složenost:
- a)  $O(n \cdot \log_2 n)$

- b)  $O(\log_2 n)$
- c)  $O(1)$
- d)  $O(n^2)$
- e)  $O(n)$

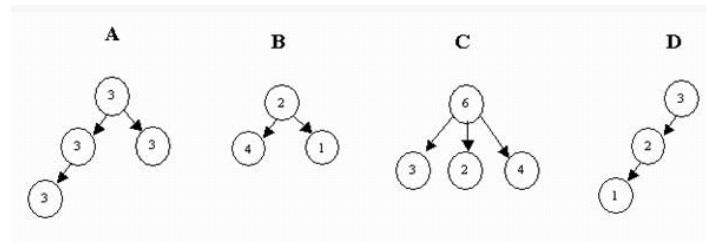
Rješenje: E

2. Stupanj stabla (koji ima  $n$  razina) je:
- a) najmanji stupanj nekog čvora u stablu

- b)  $n$
- c) najveći stupanj nekog čvora u stablu
- d) broj čvorova u stablu
- e) broj čvorova u potpunom stablu sa  $n$  razina

Rješenje: C

3. Koja od prikazanih stabala su gomile:



- a) A
- b) B
- c) C
- d) niti jedno od prikazanih stabala nije gomila
- e) A, B, C, D

Rješenje: A

4. Uz prethodno deklarirane sve podatke i već formirano binarno stablo, što radi sljedeća funkcija:

```

void ispis (cvor *glava) {
 if (glava != NULL) {
 ispis (glava -> lijevo);
 ispis (glava -> desno);
 printf ("%s \n", glava -> element);
 }
}

```

- a) uvijek ispisuje samo vrijednost elementa na koji pokazuje glava
- b) funkcija ne radi ništa jer je tipa void
- c) inorder ispisuje vrijednost elementa stabla
- d) preorder ispisuje vrijednost elementa stabla
- e) postorder ispisuje vrijednost elementa stabla

Rješenje: E

5. Koja od sljedećih tvrdnji je istinita:

- a) inorder i preorder obilaskom bit će obrađeni svi čvorovi u stablu
- b) obilazak preorder moguće je jedino primjeniti na *punim* stablima
- c) postorder obilazak uvijek obrađuje samo listove stabla
- d) preorder obilazak uvije obrađuje samo listove stabla

e) inorder obilazak stabla obrađuje dvostruko više elemenata nego postorder obilazak

Rješenje: A

6. Koliko čvorova ima koso stablo s n razina?

a)  $2^*n - 1$

b)  $n + 1$

c)  $2^n - 1$

d) n

e)  $2^n$

Rješenje: D

7. Koji od ponađenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 5, 10, 7, 3, 1, 90 algoritmom čija je složenost za najgori slučaj  $O(n \log_2 n)$ ?

a) 90

5 10

3 1 5

b) 90

10 7

5 1 3

c) 90

5 10

3 1 7

d) 90

10 7

3 1 5

e) 90

10 3

7 5 1

Rješenje: C

8. Što ispisuje funkcija

```
void ispisi(struct cvor *glava) {
 if(glava != NULL && glava->elem % 2) {
 printf(" %d ", glava->elem);
 ispisi(glava->slijed);
 }
}
```

ako se u jednostruko povezanoj listi na koju pokazuje parametar glava nalaze sljedeći cijeli brojevi :

1 57 43 13 8 11 20 10 56 53

- a) 1 57 43 13
- b) ne ispisuje ništa
- c) 1 57 43 13 8 11
- d) 1 57 43 13 8 11 20 10 56 53
- e) 1 57 43 13 11 53

Rješenje: A

9. Koji od sljedećih algoritama za sortiranje je najmanje uputno koristiti za sortiranje polja sa velikim brojem zapisa?

- a) Quick sort
- b) Bubble sort
- c) Merge sort
- d) Heap sort
- e) Shell sort

Rješenje: B

10. Koja procedura pronađe zadani element u jednostruko povezanoj listi?

```
a) cvor *trazi1 (cvor *glava, tip element) {
 cvor p;
 for (p=glava; p!=NULL; p=p->slijed)
 if (p->element != element) return p;
 return NULL;
```

```

}

b) cvor *trazi1 (cvor *glava, tip element) {

 cvor *p;

 for (p=glava; p!=NULL; p++)

 if (p->element != element) return p;

 return NULL;

}

c) cvor *trazi1 (cvor *glava, tip element) {

 cvor p;

 for (p=glava; p!=NULL; p++)

 if (p->element == element) return p;

 return NULL;

}

d) cvor *trazi1 (cvor *glava, tip element) {

 cvor p;

 if (p->element == element) return p;

 return NULL;

}

e) cvor *trazi1 (cvor *glava, tip element) {

 cvor *p;

 for (p = glava; p != NULL; p = p->sljed)

 if (p ->element == element) return p;

 return NULL;

}

```

Rješenje: E

11. Što radi sljedeća funkcija:

```
int fx (cvor *glava) {
```

```

if (glava) {
 return fx(glava->l) + fx(glava->d) + 1;
} else return 0;
}

```

- a) broji razine stabla
- b) računa zbroj elemenata u stablu
- c) vraća vrijednost  $\geq 1$  ako je stablo potpuno, 0 inače
- d) broji listove stabla
- e) broji čvorove stabla

Rješenje: E

12. Kod Quick sorta, najbolje je za stožer odabrati?
- a) slučajni odabir je najbolje

- b) svejedno je kako se bira stožer
- c) element srednji po vrijednosti u polju
- d) zadnji element polja
- e) prvi element polja

Rješenje: C

13. Neka jednostruko povezana lista sadrži čvorove sa sljedećim tipom zapisa:
- ```

struct s {
    int broj;
    struct s *sljed;
};

typedef struct s cvor;

```

Što radi funkcija f, ako je poziv funkcije f(glava, 7, &br)?

```

cvor *f(cvor *p, int broj, int *br) {
    if (p) {
        ++(*br);
    }
}

```

```

        if (p->broj == broj) return p;

        else return f(p->sljed, broj, br);

    } else {

        return NULL;

    }

}

```

Napomena: varijabla glava je pokazivač na prvi čvor u listi, a varijabla br je deklarirana i inicijalizirana kao int br = 0;

- a) Funkcija vraća pokazivač na prvi čvor u listi, te broj čvorova u listi koji sadrže broj 7
- b) Funkcija vraća pokazivač na početni čvor u listi, te broj čvorova u listi koji sadrži broj 7
- c) ??
- d) ?? Ne vidi se na screenshotu ☺
- e) ??

Rješenje: D

14. Vrijeme izvođenja sorta umetanjem je:

- a) $O(n^3)$
- b) $O(n * \log_2 n)$
- c) $O(n^2 * \log_2 n)$
- d) $O(n^2)$
- e) $O(n)$

Rješenje: D

15. Što će se ispisati funkcijom:

```

void ispis (cvor *korijen) {

    printf ("%c", korijen->element);

    if (korijen->lijevo && korijen->desno) {

        ispis (korijen->desno);

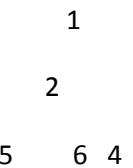
        ispis (korijen->lijevo);

    }
}

```

}

za stablo na slici pozivom funkcije



ispis (korijen);

ako je korijen u trenutku poziva pokazivač na korijen stabla?

- a) 134625
- b) 123456
- c) 521634
- d) 13462
- e) 12364

Rješenje: D

16. Ispravna deklaracija dvostruko povezane liste u memoriji glasi:

a) struct s1 {
 int mbr;
 char ime_pr[50];
 int spol;
 long pret;
 long sljed;
}

b) struct s1 {
 int mbr;
 char ime_pr[50];
 int spol;
 zapis *pret;
 zapis *sljed;
}

c) struct s1 {

```

    int mbr;
    char ime_pr[50];
    int spol;
    struct s1 *sljed;
}

d) typedef struct s1{
    int mbr;
    char ime_pr[50];
    int spol;
} zapis1;

typedef struct s2{
    zapis1 element;
    struct s2 *pred;
    struct s2 *sljed;
} zapis;

e) struct s1 {
    int mbr;
    char ime_pr[50];
    int spol;
    long *pret;
    long *sljed;
}

```

Rješenje: D

17. Koliko iznose prosječna vremena izvođenja Merge i Quick sorta?

- a) $O(n \log_2 n)$ i $O(\log_2 n)$
- b) $O(n \log_2 n)$ i $O(n)$
- c) $O(n \log_2 n)$ i $O(n \log_2 n)$

d) $O(n \log 2n)$ i $O(n)$

Rješenje: ?

e) $O(\log 2n)$ i $O(\log 2n)$

18. Kada se gomila oblikuje dodavanjem jednog po jednog elementa u stablo uz očuvanje strukture gomile, tada je vrijeme izvođenja oblikovanja gomile za najgori slučaj (n je broj ulaznih elemenata):

a) $O(\log 2n)$

b) $O(n)$

c) $O(n^2 * \log 2n)$

d) $O(n * \log 2n)$

Rješenje: D

e) $O(n^2)$

19. Što radi slijedeća funkcija?

```
int f(cvor *glava) {  
    int i = 0;  
  
    if (glava) {  
  
        if (glava->lijevo || glava->desno) i++;  
  
        i += f(glava->lijevo);  
  
        i += f(glava->desno);  
  
    }  
  
    return i;  
}
```

a) Broji čvorove u stablu koji imaju lijevo dijete.

b) Broji čvorove u stablu koji imaju oba djeteta.

c) Broji čvorove u stablu koji imaju desno dijete.

d) Niša od navedenog.

Rješenje: E

e) Broji čvorove u stablu koji nisu listovi (imaju bar jedno dijete).

20. Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 5, 10, 7, 3, 1, 90 algoritmom čija je složenost za najgori slučaj $O(n)$?
- a) 90

7 10

3 1 5

b) 90

10 7

3 1 5

c) 90

5 10

3 1 5

d) 90

10 7

5 1 3

e) 5

10 7

Rješenje: B

3 1 5

21. Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 50 5 7 10 13 1 8 algoritmom čija je složenost za najgori slučaj $O(n \log 2n)$?
- a) 50

13 10

8 7 5 1

b) 50

5 7

10 13 1 8

c) 50

13 7

5 10 1 8

d) 50

13 8

5 10 1 7

e) 50

13 8

Rješenje: ?

10 7 5 1

22. Koja od sljedećih tvrdnji NIJE istinita?

Slika (odnosi se samo na dva ponuđena odgovora):

NEMA SLIKE (stablo, nije puno)

a) Svi čvorovi sa stabla na slici su istog stupnja

b) U stablu sa slike listovi su: $\{h,i,e,f,j,k\}$

c) Maksimalni broj čvorova binarnog stabla na k-toj razini jednak je $2^k - 1$

d) Maksimalni broj čvorova binarnog stabla dubine k jednak je $2^k - 1$ za $k > 0$

e) Binarno stablo koje je visine k i ima $2^k - 1$ elemenata naziva se puno (full) binarno stablo

Rješenje: ?

23. Sortirano binarno stablo na slici (lijevo manji element, a desno veći) generirano je sljedećim ulaznim nizom brojeva:

NEMA SLIKE

a) 45, 4, 2, 9, 11, 33, 18

b) 45, 4, 33, 11, 2, 9, 18

c) 45, 9, 11, 18, 2, 33, 4

d) 9, 11, 18, 2, 33, 4, 45

Rješenje: ?

e) 2, 4, 9, 11, 18, 33, 45

24. Što će vratiti priložena funkcija za zadanu jednostruko povezanu linearu listu:

Slika (unutar čvorova prikazan je vrijednost varijable element):

```
int f(cvor *glava) {  
    if (glava) {  
        if (glava->element > 3)  
            return glava->element + f(glava->sljed);  
        else  
            return f(glava->sljed);  
    } else {  
        return 0;  
    }  
}
```

a) 6

b) 13

c) 0

d) 19

Rješenje: ?

e) 16

25. Koliko razina ima potpuno binarno stablo koje sadrži 100 čvorova i koliki je broj čvorova na posljednjoj razini ?

a) broj razina =6 broj čvorova=64

b) broj razina =7 broj čvorova=37

c) broj razina =7 broj čvorova=50

d) broj razina =7 broj čvorova=64

Rješenje: ?

e) broj razina =6 broj čvorova=50

26. Koji od ponađenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 5, 10, 7, 3, 1, 90 algoritmom čija je složenost za najgori slučaj $O(n)$?
- a) 90

5 10

3 1 5

b) 90

10 7

3 1 5

c) 90

7 10

3 1 5

d) 90

10 7

5 1 3

e) 5

10 7

Rješenje: ?

3 1 5

27. Zadano je prvih nekoliko koraka sorta ubacivanjem (polje se sortira uzlazno slijeva na desno):

12 5 9 88 23 41 4 13

5 12 9 88 23 41 4 13

5 9 12 88 23 41 4 13

5 9 12 23 88 41 4 13

Kako će izgledati polje nakon slijedeće zamjene dvaju elemenata tijekom sortiranja ubacivanjem?

- a) 5 9 12 23 4 41 88 13
- b) 5 9 12 23 41 88 4 13
- c) 5 9 12 23 88 41 13 4
- d) 5 9 12 23 88 4 41 13
- e) 5 9 12 23 13 41 4 88

Rješenje: ?

28. Koji sort ima najveće memorijske zahtjeve?

RJ: Merge Sort

29. Struktura stog se dinamički najčešće predstavlja?

RJ: Jednostruko povezanom linearnom listom

30. Zadano polje brojeva

66	88	99	22	77	55	33	11
----	----	----	----	----	----	----	----

sortira se Shell Sortom sa korakom k=3. Nakon koraka polje izgleda:

22	11	55	33	77	99	66	88
----	----	----	----	----	----	----	----

31. Koja od ponuđenih funkcija ispravno implementira Heap sort?

```
a) void HeapSort (tip A[], int n) {  
    int i;  
    StvoriGomilu (A, n/2);  
    for (i = n/2; i <=0 ; i--) {  
        Zamijeni (&A[1], &A[i]);  
        Podesi (A, 1, i-1);  
    }  
}
```

```
}
```

```
b) void HeapSort (tip A[], int n) {  
    int i;  
    StvoriGomilu (A, n);  
    for (i = n; i >= 2; i--) {  
        Zamijeni (&A[1], &A[i]);  
        Podesi (A, 1, i-1);  
    }  
}
```

```
c) void HeapSort (tip A[], int n) {  
    int i;  
    StvoriGomilu (A, 1);  
    for (i = 1; i <= n/2; i++) {  
        Zamijeni (&A[n], &A[1]);  
        Podesi (A, 1, i+1);  
    }  
}
```

```
d) void HeapSort (tip A[], int n) {  
    int i;  
    StvoriGomilu (A, 1);  
    for (i = 1; i <= n; i++) {  
        Zamijeni (&A[n], &A[1]);  
        Podesi (A, 1, i+1);  
    }  
}
```

```
e) void HeapSort (tip A[], int n) {  
    int i;  
    StvoriGomilu (A, n);  
    for (i = n/2; i <=0 ; i--) {
```

```

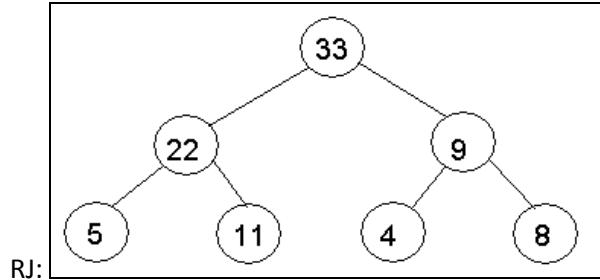
Zamijeni (&A[1], &A[i]);
Podesi (A, 1, i-1);
}
}

}

```

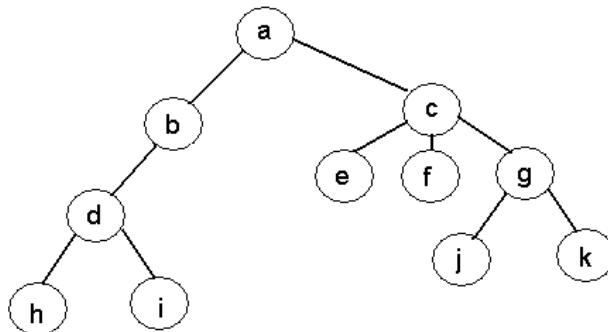
Rješenje: ?

32. Koja slika prikazuje gomilu oblikovanu ulaznim nizom (11, 33, 4, 5, 22, 59) tako da za najgori slučaj složenost bude $O(n \log_2 n)$?



33. Pretraživanje binarnog stabla **najbrže** je ako se radi o: **Sortiranom potpunom stablu**

34. Koja od slijedećih tvrdnji NIJE istinita (vezano uz sliku)?



RJ: Svi čvorovi na slici su istog stupnja.

35. Što radi:

```

void ispisi(cvor *glava) {
    cvor *p;
    for (p=glava; p!=NULL;p=p->sljed)

```

```
    printf("%d\n", p->element);  
}
```

RJ: Ispisuje sve vrijednosti elemenata jednostruko povezane linearne liste.

36. U prazno binarno stablo uneseni su elementi 20, 15, 1, 3, 7, 48, 12, 19, 35. Kolika je dubina stabla?

RJ: 6

37. Koja je od slijedećih tvrdnji za gomilu točna?

RJ: Gomila se koristi kada je do najvećeg/najmanjeg potrebno doći sa složenošću $O(1)$. Složenost reorganizacije nakon uklanjanja prvog člana je $O(\log_2 n)$. Složenost dodavanja novog člana u gomilu je $O(\log_2 n)$.

38. Koja je od slijedećih tvrdnji za sortove istinita?

RJ: Najgori slučaj vremena izvođenja Shell Sorta je $O(n^2)$

39. Koji od sljedećih algoritama za sortiranje je najmanje uputno koristiti za sortiranje polja sa velikim brojem zapisa?

- a) Quick sort
- b) Bubble sort
- c) Merge sort
- d) Heap sort
- e) Shell sort

Rješenje: B

40. Koji sort ima najveće memorijske zahtjeve?

RJ: Merge Sort

41. **bubble sort** - $O(n^2)$

42. **Sort umetanjem** - bavlja se n-1 prolaza kroz polje.

U prolazu i, $i=1, \dots, n-1$ postiže se uređenost prvih $i+1$ elemenata

tako da se na pravoj poziciji napravi slobodno mjesto za element s indeksom i.

Vrijeme izvođenja je $O(n^2)$.

43. **heap sort** - element s vrha gomile zamjenjuje se s posljednjim elementom polja, gomila se skraćuje za 1 element i podešava. Složenost podešavanja je $O(\log 2n)$.

To se obavlja n puta pa je složenost sorta $O(n \log 2n)$.

44. **shell sort** - $O(n^2)$.

45. **merge sort** - Na temelju dva sortirana polja (A i B) puni se treće (C).

Koristi se strategija "podijeli pa vladaj" uz rekurziju. $O(n \log 2 n)$

46. **quick sort** - Prosječno vrijeme $O(n \log 2 n)$, najgore $O(n^2)$

Ako je broj članova polja S jednak 0 ili 1, povratak u pozivni program.

Odabratи bilo koji član v u polju S. To je stožer.

Podijeli preostale članove polja S, $S \setminus \{v\}$ u dva odvojena skupa:

$$S1 = \{x \in S \setminus \{v\} \mid x \leq v\} \text{ i } S2 = \{x \in S \setminus \{v\} \mid x \geq v\}$$

Vrati $\{\text{quicksort}(S1), v, \text{quicksort}(S2)\}$

47. Kod Quick sorta, najbolje je za stožer odabrat?

- a) slučajni odabir je najbolje
- b) svejedno je kako se bira stožer
- c) element srednji po vrijednosti u polju
- d) zadnji element polja
- e) prvi element polja

Rješenje: C

48. Zadano polje brojeva

66 88 99 22 77 55 33 11

sortira se Shell Sortom sa korakom k=3. Nakon koraka polje izgleda:

22 11 55 33 77 99 66 88

49. Koja je od slijedećih tvrdnji za gomilu točna?

Gomila se koristi kada je do najvećeg/najmanjeg potrebno doći sa složenošću $O(1)$.

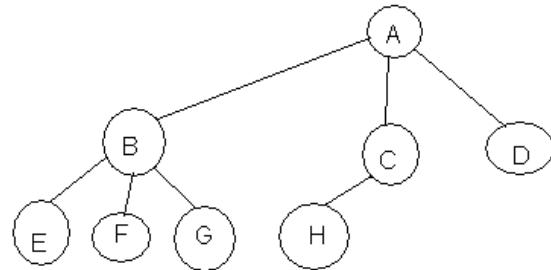
Složenost reorganizacije nakon uklanjanja prvog člana je $O(\log 2 n)$.

Složenost dodavanja novog člana u gomilu je $O(\log 2 n)$.

Rješenje: ?

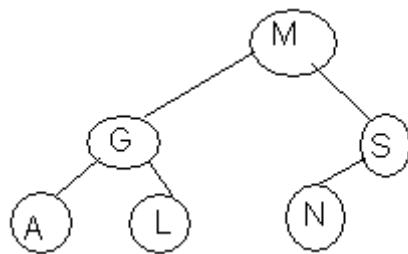
Algoritmi i strukture podataka 3. blic – pitanja s foruma ak. god. 2005/06

1. Stablo ima sljedeće atribute:



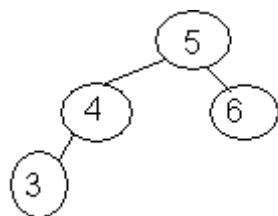
RJ: stupanj = 3, dubina = 3, potpuno

2. Zadana je funkcija (ne sjećam se koja), što se ispisuje?



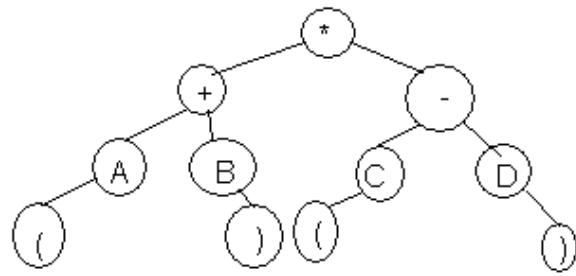
RJ: G M

3. Zadano je 5, 4, 6, 3 i napravi se stablo i ima funkcija i treba znati što ispisuje:



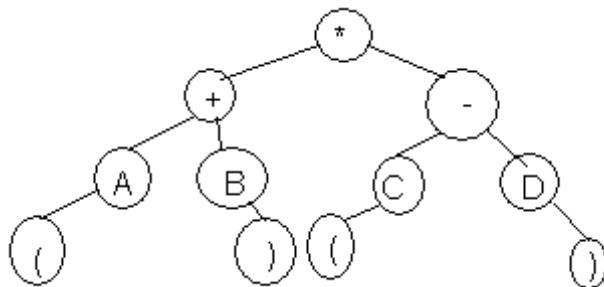
RJ: 3 4 5 6

4. Inorder obilazak stabla na slici daje izraz:



RJ: $(A + B) * (C - D)$

5. Postorder obilazak stabla na slici daje izraz:



RJ: $(A) B + (C) D - *$

6. Koliko razina ima potpuno binarno stablo koje sadrži 100 čvorova i koliki je broj čvorova na posljednjoj razini ?

a) broj razina =6 broj čvorova=64

b) broj razina=7 broj čvorova=37 ←

c) broj razina =7 broj čvorova=50

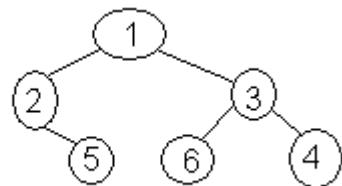
d) broj razina =7 broj čvorova=64

e) broj razina =6 broj čvorova=50

7. Što će se ispisati funkcijom:

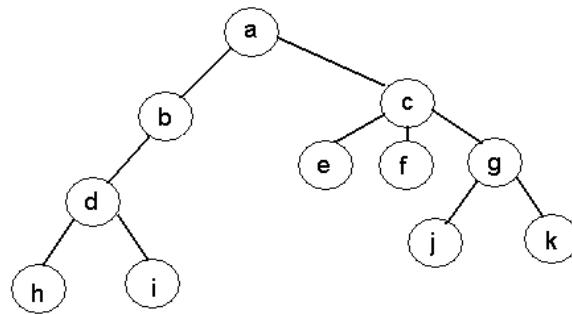
```
void ispis (cvor *korijen) {  
    printf ("%c", korijen->element);  
    if (korijen->lijevo && korijen->desno) {  
        ispis (korijen->desno);  
        ispis (korijen->lijevo);  
    }  
}
```

za stablo na slici pozivom funkcije?



RJ: 1 3 4 6 2

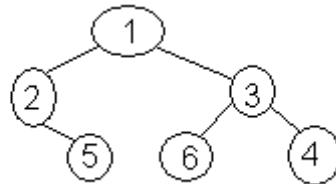
8. Koja od sljedećih tvrdnji nije istinita:



RJ:

- a) Svi čvorovi sa stabla na slici su istog stupnja ←
 - b) U stablu sa slike listovi su:{h,i,e,f,j,k}
 - c) Maksimalni broj čvorova binarnog stabla na k-toj razini jednak je $2^k - 1$
 - d) Maksimalni broj čvorova binarnog stabla dubine k jednak je $2^k - 1$ za $k > 0$
 - e) Binarno stablo koje je visine k i ima $2^k - 1$ elemenata naziva se puno (full) binarno stablo
-

9. Zadana je neka funkcija (mislim postorder) i treba znati što ispisuje:



RJ: 4 6 3 5 2 1

10. Stupanj stabla (koji ima n razina) je:

RJ:

- a) najmanji stupanj nekog čvora u stablu
- b) n
- c) najveći stupanj nekog čvora u stablu ←

d) broj čvorova u stablu

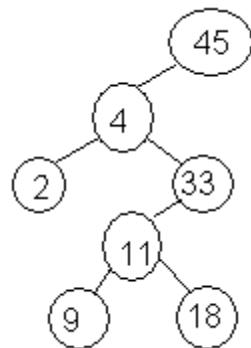
e) broj čvorova u potpunom stablu s n razina

11. Koliko čvorova ima koso stablo sa n razina:

RJ:

- a) 2^*n-1
 - b) $n+1$
 - c) 2^n-1
 - d) $n \leftarrow$**
 - e) 2^n
-

12.



Sortirano binarno stablo na slici (lijevo manji element, desno veći) generirano je sljedećim ulaznim nizom brojeva:

RJ: 45, 4, 33, 11, 2, 9, 18

13. U prazno binarno stablo uneseni su elementi 20, 15, 1, 3, 7, 48, 12, 19, 35. Kolika je dubina stabla?

RJ: 6

14. Što radi sljedeća funkcija:

```
void func (cvor *k, int *br) {  
    if (k != NULL) {  
        func(k->d, br);  
        if (k->l == NULL && k->r == NULL) (*br)++;  
        func(k->r, br);  
    }  
}
```

RJ: a) broji listove u stablu ←

- b) broji elemente u stablu
 - c) računa zbroj elemenata u stablu
 - d) broji parne elemente u stablu
 - e) ništa od navedenog
-

15. Koja funkcija vraća najveći element u stablu:

```
RJ: int func(cvor *k) {  
    if (k->l != NULL) return func(k->l);  
    else return k->element;
```

16. Koja od sljedećih tvrdnji je istinita?

- RJ: **a) inorder i preorder obilaskom bit će obrađeni svi čvorovi u stablu** ←
- b) obilazak ppreorder moguće je jedino primijeniti na lopotpunim stablima
 - c) postorder obilazak uvijek obrađuje samo listove stabla
 - d) preoder obilazak uvijek obrađuje samo listove stabla
 - e) inorder obilazak stabla obrađuje dvostruko više elemenata nego postorder obilazak
-

17. Uz prethodno ispravno deklarirane sve podatke i već formirano binarno stablo, što radi sljedeća funkcija:

```
void ispis (cvor *glava) {  
    if (glava != NULL) {  
        ispis (glava->lijево);  
        ispis (glava->десно);  
        printf ("%s \n", glava->element);  
    }  
}
```

RJ:

- a) uvijek ispisuje samo vrijednost elementa na koji pokazuje glava
 - b) funkcija ne radi ništa jer je tipa *void*
 - c) inorder ispisuje vrijednosti elemenata stabla
 - d) preorder ispisuje vrijednosti elemenata stabla
 - e) postorder ispisuje vrijednosti elemenata stabla** ←
-

18. Što radi sljedeća funkcija?

```
int f(cvor *glava) {  
    int i = 0;  
  
    if (glava) {  
  
        if (glava->lijevo || glava->desno) i++;  
  
        i += f(glava->lijevo);  
  
        i += f(glava->desno);  
  
    }  
  
    return i;  
}
```

RJ:

- a) Broji čvorove u stablu koji imaju lijevo dijete.
- b) Broji čvorove u stablu koji imaju oba dijeteta.
- c) Broji čvorove u stablu koji imaju desno dijete.
- d) Niša od navedenog.
- e) Broji čvorove u stablu koji nisu listovi (imaju bar jedno dijete).** ←

19. Što radi sljedeća funkcija:

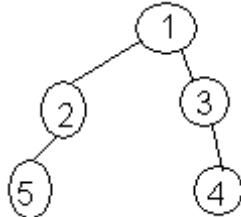
```
int fx (cvor *glava) {  
    if (glava) {  
  
        return fx(glava->l) + fx(glava->d) + 1;  
    }
```

```
    } else return 0;  
}
```

RJ:

- a) broji razine stabla
 - b) računa zbroj elemenata u stablu
 - c) vraća vrijednost ≥ 1 ako je stablo potpuno, 0 inače
 - d) broji listove stabla
 - e) broji čvorove stabla ←
-

20.



RJ: Stablo nije moguće ispisati sortirano inorder, preorder i postorder obilaskom

21. Pretraživanje binarnog stabla najbrže je ukoliko se radi o:

RJ: sortiranom potpunom stablu

22. Najefikasniji algoritam stvaranja gomile od n elemenata za najgori slučaj ima složenost:

RJ: a) $O(n \cdot \log_2 n)$

b) $O(\log_2 n)$

c) $O(1)$

d) $O(n^2)$

e) $O(n)$ ←

23. Koja je od sljedećih tvrdnji za gomilu točna?

RJ: Gomila se koristi kada je potrebno do najvećeg ili najmanjeg podatka doći algoritmom složenosti **O(1)**. Reorganizacija gomile nakon uklanjanja najvećeg ili najmanjeg podatka je složenosti **O(log₂n)**. Novi element u gomilu će se dodati algoritmom **O(log₂n)**.

24. Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 5, 10, 7, 3, 1, 90 algoritmom čija je složenost za najgori slučaj $O(n \log_2 n)$?

RJ:

a) 90

5 10

3 1 5

b) 90

10 7

5 1 3

c) 90

←

5 10

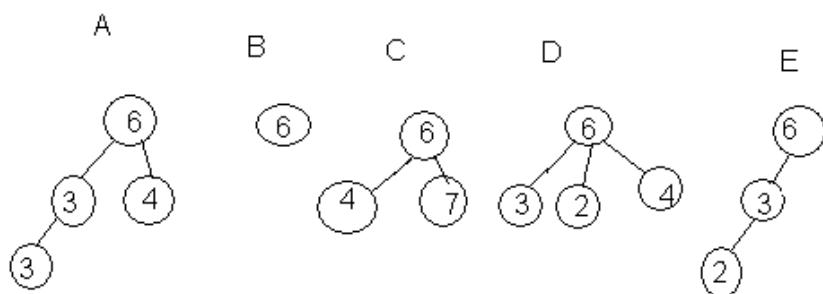
3 1 7

d) 90

10 7

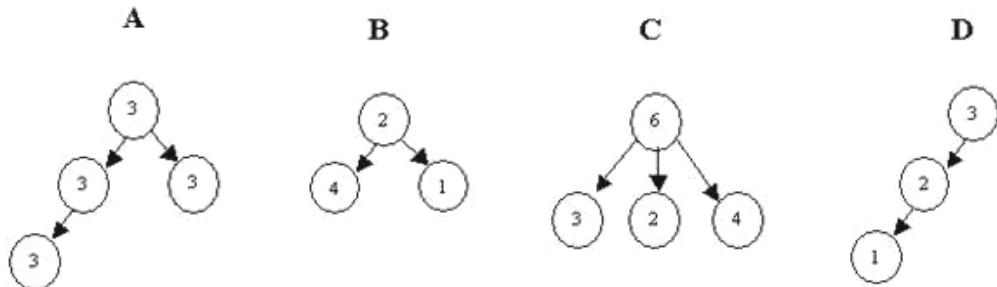
3 1 5

25. Koja od prikazanih stabla su gomile:



RJ: A, B

26. Koja od prikazanih stabala su gomile:

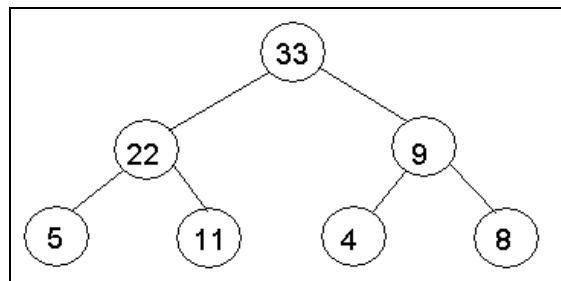


Rj:

- a) A ←
 - b) B
 - c) C
 - d) niti jedno od pokazanih stabala nije gomila
 - e) A, B, C, D
-

27. Koja slika prikazuje gomilu oblikovanu ulaznim nizom (11, 33, 4, 5, 22, 59) tako da za najgori slučaj složenost bude $O(n \log_2 n)$?

Rj:



28. Kada se gomila oblikuje dodavanjem jednog po jednog elementa u stablo uz očuvanje strukture gomile, tada je vrijeme izvođenja oblikovanja gomile za najgori slučaj (n je broj ulaznih elemenata):

RJ: a) $O(\log 2n)$

b) $O(n)$

c) $O(n^2 * \log 2n)$

d) **$O(n * \log 2n)$** ←

e) $O(n^2)$

29.

20 3 15 2 1 14 10

Ako je gomila realizirana u polju, u kojem od sljedećih elementi zapisani u polju:

RJ: 20

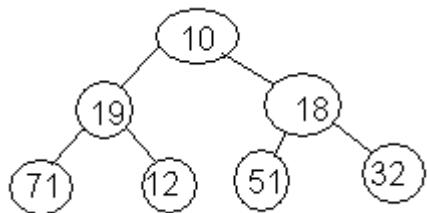
3 15

2 1 14 10

30. Stvori gomilu – složenost u najgorem slučaju poboljšane funkcije:

RJ: $O(n)$

31. Podaci za koje je potrebno stvoriti gomilu smješteni su u stablo na sljedeći način – koji će se prvi element zamijeniti (tu mi fali dio pitanja):



RJ: 51 i 18

STOG-01

Koja od sljedećih nizova naredbi u pseudokodu će zamijeniti vrijednost varijabli A i B pomoću stoga:

- a) stavi(A); stavi(B); skini(A); skini(B);
 - b) stavi(A); skini(B);
 - c) stavi(B); skini(A);
 - d) stavi(A); stavi(Pom); stavi(B); stavi(Pom); skini(A); stavi(Pom);skini(B);
 - e) stavi(A); skini(B); stavi(B); skini(A);
-

STOG-02

Na stog prikazan poljem pohranjuju se samo cijeli brojevi. Prototip funkcije za skidanje cijelog broja sa stoga je (funkcija vraća 0 ili 1, ovisno o tome da li se zapis uspio skinuti s vrha stoga):

- a) int skini(int stavka, int stog[], int *vrhStog);
- b) int skini(int *stavka, int stog[], int *vrhStog);**
- c) int skini(float stavka, float stog[], int vrhStog);
- d) int *skini(int *stavka, int stog[], int vrhStog);
- e) void *skini(int *stavka, int stog[], int n, int *vrhStog);

STOG-03

Složenost funkcije

```
int dodaj (zapis stavka, zapis stog[], int n, int *vrh) {  
    if (*vrh >= n-1) return 0;  
    (*vrh)++;  
    stog [*vrh] = stavka;  
    return 1;  
}
```

je:

- a) $O(n)$
- b) $O(\log n)$
- c) složenost ovisi o veličini zapisa stavke, pa se ne može jednoznačno odrediti
- d) $O(1)$**
- e) $O(\log_2 n)$

STOG-04

Ako je stog realiziran cjelobrojnim poljem od n elemenata, kolika je apriorna složenost skidanja SVIH elemenata sa stoga:

- a) $O(1)$
 - b) $O(n)$**
 - c) $O(n^2)$
 - d) $O(\log_2 n)$
 - e) ovisi o operacijskom sustavu
-

STOG-05

Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha a 0 u slučaju neuspjeha i ima prototip

```
int push (int element);
```

a funkcija skidanja sa stoga vraća vrijednost elementa s vrha ili -1 ako je stog prazan i ima prototip

```
int pop ();
```

što će biti na stogu nakon obavljanja sljedeće naredbe, uz prepostavku da je stog bio prazan i da stog raste s lijeva na desno:

```
push(push(pop()));
```

- a) 1 1
b) -1 1
c) 1
d) 0
e) stog će biti prazan
-

STOG-06

Funkcija za dodavanje elementa na stoga realiziran listom glasi:

a)

```
atom *dodaj (atom *vrh, int element) {  
    atom *novi;  
  
    if ((novi = (atom *) malloc(sizeof(atom))) != NULL) {  
  
        novi->element = element;  
  
        novi->sljed = vrh;  
  
    }  
  
    return novi;  
}
```

b)

```
atom *dodaj (atom *vrh, int element) {  
    atom *novi;  
  
    novi->element = element;  
  
    novi->sljed = vrh;  
  
    return novi;  
}
```

c)

```
int dodaj (atom *vrh, int element) {  
    atom *novi;  
  
    if ((novi = (atom *) malloc(sizeof(atom))) != NULL) {  
        novi->element = element;  
        novi->sljed = vrh;  
        return 1  
    }  
  
    else  
        return 0;  
}
```

d)

```
atom *dodaj (atom *vrh, int element) {  
    atom *novi;  
  
    if ((novi = (atom *) malloc(sizeof(atom))) != NULL) {  
        novi = element;  
    }  
  
    return novi;  
}
```

e)

```
int dodaj (atom *vrh, int element) {  
    atom *novi;  
  
    if ((novi = (atom *) malloc(sizeof(atom))) != NULL) {  
        novi = element;  
        return 1  
    }  
  
    else  
        return 0;  
}
```

RED-01

Ukoliko je ulaz = 1, a izlaz=4, koliko ima elemenata u redu realiziranom pomoću cirkularnog polja, ako je veličina polja 10 (prepostavite da ulaz pokazuje na prvi prazan element, dok izlaz pokazuje na prvi stavljeni element)?

- a) 7
 - b) 6
 - c) 3
 - d) 5
 - e) ne može se odrediti
-

RED-02

Neka je na sljedeći način napisana funkcija koja skida element tipa `tip` iz reda realiziranog cikličkim poljem:

```
int SkiniiIzReda (tip *element, tip red[], int n,
                    int *izlaz, int ulaz) {
    if (ulaz == *izlaz) return 0;
    (*izlaz)++;
    *izlaz %= n;
    *element = red[*izlaz];
    return 1;
}
```

Koja je od sljedećih tvrdnji **lažna**?

-
- a) Složenost funkcije je $O(1)$.
 - b) Funkcija vraća 0, ako se iz reda može skinuti točno jedan element.**
 - c) Za poziv funkcije, kada u redu postoji barem jedan zapis koji se može skinuti iz reda, funkcija vraća 1.
 - d) Funkcija vraća 1, ako je zapis uspješno skinut iz reda.
 - e) Za poziv funkcije, kada je red prazan, funkcija vraća 0.

RED-03

U red realiziran jednostruko povezanom listom pohranjuju se zapisi koji sadrže cijele brojeve. Prototip funkcije za skidanje zapisa iz tako realiziranog reda je (funkcija vraća 1 ili 0, ovisno o tome je li zapis uspješno skinut iz reda):

- a) int skini (cvor **ulaz, cvor **izlaz, int element);
 - b) int skini (cvor **ulaz, cvor **izlaz, int *element);**
 - c) void skini (cvor **ulaz, cvor **izlaz, int *element);
 - d) int skini (cvor *ulaz, cvor **izlaz, int element);
 - e) int skini (cvor *ulaz, cvor **izlaz, int *element);
-

PITANJA S FORUMA:

STOG:

-naći točnu tvrdnju-4 su jako glupe, točna je: Da bi se pristupilo elementu na dnu stoga, treba se maknuti sve sa vrha

-dva programčića sa stogom (treba samo znati kako rade funkcije dodaj i skini)

- Imamo funkcije int push(int elem) i int pull() koje rade sa stogom. Na stogu postoje već neki elementi (nije prazan). Funkcija push vraća 1 ako je uspjela, 0 inače, a funkcija pull vraća element kojeg skine sa stoga i ne vraća ništa za grešku.

Što će biti na stogu nakon naredbe push(push(push(5)) + pull());

Rješenje : 5 2

- Stog punimo sa for petljom i=1 do 10, onda ga praznimo i=1 do 5, sto je ostalo?

Rjesenje: 1,2,3,4,5

- Na stog prikazan poljem pohranjuju se samo cijeli brojevi. Prototip funkcije za skidanje cijelog broja sa stoga je:

-Ponuđene su cijele funkcije za stavljanje elementa na stog...odabratи točnu

-Stog, koji ima barem dva elementa, treba zamijeniti vrijednosti dva elementa s vrha

a) int a, b a= pop(), b=pop(), push(a), push(b)<---

- Stog ostvarem statičkim poljem od n elemenata može: primiti n elemenata

- Netočna tvrdnja: Prazan stog je greška u programu

- Funkcije *dodaj i skini* uvijek imaju istu složenost bez obzira na broj članova.

-Bio je jedan koji stavlja na stog od 0 do 10 kao, ali' veličina stoga je 5 i onda printf-a ono što popa...

Rješenje: 4 3 2 1 0

- Napisan neki, sa stogom i sad, MAXSTOG je 10, on stavlja elemente na stog i pitanje je koliko najviše može staviti? A caka je u tome što mu je vrh definiran kao 0 a ne -1 što znači da je gore već jedan element. Rj: Može se staviti 9

- Je li u funkciji *dodaj* red) potrebno primati pokazivač na izlaz iz reda po referenci i zašto?

RJ: Da, zato što u slučaju NULL vrijednosti glave podaci bivaju izbrisani (il neš u tom stilu)

- koja je složenost dohvatanja zadnjeg elementa reda (red je realiziran listom)?**RJ:** O(n)

- Koji je prototip funkcije za skidanje elemenata iz reda listom (funkcija vraća 1 ako je uspjela skinuti element, inače vraća 0)?
- U red jednostruko povezanom listom pohranjuju se cijelobrojni zapisi. Prototip f-je za skidanje iz reda (1 za uspjesno, 0 neuspjesno obavljeno)
- Koja je tvrdnja za jednostruko povezanu linearu listu je istinita: **RJ:** može ostvariti statičkom strukturom polje

Red ostvaren ciklickim poljem, koja je tvrdnja neistinita?

```
int SkinIzReda(tip *element, tip red[], int n, int *izlaz, int ulaz){
    if(ulaz==*izlaz) return 0;
    (*izlaz)++;
    *izlaz %=n;
    *element=red[*izlaz];
    return 1;
}
```

RJ: F-ja vraca 0, ako se iz reda moze skinuti točno 1 element

Zadatak:

skini, stavi

1 ako je uspješno obavljeno, 0 ako nije uspjelo

pretpostavka da na stogovima ima dovoljno mesta

Zadana je funkcija:

```
void prepis(int stog1[], int stog2[]) {  
    int element;  
    if skini(stog1, &element)  
        prepis(stog1, stog2)  
        stavi(stog2, &element)  
}
```

Rješenje je:

premještamo elemente sa stog1 na stog2, i redoslijed elemenata stog1 je ISTI! kao redoslijed elemenata stog2

zadatak:

Prototipi skini i dodaj su zadani

Što će ispisati?

Kod je isto tu bio zadan...

```
int a=1, b=2, c=3;  
  
dodaj a  
  
dodaj b  
  
dodaj c  
  
skini a  
  
skini c  
  
skini b  
  
// nisam sigurna da je takav redoslijed, ali slično je bilo ako ne isto  
printf("%d %d %d", a, b, c);
```

Rješenje je: 3 1 2

16. Stog iz starih blitzeva gdje je rješenje 3 1 2

Algoritmi i strukture podataka

3. blic – zadaci za vježbu skupljeni iz bliceva od prijašnjih godina ak. god. 2007/08

by majah

LISTE-01

Predložena je funkcija za pronađazak čvora sa zadanom cijelobrojnom šifrom u jednostruko povezanoj linearnej listi:

```
cvor *trazi(cvor *glava, int sifra){  
    if (glava->sifra == sifra) return glava;  
    if (glava->sljed)  
        return trazi(glava->sljed, sifra);  
    else  
        return NULL;  
}
```

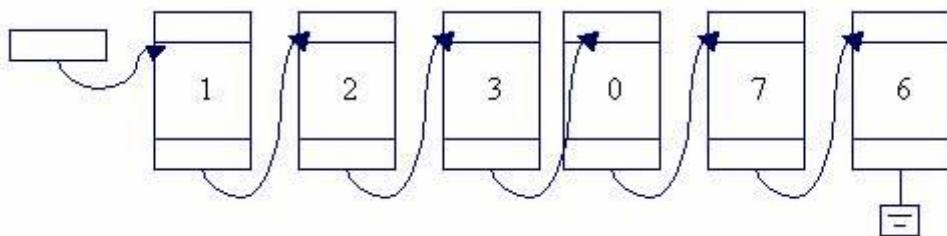
Koja od sljedećih tvrdnji (koje se odnose na predloženu funkciju) je ispravna?

- a) Funkcija je ispravna i vraća pokazivač na čvor sa zadanom šifrom ili NULL ako čvor sa zadanom šifrom ne postoji u listi.
- b) Funkcija nije ispravna: pretražuje sve čvorove liste osim zadnjeg!
- c) **Funkcija nije ispravna: ne radi za praznu listu (uzrokuje pogrešku kod poziva)!**
- d) Funkcija nije ispravna: ukoliko čvor sa zadanom šifrom ne postoji u listi, funkcija neće "nikad" završiti
- e) Funkcija nije ispravna: pretražuje sve čvorove liste osim prvog!

LISTE-02

Što će vratiti priložena funkcija za zadatu jednostruko povezani linearnu listu:

Slika (unutar čvorova prikazana je vrijednost varijable element):



```
int f(cvor *glava){  
    if (glava){  
        if (glava->element > 3)  
            return glava->element + f(glava->sljed);  
        else  
            return f(glava->sljed);  
    }else{  
        return 0;  
    }  
}
```

- a) 6
 - b) 19
 - c) 0
 - d) 13
 - e) 16
-

LISTE-03

U dvostruko povezani listu spremaju se zapisi slijedećeg tipa:

```
typedef struct s1{  
    int mbr;           // matični broj studenta  
    char ime[40+1];   // ime studenta  
    float prosjek;    // prosjek ocjena  
    struct s1 *sljed;  
    struct s1 *preth;  
} zapis;
```

Kako glasi funkcija koja izbacuje prvi element iz liste, oslobađa zauzetu memoriju te vraća 1 ako je operacija uspjela, a 0 ako operacija nije uspjela?

a)

```
int izbaci(zapis **glava, zapis **rep) {  
    if (*glava == NULL) return 0;  
    if ((*glava)->sljed == NULL) {  
        free(*glava);  
        *glava = *rep = NULL;  
    }  
    else {  
        *glava = (*glava)->sljed;  
        free((*glava)->preth);  
        (*glava)->preth = NULL;  
    }  
    return 1;  
}
```

b)

```
int izbaci(zapis **glava) {  
    if (*glava == NULL) return 0;  
  
    *glava = (*glava)->sljed;  
    free((*glava)->preth);  
    (*glava)->preth = NULL;  
    return 1;  
}
```

c)

```
int izbaci(zapis *glava, zapis *rep) {  
    if (glava == NULL) return 0;  
    if (glava->sljed == NULL) {  
        free(glava);  
        glava = rep = NULL;  
    }  
    else {  
        glava = glava->sljed;  
        free(glava->preth);  
        glava->preth = NULL;  
    }  
    return 1;  
}
```

d)

```

int izbaci(zapis **glava) {
    zapis *pom = *glava;
    if (*glava == NULL) return 0;
    glava = (glava)->sljed;
    free(pom);
    return 1;
}

e)

void izbaci(zapis **glava, zapis **rep) {
    if (*glava != NULL) {
        if ((*glava)->sljed == NULL) {
            free(*glava);
            *glava = *rep = NULL;
        }
        else {
            *glava = (*glava)->sljed;
            free((*glava)->preth);
            (*glava)->preth = NULL;
        }
    }
}

```

STABLA-01

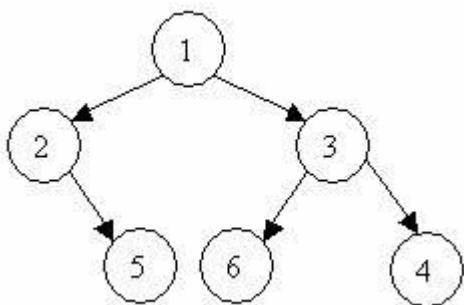
Što će se ispisati funkcijom:

```
void ispis(cvor *korijen){  
    if (korijen) {  
        ispis(korijen->desno);  
        ispis(korijen->lijevo);  
        printf("%c", korijen->element);  
    }  
}
```

za stablo na slici pozivom funkcije

ispis (korijen);

ako je korijen u trenutku poziva pokazivač na korijen stabla?



- a) 123456
- b) 146352
- c) 643521
- d) 321
- e) **463521**

Koliko razina ima potpuno binarno stablo koje sadrži 100 čvorova i koliki je broj čvorova na posljednjoj razini ?

- a) **broj razina=7 broj čvorova=37**
 - b) broj razina =6 broj čvorova=64
 - c) broj razina =7 broj čvorova=64
 - d) broj razina =7 broj čvorova=50
 - e) broj razina =6 broj čvorova=50
-

STABLA-03

Što radi slijedeća funkcija?

```
int f(cvor *glava) {  
    int i = 0;  
    if (glava) {  
        if (glava->lijevo || glava->desno) i++;  
        i += f(glava->lijevo);  
        i += f(glava->desno);  
    }  
    return i;  
}
```

- a) Broji čvorove u stablu koji imaju oba dijeteta.
 - b) Broji čvorove u stablu koji imaju lijevo dijete.
 - c) Broji čvorove u stablu koji imaju desno dijete.
 - d) Broji čvorove u stablu koji nisu listovi (imaju bar jedno dijete).**
 - e) Ništa od navedenog.
-

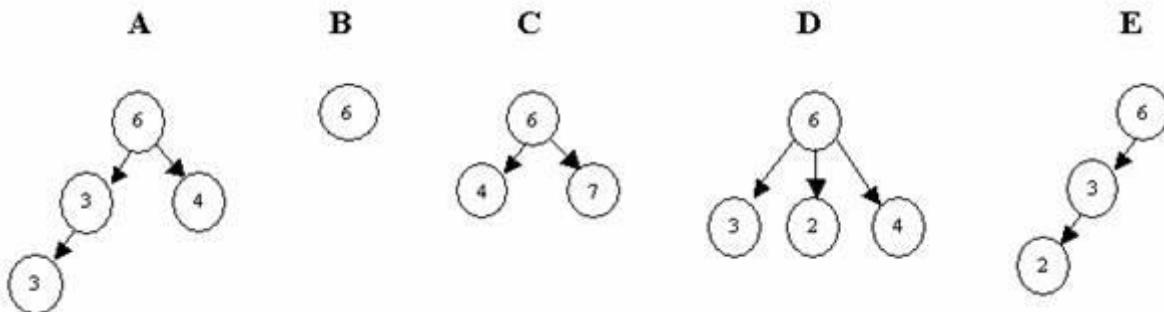
GOMILA-01

Najefikasniji algoritam stvaranja gomile od n elemenata za najgori slučaj ima složenost

- a) $O(1)$
 - b) $O(n)$**
 - c) $O(n \cdot \log_2 n)$
 - d) $O(n^2)$
 - e) $O(\log_2 n)$
-

GOMILA-02

Koja od prikazanih stabala su gomile:



- a) A, B**
 - b) B
 - c) A, B, E
 - d) A, B, C, E
 - e) A
-

GOMILA-03

Neka se u gomili koja je pohranjena u polje nalaze sljedeći podaci:

66	57	32	30	36	
----	----	----	----	----	--

Što će se desiti kada se u takvu gomilu doda podatak 35 (a nakon što se pozove funkcija *ubaci* koja dodaje elemente u gomilu)?

- a) Izbac se broj 66, a 35 se umetne na to mjesto
- b) Taj će se broj dodat na kraj polja jer je time očuvano svojstvo gomile
- c) **Broj 35 će zamijeniti mjesto s brojem 32**
- d) Broj 35 će zamijeniti mjesto s brojem 36
- e) Broj 35 će zamijeniti mjesto s brojem 30

OOP

OO-01

Zadan je program u jeziku C++:

```
class MojRazred {  
    int _MojPodatak;  
};  
  
int main () {  
    MojRazred obj;  
    obj._MojPodatak = 10;  
    return 0;  
}
```

Koja od sljedećih tvrdnji je točna?

- a) Program se može prevesti, povezati i izvoditi bez greške.
 - b) Program se prevodi uz upozorenje (warning), no ipak se može povezati i izvoditi bez greške.
 - c) Program se može prevesti i povezati. Kod izvođenja, naredba za pridruživanje se ignorira, no to ne izaziva grešku.
 - d) Program se može prevesti i povezati. Naredba za pridruživanje se izvršava, no nakon nje se «baca» iznimka.
 - e) **Program se ne može prevesti jer prevodilac javlja grešku.**
-

OO-02

Zadan je razred u jeziku C++:

```
class MojRazred {  
    int _MojPodatak;  
};
```

U glavnoj funkciji deklariran je objekt:

```
MojRazred obj;
```

Kako izgleda naredba kojom glavna funkcije članskoj varijabli _MojPodatak unutar objekta obj pridružuje vrijednost 20?

- a) obj._MojPodatak = 20;
 - b) obj ->_MojPodatak = 20;
 - c) MojRazred *pobj = &obj; (*pobj)._MojPodatak = 20;
 - d) Takva naredba postoji, no nije ni jedna od gore navedenih.
 - e) **Takva naredba ne postoji.**
-

OO-03

Promatramo dvije deklaracije u jeziku C++:

```
struct MyStruct {           class MyClass {  
    int _Mydata;           int _MyData;  
};                      };
```

Koja od sljedećih tvrdnji je točna?

- a) Dvije deklaracije su ekvivalentne u smislu da odgovarajući objekti imaju ista svojstva.
 - b) Članska varijabla u MyStruct je javna a članska varijabla u MyClass je privatna.**
 - c) Prva deklaracija nije dozvoljena jer C++ ne poznaje struct.
 - d) Druga deklaracija nije dozvoljena jer razred mora imati barem jednu člansku funkciju.
 - e) Obje deklaracije su sintaktički neispravne, zato jer su im članske variable privatne, a ne postoje članske funkcije koje bi mijenjale te varijable.
-

OO-04

Promatramo sljedeći C++ program:

```
#include <stdio.h>  
  
class MojRazred {  
public:  
    MojRazred( );  
    int * _Podaci;  
};  
  
MojRazred :: MojRazred( ) {  
    _Podaci = new int [10];  
    for ( int i=0; i<10; i++ )
```

```
_Podaci[i] = 0;  
}  
  
void Promijeni (MojRazred obj) {  
    obj._Podaci[0] = 1;  
}  
  
int main ( ) {  
    MojRazred ob;  
    Promijeni(ob);  
    printf("%d \n", ob._Podaci[0]);  
    return 0;  
}
```

Odaberite točnu tvrdnju:

- a) Program ispisuje 0 .
 - b) Program ispisuje 1.**
 - c) Program ispisuje cjelobrojnu vrijednost koja nije definirana.
 - d) Program se ne može prevesti zato jer u razredu MojRazred nije definiran copy konstruktor.
 - e) Program neće ništa ispisati zato jer će doživjeti grešku tijekom izvođenja pri pokušaju mijenjanja članske varijable unutar objekta.
-

OO-05

U C++ programu nalazi se sljedeća deklaracija:

```
class MyClass {  
public:  
    MyFunc( );  
private:  
    int _MyData;  
};
```

U glavnoj funkciji stvoren je objekt iz razreda **MyClass** na sljedeći način:

```
MyClass *p = new MyClass( );
```

Koja od sljedećih naredbi u nastavku glavne funkcije predstavlja ispravan poziv članske funkcije **MyFunc()** ?

- a) **p-> MyFunc();**
 - b) p.MyFunc();
 - c) MyFunc(*p);
 - d) MyFunc(p);
 - e) (*p)->MyFunc();
-

OO-06

Zadan je sljedeći C++ program:

```
#include <stdio.h>
```

```
class MyClass {  
public:  
    MyClass( );  
    MyClass(int a);  
    ~MyClass( );  
  
private:  
    int _data;  
};  
  
MyClass :: MyClass ( ) {  
    _data = 0;  
    printf("%d ", _data);  
}  
  
MyClass :: MyClass (int a ) {  
    _data = a;  
    printf("%d ", _data);  
}  
  
MyClass :: ~MyClass ( ) {  
    printf("destroy %d ", _data);  
}  
  
int main ( ) {  
    MyClass x;  
    MyClass *y = new MyClass(5);  
    delete y;  
}
```

Što će ispisati ovaj program?

- a) **0 5 destroy 5 destroy 0**
- b) 0 5 destroy 0 destroy 5
- c) 0 5 destroy 5
- d) 5 destroy 5
- e) 0 0 destroy 0 destroy 0

1. Razlika između deep/shallow kopiranja objekta...

6. Koliko puta se zovu konstruktor i destruktur objekta koji je stvoren jednom u main()-u naredbom *objekt ime;*

i onda opet u funkciji koja se poziva iz main()-a a koja sadržava naredbu:

*objekt *ime=new objekt;*

Konstruktor se zove dva puta, a destruktur jednom, jer se ne koristi naredba delete za drugi objekt (prvi se pobriše automatski kad se napusti main())

10. Imamo klasu student koja ima konstruktor koji prima integer. Taj integer se pohrani u člansku varijablu _X i nakon toga se ispise slovo "A" _X puta. U destrukturu se ispise slovo "B" isto _X puta.

Što će ispisati program:

```
main() { ...  
student novi(2);  
student *novi2=new student(3);  
... }
```

Rješenje: Ispisat će 5 slova "A" i 2 slova "B" jer se ne poziva destruktur za ovaj drugi objekt (ne koristi se delete)

11. Imamo klasu koja nema destruktur, a u konstruktoru se stvara polje (članska varijabla) pomoću naredbe **new**. Ispravna tvrdnja o toj klasi je **da se prostor koji zauzima polje neće oslobođiti nakon brisanja objekta.**

14. Ako klasa "K" ima metodu "metoda" koja nije statična i ako imamo naredbu *K novi;*

kako pozivamo metodu "metoda"?

Rješenje: novi.metoda();

15. Kako se inicijalizira referenca?

Rješenje: int a; &ref=a;

17. Klasa trokut u kojoj su definirana dva konstruktora (s parametrom i bez-ovaj ima a=1 b=1 c=1) i funkcija *opseg*. U main()-u se definira *trokut t* i poziva se funkcija *opseg=t.opseg();* uglavnom, **to poziva default konstruktor i opseg je 3**

18. Definiran je neki copy konstruktor za kompleksne brojeve i pita se sto radi this->re...

treba kliknuti na ono gdje piše da to ovisi o tome iz otkud je pozvan i za koje parametre...

34. Imamo destruktor ~Stog(); u nekoj definiranoj klasi Stog. Ako u glavnom programu instanciramo objekt na način Stog obj; kako će se taj objekt uništiti ili tako nešto!

Rješenje: AUTOMATSKI će se destruktor pozvati kada se izade iz glavne funkcije i na taj način će se uništiti objekt.

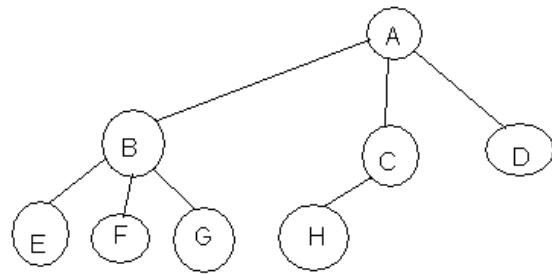
36. Zadan je kod:

```
int a;  
int &referanca = a;  
a=1;  
  
printf("%d %d ", a, referanca);  
referanca=2;  
printf("%d %d", referanca, a);
```

Što će se ispisati?

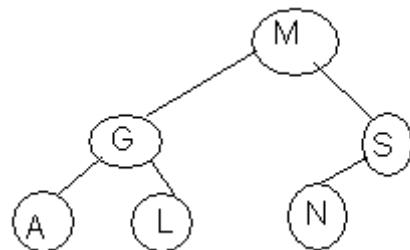
Rješenje: 1 1 2 2

1. Stablo ima sljedeće atribute:



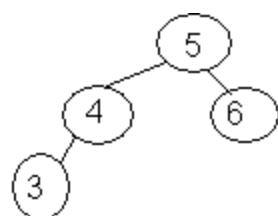
RJ: stupanj = 3, dubina = 3, potpuno

2. Zadana je funkcija (ne sjećam se koja), što se ispisuje?



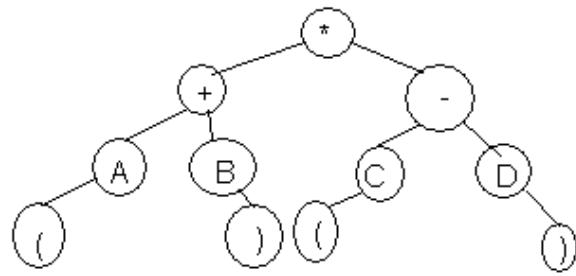
RJ: G M

3. Zadano je 5, 4, 6, 3 i napravi se stablo i ima funkcija i treba znat što ispisuje:



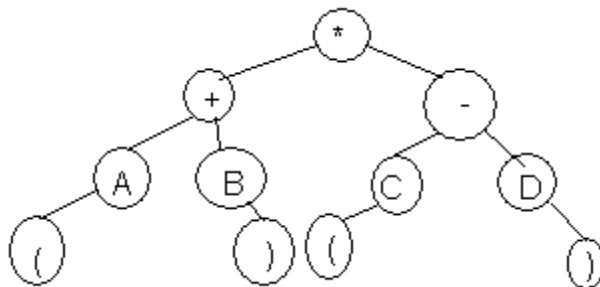
RJ: 3 4 5 6

4. Inorder obilazak stabla na slici daje izraz:



RJ: $(A + B) * (C - D)$

5. Postorder obilazak stabla na slici daje izraz:



RJ: $(A) B + (C) D - *$

6. Koliko razina ima potpuno binarno stablo koje sadrži 100 čvorova i koliki je broj čvorova na posljednjoj razini ?

a) broj razina =6 broj čvorova=64

b) broj razina=7 broj čvorova=37 ←

c) broj razina =7 broj čvorova=50

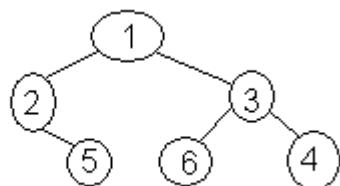
d) broj razina =7 broj čvorova=64

e) broj razina =6 broj čvorova=50

7. Što će se ispisati funkcijom:

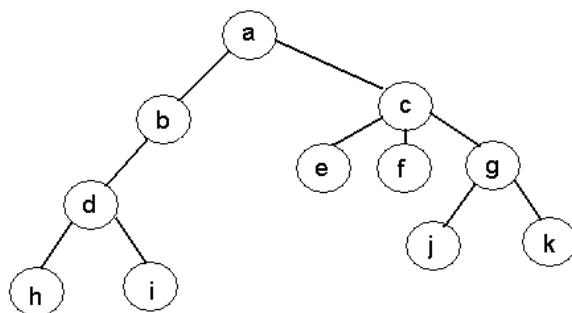
```
void ispis (cvor *korijen) {  
    printf ("%c", korijen->element);  
    if (korijen->lijevo && korijen->desno) {  
        ispis (korijen->desno);  
        ispis (korijen->lijevo);  
    }  
}
```

za stablo na slici pozivom funkcije?



RJ: 1 3 4 6 2

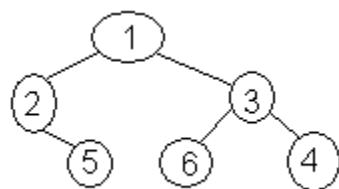
8. Koja od sljedećih tvrdnji nije istinita:



RJ:

- a) Svi čvorovi sa stabla na slici su istog stupnja ←
 - b) U stablu sa slike listovi su:{h,i,e,f,j,k}
 - c) Maksimalni broj čvorova binarnog stabla na k-toj razini jednak je $2^k - 1$
 - d) Maksimalni broj čvorova binarnog stabla dubine k jednak je $2^k - 1$ za $k > 0$
 - e) Binarno stablo koje je visine k i ima $2^k - 1$ elemenata naziva se puno (full) binarno stablo
-

9. Zadana je neka funkcija (mislim postorder) i treba znati što ispisuje:



RJ: 4 6 3 5 2 1

10. Stupanj stabla (koji ima n razina) je:

RJ:

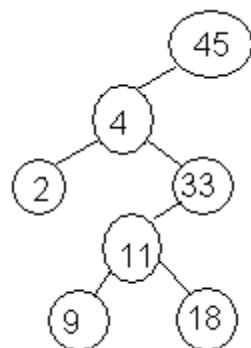
- a) najmanji stupanj nekog čvora u stablu
 - b) n
 - c) najveći stupanj nekog čvora u stablu ←
 - d) broj čvorova u stablu
 - e) broj čvorova u potpunom stablu s n razinama
-

11. Koliko čvorova ima koso stablo sa n razina:

RJ:

- a) 2^*n-1
 - b) $n+1$
 - c) 2^n-1
 - d) $n \leftarrow$**
 - e) 2^n
-

12.



Sortirano binarno stablo na slici (lijevo manji element, desno veći) generirano je sljedećim ulaznim nizom brojeva:

RJ: 45, 4, 33, 11, 2, 9, 18

13. U prazno binarno stablo uneseni su elementi 20, 15, 1, 3, 7, 48, 12, 19, 35. Kolika je dubina stabla?

RJ: 6

14. Što radi sljedeća funkcija:

```
void func (cvor *k, int *br) {  
    if (k != NULL) {  
        func(k->d, br);  
        if (k->l == NULL && k-> d == NULL) (*br)++;  
        func(k->d, br);  
    }  
}
```

RJ: a) broji listove u stablu ←

- b) broji elemente u stablu
 - c) računa zbroj elemenata u stablu
 - d) broji parne elemente u stablu
 - e) ništa od navedenog
-

15. Koja funkcija vraća najveći element u stablu:

```
RJ: int func(cvor *k) {  
    if (k->l != NULL) return func(k->l);  
    else return k->d;  
}
```

```
else return k->element;
```

16. Koja od sljedećih tvrdnji je istinita?

RJ: a) inorder i preorder obilaskom bit će obrađeni svi čvorovi u stablu ←

- b) obilazak ppreorder moguće je jedino primijeniti na Ipotpunim stablima
- c) postorder obilazak uvijek obrađuje samo listove stabla
- d) preoder obilazak uvijek obrađuje samo listove stabla
- e) inorder obilazak stabla obrađuje dvostruko više elemenata nego postorder obilazak

17. Uz prethodno ispravno deklarirane sve podatke i već formirano binarno stablo, što radi sljedeća funkcija:

```
void ispis (cvor *glava) {  
    if (glava != NULL) {  
        ispis (glava->lijево);  
        ispis (glava->десно);  
        printf ("%s \n", glava->element);  
    }  
}
```

RJ:

- a) uvijek ispisuje samo vrijednost elementa na koji pokazuje glava
- b) funkcija ne radi ništa jer je tipa *void*
- c) inorder ispisuje vrijednosti elemenata stabla
- d) preorder ispisuje vrijednosti elemenata stabla
- e) postorder ispisuje vrijednosti elemenata stabla ←

18. Što radi sljedeća funkcija?

```
int f(cvor *glava) {  
    int i = 0;  
  
    if (glava) {  
        if (glava->lijevo || glava->desno) i++;  
        i += f(glava->lijevo);  
        i += f(glava->desno);  
    }  
  
    return i;  
}
```

RJ:

- a) Broji čvorove u stablu koji imaju lijevo dijete.
- b) Broji čvorove u stablu koji imaju oba djeteta.
- c) Broji čvorove u stablu koji imaju desno dijete.
- d) Niša od navedenog.
- e) Broji čvorove u stablu koji nisu listovi (imaju bar jedno dijete). ←**

19. Što radi sljedeća funkcija:

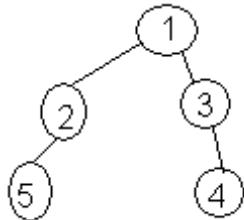
```
int fx (cvor *glava) {  
    if (glava) {
```

```
    return fx(glava->l) + fx(glava->d) + 1;  
} else return 0;  
}
```

RJ:

- a) broji razine stabla
 - b) računa zbroj elemenata u stablu
 - c) vraća vrijednost ≥ 1 ako je stablo potpuno, 0 inače
 - d) broji listove stabla
 - e) broji čvorove stabla ←
-

20.



RJ: Stablo nije moguće ispisati sortirano inorder, preorder i postorder obilaskom

21. Pretraživanje binarnog stabla najbrže je ukoliko se radi o:

RJ: sortiranom potpunom stablu

22. Najefikasniji algoritam stvaranja gomile od n elemenata za najgori slučaj ima složenost:

RJ: a) $O(n \cdot \log_2 n)$

b) $O(\log_2 n)$

c) $O(1)$

d) $O(n^2)$

e) **$O(n)$** ←

23. Koja je od sljedećih tvrdnji za gomilu točna?

RJ: Gomila se koristi kada je potrebno do najvećeg ili najmanjeg podatka doći algoritmom složenosti **$O(1)$** . Reorganizacija gomile nakon uklanjanja najvećeg ili najmanjeg podatka je složenosti **$O(\log_2 n)$** . Novi element u gomilu će se dodati algoritmom **$O(\log_2 n)$** .

24. Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 5, 10, 7, 3, 1, 90 algoritmom čija je složenost za najgori slučaj $O(n \log_2 n)$?

RJ:

a) 90

5 10

3 1 5

b) 90

10 7

5 1 3

c) 90

←

5 10

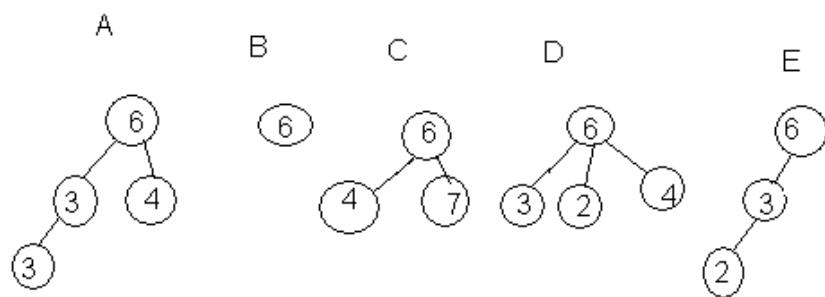
3 1 7

d) 90

10 7

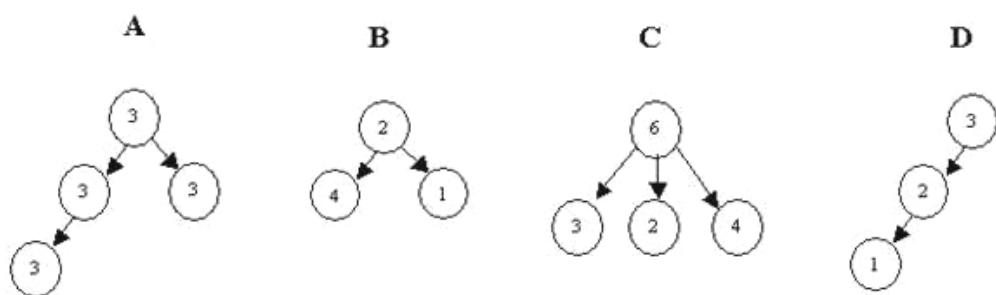
3 1 5

25. Koja od prikazanih stabla su gomile:



RJ: A, B

26. Koja od prikazanih stabala su gomile:



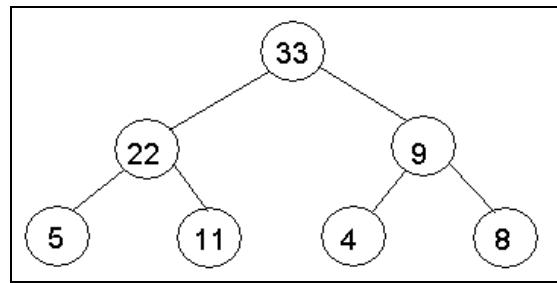
RJ:

- a) A ←
- b) B
- c) C
- d) niti jedno od pokazanih stabala nije gomila

e) A, B, C, D

27. Koja slika prikazuje gomilu oblikovanu ulaznim nizom (11, 33, 4, 5, 22, 59) tako da za najgori slučaj složenost bude $O(n \log_2 n)$?

Rj:



28. Kada se gomila oblikuje dodavanjem jednog po jednog elementa u stablo uz očuvanje strukture gomile, tada je vrijeme izvođenja oblikovanja gomile za najgori slučaj (n je broj ulaznih elemenata):

- RJ: a) $O(\log_2 n)$
b) $O(n)$
c) $O(n^2 * \log_2 n)$
d) $O(n * \log_2 n)$ ←
e) $O(n^2)$
-

29.

20 3 15 2 1 14 10

Ako je gomila realizirana u polju, u kojem od sljedećih elementi zapisani u polju:

RJ: 20

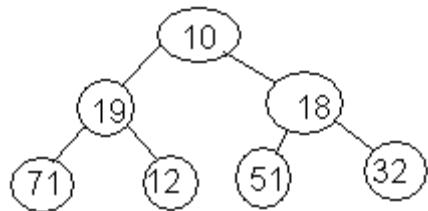
3 15

2 1 14 10

30. Stvori gomilu – složenost u najgorem slučaju poboljšane funkcije:

RJ: $O(n)$

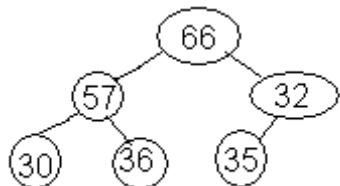
31. Podaci za koje je potrebno stvoriti gomilu smješteni su u stablo na sljedeći način – koji će se prvi element zamijeniti (tu mi fali dio pitanja):



RJ: 51 i 18

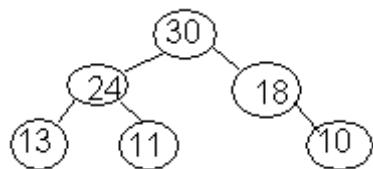
32. 66 57 32 30 36 - zadano polje, ako se doda 35 na kraj sa kime će zamijeniti mjesto (tako nekako ide pitanje):

RJ: Broj 35 će zamijeniti mjesto s brojem 32



33. Koji od sljedećih ne zadovoljava gomilu:

RJ: 10 je na krivoj strani



34. Koje ne zadovoljava svojstvo gomile (ne sjećam se detalja):

RJ: 10 7 8 9 6 5 4 3 2 1

35. 22 33 44 99 66 77 88 55 – zadano je polje i koji element će se zamijeniti (ne sjećam se detalja):

RJ: 44 i 88

36.

RJ: Gomila je potpuno binarno stablo takvo da je podatak u nekom čvoru veći ili jednak podacima u čvorovima svoje djece.

37.

99 88 66 22 77 55 33 11 – zadana je gomila (poljem), nakon prvog podešavanja pri uzlaznom heap sortu što se dogodi (ne sjećam se točno pitanja):

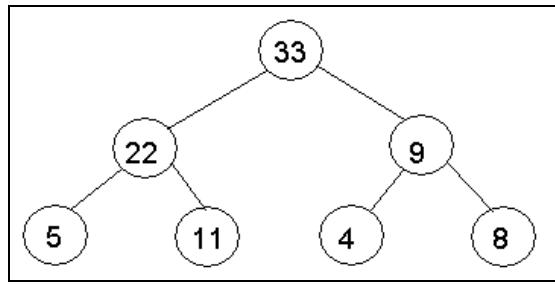
RJ: 88 77 66 22 11 55 33 99

38. Struktura stog se dinamički najčešće predstavlja?

Rj: Jednostruko povezanom linearnom listom

39. Koja slika prikazuje gomilu oblikovanu ulaznim nizom (11, 33, 4, 5, 22, 59) tako da za najgori slučaj složenost bude $O(n \log_2 n)$?

Rj:



40. Pretraživanje binarnog stabla najbrže je ako se radi o:

Rj: Sortiranom potpunom stablu

41. Što radi:

```
void ispisi(cvor *glava) {  
    cvor *p;  
    for (p=glava; p!=NULL;p=p->slijed)  
        printf("%d\n", p->element);  
}
```

Rj: Ispisuje sve vrijednosti elemenata jednostruko povezane linearne liste.

42. U prazno binarno stablo uneseni su elementi 20, 15, 1, 3, 7, 48, 12, 19, 35. Kolika je dubina stabla?

Rj: 6

43. Koja je od slijedećih tvrdnji za gomilu točna?

Rj: Gomila se koristi kada je do najvećeg/najmanjeg potrebno doći sa složenošću $O(1)$. Složenost reorganizacije nakon uklanjanja prvog člana je $O(\log_2 n)$. Složenost dodavanja novog člana u gomilu je $O(\log_2 n)$.

44. Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 50 5 7 10 13 1 8 algoritmom čija je složenost za najgori slučaj $O(n \log 2n)$?

a) 50

13 10

8 7 5 1

b) 50

5 7

10 13 1 8

c) 50

13 7

5 10 1 8

d) 50 ←

13 8

5 10 1 7

e) 50

13 8

10 7 5 1

45. Što ispisuje funkcija

```
void ispisi( struct cvor *glava ) {  
    if( glava != NULL && glava->elem % 2) {  
        printf(" %d ", glava->elem);  
        ispisi(glava->sljed);  
    }  
}
```

```
    }  
}  
}
```

ako se u jednostruko povezanoj listi na koju pokazuje parametar glava nalaze sljedeći cijeli brojevi :

1 57 43 13 8 11 20 10 56 53

- a) **1 57 43 13 ←**
 - b) ne ispisuje ništa
 - c) 1 57 43 13 8 11
 - d) 1 57 43 13 8 11 20 10 56 53
 - e) 1 57 43 13 11 53
-

46. Ispravna deklaracija dvostruko povezane liste u memoriji glasi:

a) struct s1 {
 int mbr;
 char ime_pr[50];
 int spol;
 long pret;
 long sljed;
}

b) struct s1 {
 int mbr;
 char ime_pr[50];

```
    int spol;
    zapis *pret;
    zapis *sljed;
}
```

c) struct s1 {

```
    int mbr;
    char ime_pr[50];
    int spol;
    struct s1 *sljed;
}
```

d) **typedef struct s1{** ←

```
    int mbr;
    char ime_pr[50];
    int spol;
} zapis1;
```

```
typedef struct s2{
    zapis1 element;
    struct s2 *pred;
    struct s2 *sljed;
} zapis;
```

e) struct s1 {

```
    int mbr;
    char ime_pr[50];
    int spol;
    long *pret;
```

```
    long *sljed;  
}
```

47. Što će se ispisati funkcijom:

```
void ispis (cvor *korijen) {  
    printf ("%c", korijen->element);  
    if (korijen->lijevo && korijen->desno) {  
        ispis (korijen->desno);  
        ispis (korijen->lijevo);  
    }  
}
```

za stablo na slici pozivom funkcije



ispis (korijen);

ako je korijen u trenutku poziva pokazivač na korijen stabla?

- a) 134625
 - b) 123456
 - c) 521634
 - d) 13462 ←**
 - e) 12364
-

48. Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 5, 10, 7, 3, 1, 90 algoritmom čija je složenost za najgori slučaj $O(n)$?

a) 90

7 10

3 1 5

b) 90 ←

10 7

3 1 5

c) 90

5 10

3 1 5

d) 90

10 7

5 1 3

e) 5

10 7

3 1 5

49. Koliko čvorova ima *koso stablo* s n razina?

a) $2*n-1$

b) $n+1$

c) 2^n-1

d) $n \leftarrow$

e) 2^n

50. Što će ispisati funkcija:

```
void ispis (cvor *korijen) {  
    printf ("%c", korijen->element);  
    if (korijen->lijevo && korijen->desno) {  
        ispis (korijen->desno);  
        ispis (korijen->lijevo);  
    }  
}
```

za stablo

```
 1  
 2     3  
 5     6     4
```

RJ: 1 3 4 6 2

51. Kada se gomila oblikuje dodavanjem jednog po jednog elementa u stablo uz očuvanje strukture gomile, tada je vrijeme izvođenja oblikovanja gomile za najgori slučaj (n je broj ulaznih elemenata):

- a) $O(\log 2n)$
 - b) $O(n)$
 - c) $O(n^2 \cdot \log 2n)$
 - d) $O(n \cdot \log 2n)$ ←**
 - e) $O(n^2)$
-

52. Od elemenata {20, 15, 1, 3, 7, 48, 12, 19, 35} formirano je sortirano binarno stablo. Kolika je dubina stabla?

RJ: 6

53. Koja procedura pronalazi zadani element u jednostruko povezanoj listi?

a)

```
cvor *trazi1(cvor *glava, tip element) {
    cvor *p;
    for (p=glava; p!=NULL; p++)
        if (p->element==element)
            return p;
    return NULL;
}
```

b)

```
cvor *trazi1(cvor *glava,tip element) {  
    cvor *p;  
    if (p->element==element) return p;  
    return NULL;  
}
```

c)

```
cvor *trazi1(cvor *glava,tip element) {  
    cvor *p;  
    for (p=glava;p!=NULL;p=p->sljed)  
        if (p->element!=element)  
            return p;  
    return NULL;  
}
```

d)

```
cvor *trazi1(cvor *glava,tip element){ ←  
    cvor *p;  
    for (p=glava;p!=NULL;p=p->sljed)  
        if (p->element==element)  
            return p;  
    return NULL;  
}
```

e)

```
cvor *trazi1(cvor *glava,tip element){  
    cvor *p;
```

```
for (p=glava;p!=NULL;p++)  
    if (p->element!=element)  
        return p;  
  
    return NULL;  
}
```

54. Koji je stupanj stabla s **n** razina?

- a) najmanji stupanj nekog čvora u stablu
 - b) n
 - c) **najveći stupanj nekog čvora u stablu** ←
 - d) broj čvorova u stablu
 - e) broj čvorova u potpunom stablu s **n** razina
-

55. Što radi zadana funkcija za poziv `f(glava, 7, &br)` ako je glava pokazivač na početak jednostruko povezane liste?

```
cvor *f(cvor *p,int broj,int *br) {  
    if (p) {  
        ++(*br);  
        if (p->broj==broj)  
            return p;  
        else  
            return f(p->slijed,broj,br);  
    } else return NULL;
```

}

RJ: vraća pokazivač na čvor koji sadrži broj i njegov redni broj u listi (preko br)

57. Dinamička struktura za jednostruko povezanu listu sadrži:

RJ: Pokazivac na prvi element liste i proizvoljan broj čvorova

58. Ovdje je napisana funkcija za dodavanje elemenata u stog ostvaren jednostruko povezanim listom. Na kraju funkcije je izostavljena jedna naredba. Treba izabrati naredbu koja ide na to mjesto da funkcija radi.

Funkcija otprilike izgleda ovako:

```
dodajelement(element **vrh,int vrijednost) {  
    element novi = new element;  
    novi->vrijednost=vrijednost;  
    novi->slijed=*vrh;  
    ##### <- Što nedostaje?  
}
```

RJ: Nedostaje naredba *vrh=novi;

61. Sto će se dogoditi nakon što u ovu gomilu ubacimo broj 35 i pozovemo funkciju "ubaci"?

66 57 32 30 36 ← ubacujemo broj 35

RJ: Zamijeniti će se 35 i 32

62. Koja procedura trazi element u listi?

RJ:

```
cvor *trazi (cvor *g, tip elem){  
    cvor *p;  
    for (p=g;p!=NULL;p=p->slijed)  
        if (p->vrijednost==elem) return p;  
    return NULL;  
}
```

65. Zadano je:

1
2 3
5 9 4

Što će se ispisati s:

```
void ispis (cvor *korijen){  
    if korijen{  
        ispis (korijen->desno);  
        ispis (korijen->lijevo);
```

```
    printf("%c", korijen->element);  
}  
}
```

RJ: 4 9 3 5 2 1

RJ: Da, zato što u slučaju NULL vrijednosti glave podaci bivaju izbrisani (il neš u tom stilu)

67. Dinamička podatkovna struktura za real. jednostruko povezane liste sastoji se od:

RJ: pokazivača na prvi element liste i proizvoljnog broja povezanih čvorova

71. Koja tvrdnja nije ispravna za zadalu funkciju?

```
cvor *trazi(cvor *glava, int sifra) {  
    if (glava->sifra==sifra) return glava;  
    if (glava->slijed)  
        return trazi(glava->slijed, sifra);  
    else  
        return NULL;  
}
```

RJ: F-ja nije ispravna, ne radi za praznu listu.

72. Sto ce se dogoditi pozivom `f(glava), f(glava)` f-je:

```
void f(cvor *glava) {  
    if (glava->sljed) {  
        printf ("%s\n", glava->ime);  
        f(glava->sljed);  
    }  
}
```

RJ: Dva puta ispisuje imena osim od posljednjeg cvora.

75. Koja je tvrdnja za jednostruko povezanu linearnu listu je istinita:

RJ: može ostvariti statičkom strukturu polje

76. Koja od ponuđenih funkcija ispravno implementira Heap sort?

a) void HeapSort(tip A[], int n) {
 int i;
 StvoriGomilu(A, n/2);
 for (i=n/2; i>=0; i--) {
 Zamijeni (&A[1], &A[i]);
 Podesi (A, 1, i-1);
 }
}

```

b) void HeapSort(tip A[],int n) {           ←
    int i;
    StvoriGomilu(A,n);
    for (i=n;i>=2;i--) {
        Zamijeni (&A[1],&A[i]);
        Podesi (A,1,i-1);
    }
}

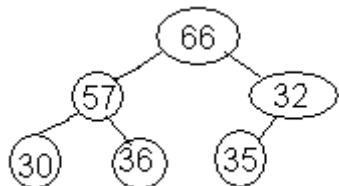
c) void HeapSort(tip A[],int n) {
    int i;
    StvoriGomilu(A,1);
    for (i=1;i<=n/2;i++) {
        Zamijeni (&A[n],&A[1]);
        Podesi (A,1,i+1);
    }
}

d) void HeapSort(tip A[],int n) {
    int i;
    StvoriGomilu(A,1);
    for (i=1;i<=n;i++) {
        Zamijeni (&A[n],&A[1]);
        Podesi (A,1,i+1);
    }
}

```

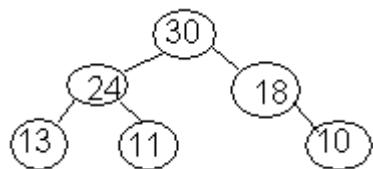
32. 66 57 32 30 36 - zadano polje, ako se doda 35 na kraj sa kime će zamijeniti mjesto (tako nekako ide pitanje):

RJ: Broj 35 će zamijeniti mjesto s brojem 32



33. Koji od sljedećih ne zadovoljava gomilu:

RJ: 10 je na krivoj strani



34. Koje ne zadovoljava svojstvo gomile (ne sjećam se detalja):

RJ: 10 7 8 9 6 5 4 3 2 1

35. 22 33 44 99 66 77 88 55 – zadano je polje i koji element će se zamijeniti (ne sjećam se detalja):

RJ: 44 i 88

36.

RJ: Gomila je potpuno binarno stablo takvo da je podatak u nekom čvoru veći ili jednak podacima u čvorovima svoje djece.

37.

99 88 66 22 77 55 33 11 – zadana je gomila (poljem), nakon prvog podešavanja pri uzlaznom heap sortu što se dogodi (ne sjećam se točno pitanja):

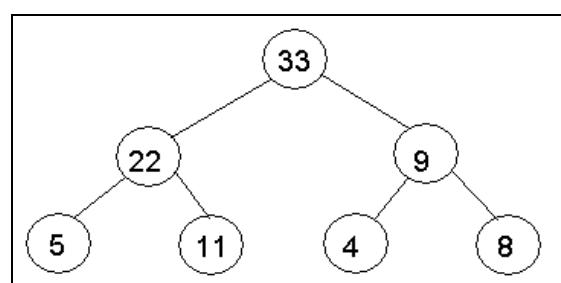
RJ: 88 77 66 22 11 55 33 99

38. Struktura stog se dinamički najčešće predstavlja?

Rj: Jednostruko povezanom linearном listom

39. Koja slika prikazuje gomilu oblikovanu ulaznim nizom (11, 33, 4, 5, 22, 59) tako da za najgori slučaj složenost bude $O(n \log_2 n)$?

Rj:



40. Pretraživanje binarnog stabla najbrže je ako se radi o:

Rj: Sortiranom potpunom stablu

41. Što radi:

```
void ispisi(cvor *glava) {  
    cvor *p;  
    for (p=glava; p!=NULL;p=p->slijed)  
        printf("%d\n", p->element);  
}
```

Rj: Ispisuje sve vrijednosti elemenata jednostruko povezane linearne liste.

42. U prazno binarno stablo uneseni su elementi 20, 15, 1, 3, 7, 48, 12, 19, 35. Kolika je dubina stabla?

Rj: 6

43. Koja je od slijedećih tvrdnji za gomilu točna?

Rj: Gomila se koristi kada je do najvećeg/najmanjeg potrebno doći sa složenošću $O(1)$. Složenost reorganizacije nakon uklanjanja prvog člana je $O(\log_2 n)$. Složenost dodavanja novog člana u gomilu je $O(\log_2 n)$.

44. Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 50 5 7 10 13 1 8 algoritmom čija je složenost za najgori slučaj $O(n \log 2 n)$?

a) 50

13 10

8 7 5 1

b) 50

5 7

10 13 1 8

c) 50

13 7

5 10 1 8

d) 50

←

13 8

5 10 1 7

e) 50

13 8

10 7 5 1

45. Što ispisuje funkcija

```
void ispisi( struct cvor *glava ) {  
    if( glava != NULL && glava->elem % 2) {  
        printf(" %d ", glava->elem);  
        ispisi(glava->sljed);  
    }  
}
```

```
 }  
 }
```

ako se u jednostruko povezanoj listi na koju pokazuje parametar glava nalaze sljedeći cijeli brojevi :

1 57 43 13 8 11 20 10 56 53

a) **1 57 43 13 ←**

b) ne ispisuje ništa

c) 1 57 43 13 8 11

d) 1 57 43 13 8 11 20 10 56 53

e) 1 57 43 13 11 53

46. Ispravna deklaracija dvostrukog povezane liste u memoriji glasi:

a) struct s1 {

```
    int mbr;  
    char ime_pr[50];  
    int spol;  
    long pret;  
    long sljed;
```

}

b) struct s1 {

```
    int mbr;  
    char ime_pr[50];  
    int spol;  
    zapis *pret;  
    zapis *sljed;
```

}

c) struct s1 {

```
    int mbr;  
    char ime_pr[50];  
    int spol;  
    struct s1 *sljed;
```

}

d) **typedef struct s1{ ←**

```
    int mbr;  
    char ime_pr[50];  
    int spol;  
} zapis1;
```

typedef struct s2{

```
    zapis1 element;
    struct s2 *pred;
    struct s2 *sljed;
} zapis;
```

```
e) struct s1 {
    int mbr;
    char ime_pr[50];
    int spol;
    long *pret;
    long *sljed;
}
```

47. Što će se ispisati funkcijom:

```
void ispis (cvor *korijen) {
    printf ("%c", korijen->element);
    if (korijen->lijevo && korijen->desno) {
        ispis (korijen->desno);
        ispis (korijen->lijevo);
    }
}
```

za stablo na slici pozivom funkcije

1

2 3

5 6 4

ispis (korijen);

ako je korijen u trenutku poziva pokazivač na korijen stabla?

- a) 134625
 - b) 123456
 - c) 521634
 - d) 13462 ←**
 - e) 12364
-

48. Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 5, 10, 7, 3, 1, 90 algoritmom čija je složenost za najgori slučaj $O(n)$?

a) 90

7 10

3 1 5

b) 90 ←

10 7

3 1 5

c) 90

5 10

3 1 5

d) 90

10 7

5 1 3

e) 5

10 7

3 1 5

49. Koliko čvorova ima *koso* stablo s **n** razina?

a) 2^*n-1

b) $n+1$

c) 2^n-1

d) **n** ←

e) 2^n

50. Što će ispisati funkcija:

```
void ispis (cvor *korijen) {  
    printf ("%c", korijen->element);  
    if (korijen->lijevo && korijen->desno) {  
        ispis (korijen->desno);  
        ispis (korijen->lijevo);  
    }  
}
```

za stablo

1
2 3
5 6 4

RJ: 1 3 4 6 2

51. Kada se gomila oblikuje dodavanjem jednog po jednog elementa u stablo uz očuvanje strukture gomile, tada je vrijeme izvođenja oblikovanja gomile za najgori slučaj (n je broj ulaznih elemenata):

- a) $O(\log_2 n)$
 - b) $O(n)$
 - c) $O(n^2 \cdot \log_2 n)$
 - d) $O(n \cdot \log_2 n)$ ←**
 - e) $O(n^2)$
-

52. Od elemenata {20, 15, 1, 3, 7, 48, 12, 19, 35} formirano je sortirano binarno stablo. Kolika je dubina stabla?

RJ: 6

53. Koja procedura pronađe zadani element u jednostruko povezanoj listi?

a)

```
cvor *trazi1(cvor *glava, tip element) {
    cvor *p;
    for (p=glava; p!=NULL; p++)
        if (p->element==element)
            return p;
    return NULL;
}
```

b)

```
cvor *trazi1(cvor *glava, tip element) {
```

```
cvor *p;  
if (p->element==element) return p;  
return NULL;  
}
```

c)

```
cvor *trazi1(cvor *glava,tip element) {  
cvor *p;  
for (p=glava;p!=NULL;p=p->slijed)  
if (p->element!=element)  
return p;  
return NULL;  
}
```

d)

```
cvor *trazi1(cvor *glava,tip element){  
cvor *p;  
for (p=glava;p!=NULL;p=p->slijed)  
if (p->element==element)  
return p;  
return NULL;  
}
```

e)

```
cvor *trazi1(cvor *glava,tip element){  
cvor *p;  
for (p=glava;p!=NULL;p++)  
if (p->element!=element)  
return p;  
return NULL;
```

}

54. Koji je stupanj stabla s **n** razina?

- a) najmanji stupanj nekog čvora u stablu
 - b) n
 - c) najveći stupanj nekog čvora u stablu** ←
 - d) broj čvorova u stablu
 - e) broj čvorova u potpunom stablu s **n** razina
-

55. Što radi zadana funkcija za poziv `f (glava, 7, &br)` ako je glava pokazivač na početak jednostruko povezane liste?

```
cvor *f(cvor *p,int broj,int *br) {  
    if (p) {  
        ++(*br);  
        if (p->broj==broj)  
            return p;  
        else  
            return f(p->sljed,broj,br);  
    } else return NULL;  
}
```

RJ: vraća pokazivač na čvor koji sadrži broj i njegov redni broj u listi (preko br)

56. Koja tvrdnja **nije točna** za ovu funkciju:

```
skiniizred(tip *element,tip red[], int n, int *izlaz,int ulaz){  
    if (ulaz==*izlaz) return 0;  
    (*izlaz)++;  
    *izlaz%=n;  
    *element=red[*izlaz];  
    return 1;  
}
```

RJ: Funkcija vraća 1 ako se iz reda može skinuti samo 1 element

57. Dinamička struktura za jednostruko povezanu listu sadrži:

RJ: Pokazivac na prvi element liste i proizvoljan broj čvorova

58. Ovdje je napisana funkcija za dodavanje elemenata u stog ostvaren jednostruko povezanim listom. Na kraju funkcije je izostavljena jedna naredba. Treba izabrati naredbu koja ide na to mjesto da funkcija radi.

Funkcija otprilike izgleda ovako:

```
dodajelement(element **vrh,int vrijednost) {  
    element novi = new element;  
    novi->vrijednost=vrijednost;  
    novi->slijed=*vrh;
```

```
##### <- Što nedostaje?  
}
```

RJ: Nedostaje naredba `*vrh=novi;`

59. Treba li se kod funkcije dodaj_u_red (red je ostvaren listom) mijenjati i pokazivac "izlaz"?

RJ: Treba, jer ako je red prazan, moramo ga postaviti na prvi dodani element

60. Kako izgleda funkcija za dodavanje elemenata u red ostvaren poljem?

RJ: int dodajured (struct zapis *red,int *ulaz,int izlaz, zapis elem, int n)

61. Sto ce se dogoditi nakon sto u ovu gomilu ubacimo broj 35 i pozovemo funkciju "ubaci"?

66 57 32 30 36 ← ubacujemo broj 35

RJ: Zamijeniti ce se 35 i 32

62. Koja procedura trazi element u listi?

RJ:

```
cvor *trazi (cvor *g, tip elem) {  
    cvor *p;
```

```
for (p=g;p!=NULL;p=p->slijed)
    if (p->vrijednost==elem) return p;
return NULL;
}
```

63. Koja je slozenost funkcije ubaci_u_red za red izveden poljem

RJ: O(1)

64. Koji je prototip funkcije za dodavanje u red poljem. Ako je red pun, polje se dinamički udvostruči. Fja vraća 1 ako je dodavanje uspjelo, inače vraća 0.

RJ: int DodajURed (int element, int *red, int n, int *izlaz, int *ulaz)

65. Zadano je:

```
1
2   3
5   9 4
```

Što će se ispisati s:

```
void ispis (cvor *korijen) {
    if korijen{
        ispis (korijen->desno);
        ispis (korijen->ljevo);
        printf("%c",korijen->element);
    }
}
```

RJ: 4 9 3 5 2 1

66. Je li u funkciji `dodajured(int element, cvor **ulaz, cvor **izlaz)` potrebno primati pokazivač na izlaz iz reda po referenci i zašto?

RJ: Da, zato što u slučaju NULL vrijednosti glave podaci bivaju izbrisani (il neš u tom stilu)

67. Dinamička podatkovna struktura za real. jednostruko povezane liste sastoji se od:

RJ: pokazivača na prvi element liste i proizvoljnog broja povezanih čvorova

68. koja je složenost dohvaćanja zadnjeg elementa reda (red je realiziran listom)?

RJ: $O(n)$

69. Koji je prototip funkcije za skidanje elemenata iz reda listom (funkcija vraća 1 ako je uspjela skinuti element, inače vraća 0)?

RJ: `int SkiniiIzReda (int *element, atom **ulaz, atom **izlaz)`

70. koja je složenost funkcije koja skida ZADNJI element iz reda?

RJ: $O(n)$

71. Koja tvrdnja nije ispravna za zadanu funkciju?

```
cvor *trazi(cvor *glava, int sifra){  
    if (glava->sifra==sifra) return glava;  
    if (glava->slijed)  
        return trazi(glava->slijed, sifra);  
    else  
        return NULL;  
}
```

RJ: F-ja nije ispravna, ne radi za praznu listu.

72. Sto ce se dogoditi pozivom f(glava), f(glava) f-je:

```
void f(cvor *glava) {  
    if (glava->slijed) {  
        printf ("%s\n", glava->ime);  
        f(glava->slijed);  
    }  
}
```

RJ: Dva puta ispisuje imena osim od posljednjeg cvora.

73. U red jednostruko povezanom listom pohranjuju se cjelobrojni zapis. Prototip f-je za skidanje iz reda (1 za uspjesno, 0 neuspjesno obavljeno)

RJ: skin (cvor **ulaz, cvor **izlaz, int *element)

74. Red ostvaren ciklickim poljem, koja je tvrdnja neistinita?

```
int SkiniiZReda(tip *element, tip red[], int n, int *izlaz, int ulaz){  
    if(ulaz==*izlaz) return 0;  
    (*izlaz)++;  
    *izlaz %=n;  
    *element=red[*izlaz];  
    return 1;  
}
```

RJ: F-ja vraca 0, ako se iz reda moze skinuti tocno 1 element.

75. Koja je tvrdnja za jednostruko povezanu linearu listu je istinita:

RJ: može ostvariti statičkom strukturu polje

76. Koja od ponuđenih funkcija ispravno implementira Heap sort?

a) void HeapSort(tip A[],int n) {
 int i;
 StvoriGomilu(A,n/2);
 for (i=n/2;i>=0;i--) {
 Zamijeni(&A[1],&A[i]);
 Podesi(A,1,i-1);
 }
}

b) void HeapSort(tip A[],int n) {
 int i;
 StvoriGomilu(A,n);
 for (i=n;i>=2;i--) {
 Zamijeni(&A[1],&A[i]);
 Podesi(A,1,i-1);
 }
}

c) void HeapSort(tip A[],int n) {
 int i;
 StvoriGomilu(A,1);
 for (i=1;i<=n/2;i++) {
 Zamijeni(&A[n],&A[1]);
 Podesi(A,1,i+1);
 }
}

d) void HeapSort(tip A[],int n) {
 int i;
 StvoriGomilu(A,1);
 for (i=1;i<=n;i++) {
 Zamijeni(&A[n],&A[1]);

```

        Podesi(A,1,i+1);

    }

}

e) void HeapSort(tip A[],int n) {
    int i;
    StvoriGomilu(A,n);
    for (i=n/2;i<=0;i--) {
        Zamijeni(&A[1],&A[i]);
        Podesi(A,1,i-1);
    }
}

```

Funkcija za dodavanje elementa na stoga realiziran listom glasi:

```

int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = element;
        novi->sljed = stog->vrh;
        stog->vrh = novi;
        return 1;
    }
    else return 0;
}

int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = element;
        return 1;
    }
    else return 0;
}

int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = stog->vrh;
        novi->sljed = stog->vrh;
        return 1;
    }
    else return 0;
}

```

```

int dodaj (int element, Stog *stog) {
    atom *novi;

```

```

novi->element = element;
novi->sljed = stog->vrh;
stog->vrh = novi;
return 1;
}

int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi = element;
        return 1;
    }
    else return 0;
}

```

Funkcija push stavlja elemente na stog. Ako je operacija uspješno obavljena funkcija vraća 1, a u slučaju greške vraća 0. Prototip funkcije je:

```
int push (int element, Stog *stog);
```

Funkcija pop skida element sa stoga i vraća njegovu vrijednost ili -1 u slučaju greške. Prototip funkcije pop je:

```
int pop (Stog *stog);
```

Što će ispisati sljedeći programski odsječak, uz prepostavku da je prije izvođenja stog prazan i da na njemu ima dovoljno mesta.

```

for (i=0; i<5; i++)
    push(i, &stog);

for (i=5; i>=0; i--)
    printf("%d ", pop(&stog));

```

```

0 1 2 3 4
4 3 2 1 0
5 4 3 2 1 0
4 3 2 1 0 -1
0 1 2 3 4 5

```

Funkcija push stavlja elemente na stog. Ako je operacija uspješno obavljena funkcija vraća 1, a u slučaju greške vraća 0. Prototip funkcije je:

```
int push (int element, Stog *stog);
```

Funkcija pop skida element sa stoga i vraća njegovu vrijednost ili -1 u slučaju greške. Prototip funkcije je:

```
int pop (Stog *stog);
```

Što će ispisati sljedeći programski odsječak, uz prepostavku da je prije izvođenja stog prazan i da na njemu ima dovoljno mesta.

```

printf("%d ", push(pop(&stog),&stog));
printf("%d ", pop(&stog));

```

```

0 -1
-1 1
1 -1
1 0
1 1

```

Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

AHYCO

Provjera: ASP Blic 2

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10 11 12
c 28 m 31 s

Broj mogućih točnih odgovora: 1

Ocjenjivanje provjere

Pretpostavak je da postoje funkcije za operacije nad stogom inicijalizacija `skini` i dodaj sa sljedećim prototipima:
`int skini (int *element, Stog *stog);`
`int dodaj (int element, Stog *stog);`
Što će ispisati sljedeći program?

```
#include <stdio.h>
#define MAXSTOG 100
typedef struct {
    int vrh, polje[MAXSTOG];
} Stog;
void init_stog(Stog *stog){
    stog->vrh = -1;
}
int main() {
    Stog stog;
    int a=1,b=2,c=3;
    init_stog(&stog);
    dodaj(a,&stog);
    dodaj(b,&stog);
    dodaj(c,&stog);
    skini(&a,&stog);
    skini(&c,&stog);
    skini(&b,&stog);
    printf("%d %d %d",a,b,c);
}
```

Done

start zadi - Paint Glavni izbornik - Mozilla... Pofuk, Robert (00364... HR 9:43

Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

AHYCO

Provjera: ASP Blic 2

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10 11 12
c e 27 m 59 s

Broj mogućih točnih odgovora: 1

Ocjenjivanje provjere

Koja od sljedećih tvrdnji vezanih uz rekurziju JE istinita?

- a) rekurziju koristi isključivo globalne varijable za prijenos vrijednosti
- b) maksimalni broj poziva rekurzije obrnuto je proporcionalan broju registara procesora
- c) ne mora postojati slučaj koji se rješava bez rekurzije
- d) rekurzija je istovremen poziv dva ili više potprograma
- e) za pohranu podataka i povratak iz rekurzije koristi se stog

Done

start zadi - Paint Glavni izbornik - Mozilla... Pofuk, Robert (00364... HR 9:44

Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

AHyCo

Provjera: ASP Blic 2

Robert Pofuk [Odjava](#)

Redni broj pitanja: 5

1 2 3 4 5 6 7 8 9 10 11 12

c e c e b

25 m 16 s

Broj mogućih točnih odgovora: 1

Ocenjivanje provjere

Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha i 0 u slučaju neuspjeha i ima prototip
int dodaj (int element, Stog *stog);
što će biti na stogu nakon obavljanja sljedeće naredbe, uz pretpostavku da je stog bio prazan, da na njega stane najmanje 3 elementa i da stog raste s lijeva na desno?
dodaj(dodaj(dodaj(5, &stog)+2, &stog), &stog);

- | | |
|----|-------|
| a) | 5 3 4 |
| b) | 5 3 1 |
| c) | 1 1 1 |
| d) | 5 1 1 |
| e) | 1 3 5 |

Done

start zad34 - Paint Glavni izbornik - Mozilla... Pofuk, Robert (00364... HR 9:47

Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha a 0 u slučaju neuspjeha i ima prototip
int dodaj (int element, Stog *stog);
a funkcija skidanja sa stoga vraća vrijednost elementa s vrha ili -1 ako je stog prazan i ima prototip
int skin (int *element, Stog *stog);
što će biti na stogu nakon obavljanja sljedećih naredbi, uz pretpostavku da je stog bio prazan i da stog raste s lijeva na desno:
dodaj (5, &stog);
dodaj (dodaj(skin(&element, &stog), &stog), &stog);

- | | |
|----|---------------------|
| a) | 5 |
| b) | stog će biti prazan |
| c) | 5 0 |
| d) | -1 1 |
| e) | 5 1 |

Done

start zad5 - Paint Glavni izbornik - Mozilla... Pofuk, Robert (00364... HR 9:48

Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

AHyCo

Provjera: ASP Blic 2

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10 11 12

c e c e b

25 m 16 s

Broj mogućih točnih odgovora: 1

Ocenjivanje provjere

Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha i 0 u slučaju neuspjeha i ima prototip
int dodaj (int element, Stog *stog);
što će biti na stogu nakon obavljanja sljedeće naredbe, uz pretpostavku da je stog bio prazan, da na njega stane najmanje 3 elementa i da stog raste s lijeva na desno?
dodaj(dodaj(dodaj(5, &stog)+2, &stog), &stog);

- | | |
|----|-------|
| a) | 5 3 4 |
| b) | 5 3 1 |
| c) | 1 1 1 |
| d) | 5 1 1 |
| e) | 1 3 5 |

Done

start zad34 - Paint Glavni izbornik - Mozilla Firefox Pofuk, Robert (00364... 9:47

Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha a 0 u slučaju neuspjeha i ima prototip
int dodaj (int element, Stog *stog);
a funkcija skidanja sa stoga vraća vrijednost elementa s vrha ili -1 ako je stog prazan i ima prototip
int skinji (int *element, Stog *stog);
što će biti na stogu nakon obavljanja sljedećih naredbi, uz pretpostavku da je stog bio prazan i da stog raste s lijeva na desno:
dodaj (5, &stog);
dodaj (dodaj(skinji(&element, &stog), &stog), &stog);

- | | |
|----|---------------------|
| a) | 5 |
| b) | stog će biti prazan |
| c) | 5 0 |
| d) | -1 1 |
| e) | 5 1 |

Done

start zad5 - Paint Glavni izbornik - Mozilla Firefox Pofuk, Robert (00364... 9:48

Glavni izbornik - Mozilla Firefox

http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox

Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10 11 12

a

29 m 47 s

Ocenjivanje provjere

Broj mogućih točnih odgovora: 1

Struktura stog kao **dinamička struktura** najčešće se predstavlja:

a) Jednostruko povezanim linearnom listom

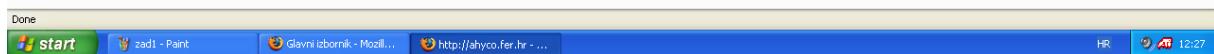
b) Struktura stog ne može se predstaviti dinamičkom strukturu

c) Dvostruko povezanim linearnom listom

d) Dinamičkom strukturom gomila

e) Binarnim stablom

Done



zad2 - Paint

http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox

Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja: 2

Broj mogućih točnih odgovora: 1

Što radi sljedeća funkcija?

```
void func( cvor *k, int *br ) {
    if( k != NULL ) {
        func(k->l, br);
        if( k->l == NULL && k->d == NULL ) (*br)++;
        func(k->d, br);
    }
}
```

a) broji listove u stablu
b) računa zbroj elemenata u stablu
c) broji elemente stabla
d) broji parne elemente u stablu
e) ništa od navedenoga

29 m 00 s

Ocenjivanje provjere

Done

For Help, click Help Topics on the Help Menu.

start Glavni izbornik - Mozilla... http://ahyco.fer.hr - ... zad2 - Paint HR 12:28

Glavni izbornik - Mozilla Firefox

http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox

Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja: 4

Broj mogućih točnih odgovora: 1

1 2 3 4 5 6 7 8 9 10 11 12

a a a

27 m 24 s

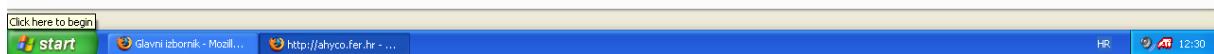
Ocenjivanje provjere

Uz prethodno ispravno deklarirane sve podatke i već formiranu jednostruku povezanu linearu listu, što radi sljedeća funkcija:

```
void ispis (cvor *glava) {
    cvor *p;
    for (p = glava; p != NULL; p = p->slijed) {
        printf ("%d\n", p->element);
    }
}
```

a) ispisuje vrijednost svakog drugog elemenata jednostruko povezane linearne liste
b) funkcija ne radi ništa jer je tipa void
c) ispisuje vrijednosti elemenata svih čvorova binarnog stabla
d) ispisuje vrijednosti parnih elemenata jednostruko povezane linearne liste
e) ispisuje vrijednosti svih elemenata jednostruko povezane linearne liste

Done



Glavni izbornik - Mozilla Firefox
http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox

Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja: 5

1 2 3 4 5 6 7 8 9 10 11 12

a a a e

26 m 02 s

Broj mogućih točnih odgovora: 1

Ocenjivanje provjere

Koja funkcija vraća najveći element u stablu (koje je uređeno po principu lijevo veći a desno manji) ? U stablu se nalaze cijeli brojevi.

a)

```
int func( cvor *k ) {  
    while( k->d != NULL ) k = k->d;  
    return k->elem;  
}
```

b)

```
int func( cvor *k ) {  
    func(k->l);  
    func(k->d);  
}
```

c)

```
int func( cvor *k ) {  
    func(k->d);  
    func(k->l);  
}
```

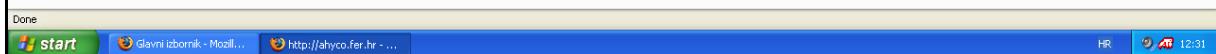
d)

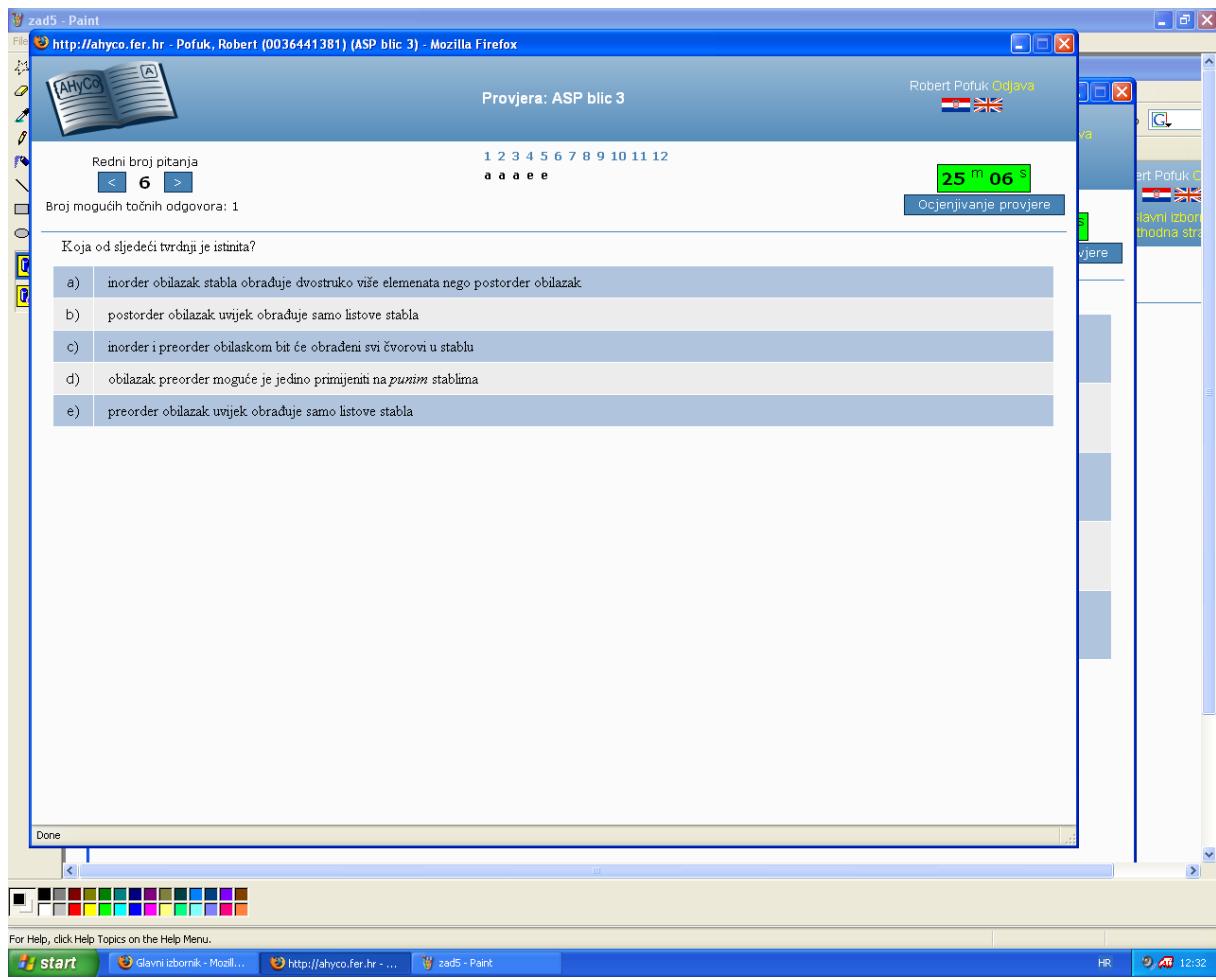
```
int func( cvor *k ) {  
    while( k->l != NULL ) k = k->l;  
    return k->elem;  
}
```

e)

```
int func( cvor *k ) {  
    if( k->l != NULL ) return func(k->l);  
    else return k->elem;  
}
```

Done





Glavni izbornik - Mozilla Firefox

http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox

Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja: 7

1 2 3 4 5 6 7 8 9 10 11 12

a a a e c

24 m 27 s

Ocenjivanje provjere

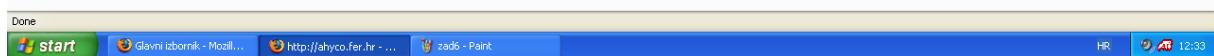
Broj mogućih točnih odgovora: 1

Uz prethodno ispravno deklarirane sve podatke i već formirano binarno stablo, što radi sljedeća funkcija:

```
void ispis (cvor *glava) {
    if (glava != NULL) {
        ispis (glava->lijevo);
        ispis (glava->desno);
        printf ("%s \n", glava->element);
    }
}
```

a) uvijek ispisuje samo vrijednost elementa na koji pokazuje glava
b) preorder ispisuje vrijednosti elemenata stabla
c) inorder ispisuje vrijednosti elemenata stabla
d) postorder ispisuje vrijednosti elemenata stabla
e) funkcija ne radi ništa jer je tip void

Done



Glavni izbornik - Mozilla Firefox

http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox

Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja: 8

1 2 3 4 5 6 7 8 9 10 11 12

a a a e c b

23 m 29 s

Ocenjivanje provjere

Broj mogućih točnih odgovora: 1

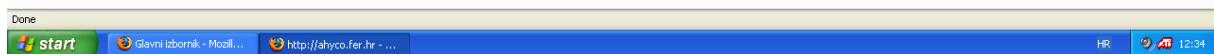
Neka se u gomili koja je pohranjena u polje nalaze sljedeći podaci:

66	57	32	30	36
----	----	----	----	----

Što će se desiti kada se u takvu gomilu doda podatak 35 (a nakon što se pozove funkcija *ubaci* koja dodaje elemente u gomilu)?

- a) Broj 35 će zamijeniti mjesto s brojem 32
- b) Taj će se broj dodat na kraj polja jer je time očuvano svojstvo gomile
- c) Izbaci se broj 66, a 35 se umetne na to mjesto
- d) Broj 35 će zamijeniti mjesto s brojem 30
- e) Broj 35 će zamijeniti mjesto s brojem 36

Done



untitled - Paint

http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox

Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja: 9 / 12

Broj mogućih točnih odgovora: 1

U jednostruko povezani listu spremaju se zapisi slijedećeg tipa:

```
typedef struct s1{
    int mbr;           // matični broj studenta
    char ime[40+1];   // ime studenta
    float prosjek;    // prosjek ocjena
    struct s1 *sljed;
} zapis;
```

Kako glasi funkcija koja pronalazi u listi zapis o studentu sa zadanim matičnim brojem i vraća taj zapis u glavni program. Ako takav student ne postoji u listi funkcija vraća NULL. Lista nije sortirana.

a) `zapis *nadji(zapis *glava, int p_mbr) {
 if (glava == NULL) return NULL;
 if (glava->mbr == p_mbr) return glava;
 else glava = glava->sljed;
}`

b) `zapis *nadji(zapis *glava, int p_mbr) {
 while (glava && (glava->mbr < p_mbr))
 glava = glava->sljed;
 return glava;
}`

c) `zapis *nadji(zapis *glava, int p_mbr) {
 while (glava && (glava->mbr != p_mbr))
 glava = glava->sljed;
 return glava;
}`

d) `void nadji(zapis *glava, int p_mbr, zapis element) {
 while (glava && (glava->mbr != p_mbr))
 glava = glava->sljed;
 element = *glava;
}`

e) `void nadji(zapis *glava, int p_mbr, zapis element) {
 while (glava && (glava->mbr != p_mbr))
 glava = glava->sljed;
}`

Done

For Help, click Help Topics on the Help Menu.

Untitled - Paint

start Glavni izbornik - Mozilla... http://ahyco.fer.hr - ... untitled - Paint

HR 12:35

zad9 - Paint

File <http://ahyco.fer.hr> - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox

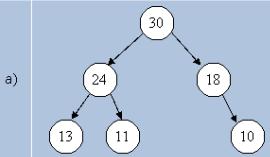
Redni broj pitanja < 10 >

1 2 3 4 5 6 7 8 9 10 11 12
a a a e e c b a c

20 m 01 s

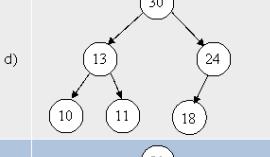
Ocenjivanje provjere

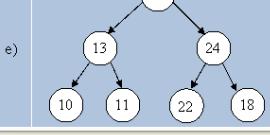
Koje od sljedećih binarnih stabala NE zadovoljava svojstva gomile?

a)  A binary tree with root 30. It has left child 24 and right child 18. Node 24 has left child 13 and right child 11. Node 18 has right child 10.

b)  A binary tree with root 30. It has no children.

c)  A binary tree with root 30. It has left child 24.

d)  A binary tree with root 30. It has left child 13 and right child 24. Node 13 has left child 10 and right child 11. Node 24 has right child 18.

e)  A binary tree with root 30. It has left child 13 and right child 24. Node 13 has left child 10 and right child 11. Node 24 has left child 22 and right child 18.

Done

e) glava = glava->aliisti

For Help, click Help Topics on the Help Menu.

start Glavni izbornik - Mozilla... http://ahyco.fer.hr... zad9 - Paint HR 12:37

Glavni izbornik - Mozilla Firefox

http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox



Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja
11

1 2 3 4 5 6 7 8 9 10 11 12
a a a e c b a c a

18 m 41 s

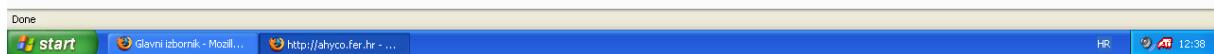
Broj mogućih točnih odgovora: 1

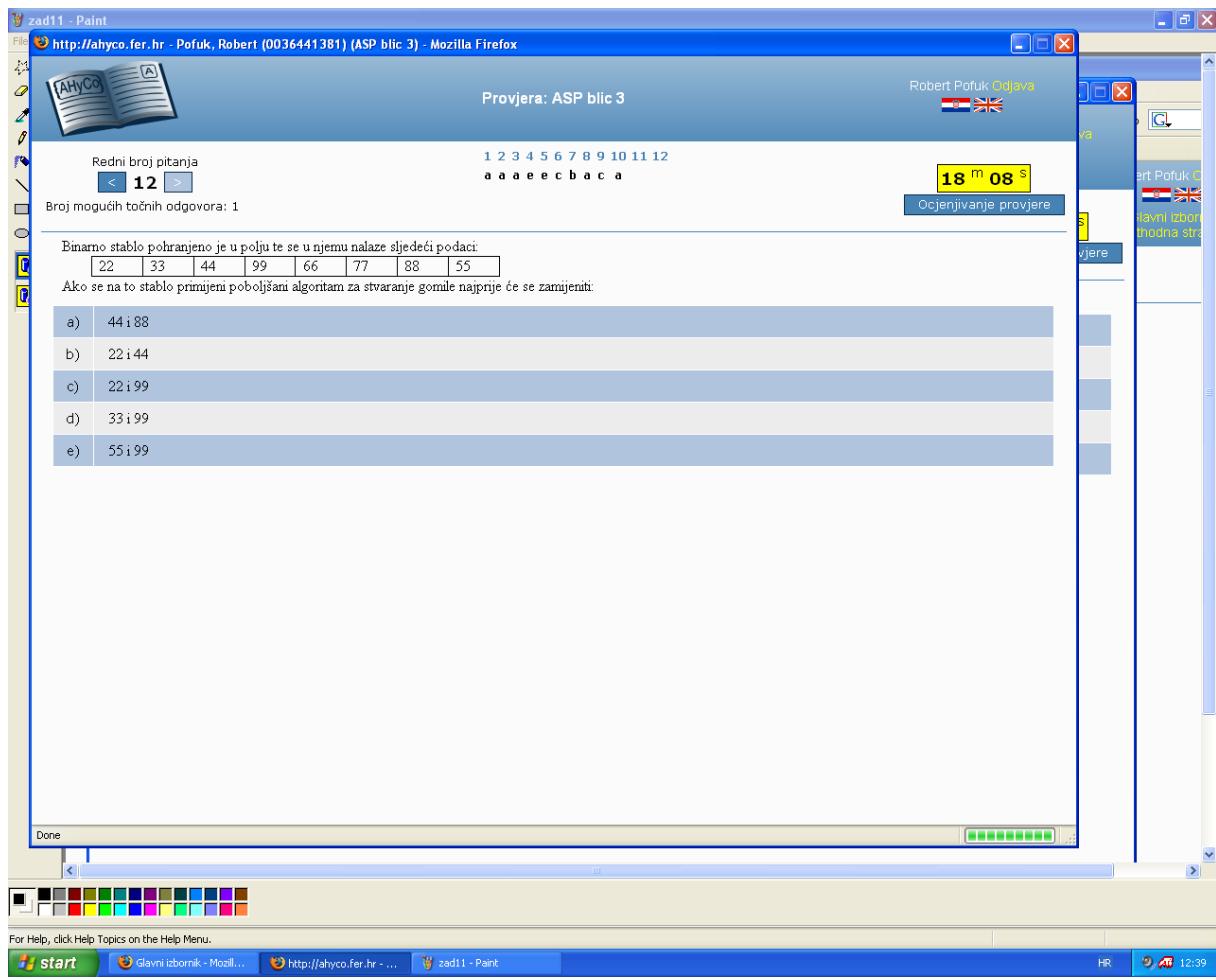
Ocjenjivanje provjere

Ako je gomila realizirana u polju, u kojem od sljedećih slučajeva elementi zapisani u polju NE zadovoljavaju svojstva gomile?

a) 10 9 8 7 6 5 4 3 2 1
b) 10 8 9 4 5 6 7 1 2 3
c) 10 8 9 6 7 4 5 2 3 1
d) 10 7 8 9 6 5 4 3 2 1
e) 10 9 8 4 5 6 7 1 2 3

Done





Medvidović, Marijan (0036447637) (ASP Blic 2) - Mozilla Firefox
http://ahyo.hr/provjere/Provjera.aspx
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

Ako je zadana struktura:

```
#define MAXSTOG 10
typedef struct {
    int vrh, polje[MAXSTOG];
} Stog;
i funkcija stavljanja na stog:
int dodaj (int element, Stog *stog) {
    if (stog->vrh >= MAXSTOG-1) return 0;
    stog->vrh++;
    stog->polje[stog->vrh] = element;
    return 1;
}
```

Na početku programa je:

```
stog->vrh = -1;
```

Koliko se ukupno elemenata može pohraniti na stog?

a) 8 elemenata
b) 9 elemenata
c) 5 elemenata
d) 10 elemenata

Done

start Medvidović, Marijan (...) My Pictures Copy of untitled - Paint HR 10:18

Medvidović, Marijan (0036447637) (ASP Blic 2) - Mozilla Firefox
http://ahyco.hr/Provjere/Provjera.aspx

 Pregledavanje rezultata provjere: ASP Blic 2

Marijan Medvidović Odjava

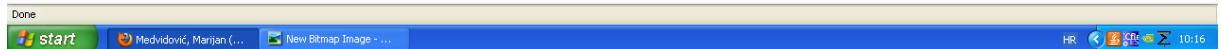

Redni broj pitanja
1 Odgovor
Broj mogućih točnih odgovora: 1 Točan
Mogući broj bodova: 0,50 odgovor
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10
c d a c e b d c a e
c d a c e b d c a e


Kraj pregledavanja

Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha a 0 u slučaju neuspjeha i ima prototip
int dodaj (int element, Stog *stog);
a funkcija skidanja sa stoga vraća vrijednost elementa s vrha ili -1 ako je stog prazan i ima prototip
int skinji (int *element, Stog *stog);
što će biti na stogu nakon obavljanja sljedeće naredbe, uz pretpostavku da je stog bio prazan i da stog raste s lijeva na desno:
dodaj (dodaj (skinji (&element, &stog), &stog), &stog);

- | | |
|----|---------------------|
| a) | 1 |
| b) | 1 1 |
| c) | -1 1 |
| d) | 0 |
| e) | stog će biti prazan |



 http://ahyco.fer.hr/Provjere/Provjera.aspx

Pregledavanje rezultata provjere: ASP Blic 2

Redni broj pitanja
2 Odgovor
Broj mogućih točnih odgovora: 1 Točan odgovor
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,00

1 2 3 4 5 6 7 8 9 10
d e a d c c e e b
d c e a c b c e e b

Kraj pregledavanja

Zadana je struktura:

```
#define MAXSTOG 5
typedef struct {
    int vrh, polje[MAXSTOG];
} Stog;
```

Kako glasi prototip funkcije za stavljanje cijelog broja na stog (funkcija vraća 0 ili 1, ovisno o tome da li se zapis uspio pohraniti na vrh stoga):

a) void dodaj (int element, Stog stog);
b)
c) int dodaj (int element, Stog *stog);
d)
e)

Waiting for http://ahyco.fer.hr/Provjere/TekstOdgovora.aspx?IdKlasичноPitanjeOdgovor=2628&FrameId=ctl00_cphBody_1

Pregledavanje rezultata provjere: ASP Blic 2

Redni broj pitanja
1 Odgovor
Broj mogućih točnih odgovora: 1 Točan odgovor
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10
d e a d c c e e b
d c e a c b c e e b

Kraj pregledavanja

Koja od sljedećih nizova naredbi u pseudokodu će zamijeniti vrijednost varijabli A i B pomoću stoga:

a) stavi(A); stavi(Pom); stavi(B); stavi(Pom); skinici(A); stavi(Pom);skinici(B);
b) stavi(B); skinici(A);
c) stavi(A); skinici(B);
d) stavi(A); stavi(B); skinici(A); skinici(B);
e) stavi(A); skinici(B); stavi(B); skinici(A);

Redni broj pitanja

< 2 >

Odgovor

Broj mogućih točnih odgovora: 1

Točan

Mogući broj bodova: 0,50

Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

c a a e e b d b a e

c a a e e b d a a e



Kraj pregledavanja

Funkcija za dodavanje elemenata na stoga realiziran listom glasi:

```
a) int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = element;
        novi->sljed = stog->vrh;
        stog->vrh = novi;
        return 1;
    }
    else return 0;
}
```

```
b) int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi = element;
        return 1;
    }
    else return 0;
}
```

```
int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = stog->vrh;
```

Redni broj pitanja

< 7 >

Odgovor

Broj mogućih točnih odgovora: 1

Točan

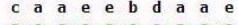
Mogući broj bodova: 0,50

Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

c a a e e b d b a e

c a a e e b d a a e



Kraj pregledavanja

Funkcija push stavlja elemente na stog. Ako je operacija uspješno obavljena funkcija vraća 1, a u slučaju greške vraća 0. Prototip funkcije je:

```
int push (int element, Stog *stog);
```

Funkcija pop skida element sa stoga i vraća njegovu vrijednost ili -1 u slučaju greške. Prototip funkcije pop je:

```
int pop (Stog *stog);
```

Što će ispisati sljedeći programski odsječak, uz prepostavku da je prije izvođenja stog prazan i da na njemu ima dovoljno mesta.

```
for (i=0; i<5; i++)
    push(i, &stog);

for (i=5; i>=0; i--)
    printf("%d ", pop(&stog));
```

a) 0 1 2 3 4

b) 5 4 3 2 1 0

c) 4 3 2 1 0

d) 4 3 2 1 0 -1

e) 0 1 2 3 4 5

AHyCo

Provjera: ASP blic 3

Odjava

Redni broj pitanja
1 >

Broj mogućih točnih odgovora: 1

1 2 3 4 5 6 7 8 9 10
c b e a e b e e d c

13 m 00 s

Ocenjivanje provjere

Predložena je funkcija za pronađak čvora sa zadanim cijelobrojnom šifrom u jednostruko povezanoj linearnej listi:

```
cvor *trazi(cvor *glava, int sifra){  
    if (glava->sifra == sifra) return glava;  
    if (glava->sljed)  
        return trazi(glava->sljed, sifra);  
    else  
        return NULL;  
}
```

Koja od sljedećih tvrdnji (koje se odnose na predloženu funkciju) je ispravna?

a) Funkcija nije ispravna: pretražuje sve čvorove liste osim zadnjeg!
b) Funkcija je ispravna i vraća pokazivač na čvor sa zadanim šifrom ili NULL ako čvor sa zadanim šifrom ne postoji u listi.
c) Funkcija nije ispravna: ne radi za praznu listu (uzrokuje pogrešku kod poziva)!
d) Funkcija nije ispravna: ukoliko čvor sa zadanim šifrom ne postoji u listi, funkcija neće "nikad" završiti.
e) Funkcija nije ispravna: pretražuje sve čvorove liste osim prvog!

Done

Internet 100%

AHyCo

Provjera: ASP blic 3

Odjava

Redni broj pitanja
2

Broj mogućih točnih odgovora: 1

1 2 3 4 5 6 7 8 9 10
c b e a e b e e d c

12 m 11 s

Ocjenjivanje provjere

Koja od prikazanih stabala su gomile:

A

```
graph TD; 6((6)) --> 3((3)); 6 --> 4((4)); 3 --> 2((2)); 4 --> 5((5))
```

B

```
graph TD; 6((6))
```

C

```
graph TD; 6((6)) --> 4((4)); 6 --> 7((7)); 4 --> 3((3))
```

D

```
graph TD; 6((6)) --> 3((3)); 6 --> 4((4)); 3 --> 2((2))
```

E

```
graph TD; 6((6)) --> 3((3)); 6 --> 2((2)); 3 --> 4((4))
```

a) B

d) A, B

c) A, B, E

d) A

e) A, B, C, E

Done

Internet

100%

 Provjera: ASP blic 3

Redni broj pitanja
3

Broj mogućih točnih odgovora: 1

Kada se gomila oblikuje dodavanjem jednog po jednog elementa u stablo uz očuvanje strukture gomile, tada je vrijeme izvođenja oblikovanja gomile za najgori slučaj (n je broj ulaznih elemenata):

a)	$O(n^2 * \log_2 n)$
b)	$O(n)$
c)	$O(n^2)$
d)	$O(\log_2 n)$
e)	$O(n * \log_2 n)$

1 2 3 4 5 6 7 8 9 10
c b e a e b e e d c

11 m 45 s

Ocjenjivanje provjere

Done

Internet 100%

 Provjera: ASP blic 3

Redni broj pitanja
4

Broj mogućih točnih odgovora: 1

1 2 3 4 5 6 7 8 9 10
c b e a e b e e d c

09 m 34 s

Ocjenjivanje provjere

U prazno binarno stablo (uređeno tako da se lijevo stavlja manji a desno veći broj) su redom ubaćeni sljedeći brojevi:
20, 15, 1, 3, 7, 48, 12, 19, 35

Kolika je dubina tako dobivenog stabla ?(uzmite da je korijen stabla na razini 1)

a)	6
b)	8
c)	5
d)	7
e)	4

Done

Internet 100%

AHyCo

Provjera: ASP blic 3

Java

Redni broj pitanja
5

1 2 3 4 5 6 7 8 9 10
c b e a e b e e d c

09 m 11 s

Ocjenjivanje provjere

Koja od sljedećih tvrdnji za jednostruko povezanu linearnu listu je istinita?

a) Zadnji podatak u listi pokazuje na prvi podatak iz te liste

b) Kad je lista prazna pokazivač glava pokazuje sam na sebe

c) Podaci se mogu dodavati isključivo na početak

d) Takva lista isključivo se realizira u datoteci na disku

e) Takva lista može se realizirati statičkom strukturu podataka (poljem)

Done

Internet

100%

AHyCo

Provjera: ASP blic 3

Odjava

Redni broj pitanja
1 2 3 4 5 6 7 8 9 10
c b e a e b e e d c

08 m 50 s

Ocjenjivanje provjere

Broj mogućih točnih odgovora: 1

Koja od sljedeći tvrdnji je istinita?

a) inorder obilazak stabla obrađuje dvostruko više elemenata nego postorder obilazak

b) inorder i preorder obilaskom bit će obrađeni svi čvorovi u stablu

c) postorder obilazak uvijek obrađuje samo listove stabla

d) obilazak preorder moguće je jedino primijeniti na *punim* stablima

e) preorder obilazak uvijek obrađuje samo listove stabla

Done

Internet

100%

 Provjera: ASP blic 3

Redni broj pitanja
7

Broj mogućih točnih odgovora: 1

Ako je gomila realizirana u polju, u kojem od sljedećih slučajeva elementi zapisani u polju zadovoljavaju svojstvo gomile ?

a) 10 1 9 4 5 7 8
b) 20 10 15 11 6 7 8
c) 20 7 15 8 6 10 11
d) 10 7 9 4 8 1 2
e) 20 3 15 2 1 14 10

Odjava |   08 m 30 s | Ocjenjivanje provjere

Done

AHyCo

Provjera: ASP blic 3

Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10
c b e a e b e e d c

Broj mogućih točnih odgovora: 1 08 m 13 s

Ocjenjivanje provjere

U jednostruko povezanu listu spremaju se zapisi slijedećeg tipa:

```
typedef struct s1{  
    int mbr;           // matični broj studenta  
    char ime[40+1];   // ime studenta  
    float prosjek;    // prosjek ocjena  
    struct s1 *sljed;  
} zapis;
```

Kako glasi funkcija koja pronađi u listi zapis o studentu sa zadanim matičnim brojem i vraća taj zapis u glavni program. Ako takav student ne postoji u listi funkcija vraća NULL. Lista nije sortirana.

a) `zapis *nadji(zapis *glava, int p_mbr) {
 if (glava == NULL) return NULL;
 if (glava->mbr == p_mbr) return glava;
 else glava = glava->sljed;
}`

b) `void nadji(zapis *glava, int p_mbr, zapis element) {
 while (glava && (glava->mbr != p_mbr))
 glava = glava->sljed;
 return glava;
}`

c) `zapis *nadji(zapis *glava, int p_mbr) {
 while (glava && (glava->mbr < p_mbr))
 glava = glava->sljed;
 return glava;
}`

d) `void nadji(zapis *glava, int p_mbr, zapis element) {
 while (glava && (glava->mbr != p_mbr))
 glava = glava->sljed;
 element = *glava;
}`

e) `zapis *nadji(zapis *glava, int p_mbr) {
 while (glava && (glava->mbr != p_mbr))
 glava = glava->sljed;
 return glava;
}`

Done 100% Internet

 Provjera: ASP blic 3

Redni broj pitanja
9

Broj mogućih točnih odgovora: 1

Koliku dubinu i broj elemenata u zadnjoj razini ima potpuno binarno stablo sa 300 elemenata? Pretpostaviti da je korijen stabla na dubini 1.

a) Ne može se odrediti.

b) Dubina = 10, broj elemenata zadnje razine stabla = 10.

c) Dubina = 9, broj elemenata zadnje razine stabla = 44.

d) Dubina = 9, broj elemenata zadnje razine stabla = 45.

e) Dubina = 300, broj elemenata zadnje razine stabla = 1.

1 2 3 4 5 6 7 8 9 10
c b e a e b e e d c

07 m 55 s

Ocjenjivanje provjere

Done

Glavni izbornik - Mozilla Firefox

http://ahyco.fer.hr - Pofuk, Robert (0036441381) (ASP blic 3) - Mozilla Firefox



Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10 11 12
3 a a

28 m 20 s

Broj mogućih točnih odgovora: 1

Ocjenjivanje provjere

Koja od sljedećih tvrdnji za jednostruko povezanu linearnu listu je istinita?

a) Podaci se mogu dodavati isključivo na početak
b) Zadnji podatak u listi pokazuje na prvi podatak iz te liste
c) Kad je lista prazna pokazivač glava pokazuje sam na sebe
d) Takva lista isključivo se realizira u datoteci na disku
e) Takva lista može se realizirati statičkom strukturu podataka (poljem)

Done

Done

start Glavni izbornik - Mozilla... http://ahyco.fer.hr... zad2 - Paint HR 12:29



Provjera: ASP blic 3

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10
10 c b e a e b e d c

07 m 33 s

Broj mogućih točnih odgovora: 1

Ocjenjivanje provjere

Ako je gomila realizirana u polju, u kojem od sljedećih slučajeva elementi zapisani u polju NE zadovoljavaju svojstva gomile?

a) 10 8 9 4 5 6 7 1 2 3
b) 10 9 8 4 5 6 7 1 2 3
c) 10 7 8 9 6 5 4 3 2 1
d) 10 9 8 7 6 5 4 3 2 1
e) 10 8 9 6 7 4 5 2 3 1

Done

Broj mogućih odgovora: 1 odgovor
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

Kakav je sadržaj stoga nakon izvođenja funkcije `funkcija`, ako stog prije poziva nije prazan?

Funkcije za operacije nad stogom skini i dodaj vraćaju 1 ako su obavile traženu zadaću, a 0 ako nisu, te imaju sljedeće prototipe:

```

int dodaj (tip element, Stog *stog);
int skini (tip *element, Stog *stog);

#include <stdio.h>
#define MAXSTOG 100

```

```

void funkcija(Stog *stog) {
    Stog pomStog;
    int i;
    init_stog(&pomStog);
    while (skini(&i, stog)) {
        if (i>=0) dodaj(i, &pomStog);
    }
    while (skini(&i, &pomStog)) {
        if (i<0){
            dodaj(i, stog);
        }
    }
}

```

a) Stog je prazan
b) Sadržaj stoga je nepoznat

Sadržaj stoga je nepoznat

1.što je gomila? (potpuno binarno stablo sa svojstvom da je svaki čvor veći od svoje djece)

Pitanja iz 2.blica:

1.što je gomila? (potpuno binarno stablo sa svojstvom da je svaki čvor veći od svoje djece)

2. zadan je uzlazni heap sort u obliku polja i pita te kako izgleda polje nakon prvog koraka heapsorta
zadano: 99 88 66 22 77 55 33 11

(prvo se zamijeni korijen i zadnji element pa se dobije:

11 88 66 22 77 55 33 99. i onda se sortira tako da ima opet svojstvo gomile, s tim da se zadnji član ne gleda. I dobije se:

88 77 66 22 11 55 33 99)

3. zadano je polje sortirano po relaciji *min heap* i pita te kako izgleda polje nakon prvog koraka sortiranja SILAZNIM heap sortom?

zadano: 11 55 22 66 88 99 33 77

opet zamijeniš korijen i zadnji element, time dobiješ:

77 55 22 66 88 99 33 11. i onda sortiraš tako da svaki čvor bude manji od svoje djece. Prvo zmjeniš 22 i 77, a u idućem koraku 77 i 33 (11 ne gledaš jer si ju izbacio). time dobiješ rezultat: 22 55 33 66 88 99 77 11

4. onaj kod gdje je odgovor da f-ja broji koliko ima čvorova s barem jednim djetetom

5. inicijalizacija stoga.. što će ispisati?

```
a=1;b=2;c=3;  
stavi (&a, *stog)  
stavi (&b, *stog)  
stavi (&c, *stog)  
skini (&c, *stog)  
skini (&a, *stog)  
skini (&b, *stog)  
printf ("%d %d %d", a, b, c) --> ispiše 2 1 3
```

6. moraš odabratи kod koji obilazi stablo inorderom. Točno je:

```
if (korijen!=NULL) {  
  
    ispis (korijen->lijevo);  
  
    printf (korijen->element);  
  
    ispis (korijen->desno);  
  
}
```

7. moraš odabratи strukturu kojom se određuje dvostruka povezana linearna lista (jedina koja ima "dva odlomka" :P), izgleda ovako:

```
typedef struct s1 {  
int mbr;  
char ime [20];  
int spol;  
}zapis1;  
typedef struct s2 {  
zapis1 element;  
struct s2 *pred;  
struct s2 *sljed;  
}zapis;
```

8. onaj kod gdje moraš izbaciti prvi element iz liste

9. dodaj (dodaj(dodaj (6, stog)+3, &stog), &stog) (na stogu je 6 4 1)

10. ono sa dubinom 300 gdje je odg 9 i 45

Dane su funkcije za rad sa stogom:

```
int dodaj (int element, Stog *stog);
int skini (int *element, Stog *stog);

koja vraćaju vrijednost 1 ako je operacija uspješno obavljena, tj. vrijednost 0 ako operacija nije obavljena. Što radi funkcija propis, uz pretpostavku da na stogovima ima dovoljno mesta?
```

```
void propis(Stog *stog1, Stog *stog2) {
    int element;
    if (skini(&element, stog1)) {
        propis(stog1, stog2);
        dodaj(element, stog2);
    }
}
```

- a) Zapisuje elemente stoga 1 u stog 1 obrnutim redoslijedom
- b) Funkcija samo prazni stog 1 i ne zapisuje ništa u stog 2.
- c) Premješta sve elemente stoga 1 na vrh stoga 2. Premješteni elementi na stogu 1 poredani obrnutim redoslijedom u odnosu na stog 2.
- d)** Premješta sve elemente stoga 1 na vrh stoga 2. Premješteni elementi na stogu 1 poredani istim redoslijedom kao na stogu 2.
- e) Zapisuje elemente stoga 2 u stog 2 obrnutim redoslijedom

Gospodnetić, Petra (0036449369) (ASP Blc 2) - Windows Internet Explorer
http://www.fes.hr/Project/Project.aspx

```
int dodaj (tip element, Stog *stog);
int skini (tip *element, Stog *stog);

#include <stdio.h>
#define MAXSTOG 100

void funkcije(Stog *stog) {
    Stog pomstog;
    int i;
    init_stog(&pomstog);
    while (skini(&i, stog)) {
        if (i>=0) dodaj(i, &pomstog);
    }
    while (skini(&i, &pomstog)) {
        if (i<0){
            dodaj(i, stog);
        }
    }
}

a) Stog sadrži samo elemente <=0
b) Sadržaj stoga je nepromijenjen
c) Sadržaj stoga je nepoznat
d) Stog je prazan
e) Stog sadrži samo elemente >0
```

Done

Redni broj pitanja

< 2 >

Odgovor

Broj mogućih točnih odgovora: 1

Točan

Mogući broj bodova: 0,50

Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

c a a e e b d b a e

c a a e e b d a a e



Kraj pregledavanja

Funkcija za dodavanje elemenata na stoga realiziran listom glasi:

```
a) int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = element;
        novi->sljed = stog->vrh;
        stog->vrh = novi;
        return 1;
    }
    else return 0;
}
```

```
b) int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi = element;
        return 1;
    }
    else return 0;
}
```

```
int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = stog->vrh;
```

Redni broj pitanja

< 7 >

Odgovor

Broj mogućih točnih odgovora: 1

Točan

Mogući broj bodova: 0,50

Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

c a a e e b d b a e

c a a e e b d a a e



Kraj pregledavanja

Funkcija push stavlja elemente na stog. Ako je operacija uspješno obavljena funkcija vraća 1, a u slučaju greške vraća 0. Prototip funkcije je:

```
int push (int element, Stog *stog);
```

Funkcija pop skida element sa stoga i vraća njegovu vrijednost ili -1 u slučaju greške. Prototip funkcije pop je:

```
int pop (Stog *stog);
```

Što će ispisati sljedeći programski odsječak, uz prepostavku da je prije izvođenja stog prazan i da na njemu ima dovoljno mesta.

```
for (i=0; i<5; i++)
    push(i, &stog);

for (i=5; i>=0; i--)
    printf("%d ", pop(&stog));
```

a) 0 1 2 3 4

b) 5 4 3 2 1 0

c) 4 3 2 1 0

d) 4 3 2 1 0 -1

e) 0 1 2 3 4 5

<http://ahyco.fer.hr/Provjere/Provjera.aspx>

Pregledavanje rezultata provjere: ASP Blic 2

Redni broj pitanja
2 Odgovor
Broj mogućih točnih odgovora: 1 Točan odgovor
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,00

1 2 3 4 5 6 7 8 9 10
d e a d c c e e b
d c e a c b c e e b

Kraj pregledavanja

Zadana je struktura:

```
#define MAXSTOG 5
typedef struct {
    int vrh, polje[MAXSTOG];
} Stog;
```

Kako glasi prototip funkcije za stavljanje cijelog broja na stog (funkcija vraća 0 ili 1, ovisno o tome da li se zapis uspio pohraniti na vrh stoga):

a) void dodaj (int element, Stog stog);
b)
c) int dodaj (int element, Stog *stog);
d)
e)

Waiting for http://ahyco.fer.hr/Provjere/TekstOdgovora.aspx?IdKlasickoPitanjeOdgovor=2628&FrameId=ctl00_cphBody_1

Gospodnetić, Petra (0036449360) (ASP Blic 2) - Windows Internet Explorer

Pregledavanje rezultata provjere: ASP Blic 2

Petra Gospodnetić Odjava

Redni broj pitanja
4 Odgovor
Broj mogućih točnih odgovora: 1 Točan odgovor
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10
d d e b a b d b e
d d c e c a b c b e

Kraj pregledavanja

Dane su funkcije za rad sa stogom:

```
int dodaj (int element, Stog *stog);
int skinji (int *element, Stog *stog);
```

koje vraćaju vrijednost 1 ako je operacija uspješno obavljena, tj. vrijednost 0 ako operacija nije obavljena. Što radi funkcija prepis, uz pretpostavku da na stogu ima mesta dovoljno mesta?

```
void prepis(Stog *stog1, Stog *stog2) {
    int element;
    if (skinji(&element, stog1)) {
        prepis(stog1, stog2);
        dodaj(element, stog2);
    }
}
```

a) Zapisuje elemente stoga 2 u stog 2 obrnutim redoslijedom
b)
c)
d)
e)

Waiting for http://ahyco.fer.hr/Provjere/TekstOdgovora.aspx?IdKlasickoPitanjeOdgovor=2812&FrameId=ctl00_cphBody_KontrolaZaPrikaz

Medvidović, Marijan (0036447637) (ASP Blic 2) - Mozilla Firefox
http://ahyo.hr/provjere/Provjera.aspx
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

Ako je zadana struktura:

```
#define MAXSTOG 10
typedef struct {
    int vrh, polje[MAXSTOG];
} Stog;
i funkcija stavljanja na stog:
int dodaj (int element, Stog *stog) {
    if (stog->vrh >= MAXSTOG-1) return 0;
    stog->vrh++;
    stog->polje[stog->vrh] = element;
    return 1;
}
```

Na početku programa je:

```
stog->vrh = -1;
```

Koliko se ukupno elemenata može pohraniti na stog?

a) 8 elemenata
b) 9 elemenata
c) 5 elemenata
d) 10 elemenata

Done

start Medvidović, Marijan (...) My Pictures Copy of untitled - Paint HR 10:18

Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

AHycO

Provjera: ASP Blic 2

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10 11 12

Broj mogućih točnih odgovora: 1

c 28 m 31 s

Ocenjivanje provjere

Pretpostavkaje da postoje funkcije za operacije nad stogom inicijalizacija skinii i dodaj sa sljedećim prototipima:
int skinii (int *element, Stog *stog);
int dodaj (int element, Stog *stog);
Što će ispisati sljedeći program?

```
#include <stdio.h>
#define MAXSTOG 100
typedef struct {
    int vrh, polje[MAXSTOG];
} Stog;
void init_stog(Stog *stog) {
    stog->vrh = -1;
}
int main() {
    Stog stog;
    int a=1,b=2,c=3;
    init_stog(&stog);
    dodaj(a,&stog);
    dodaj(b,&stog);
    dodaj(c,&stog);
    skinii(&a,&stog);
    skinii(&c,&stog);
    skinii(&b,&stog);
    printf("%d %d %d",a,b,c);
}
```

Done

zadi - Paint Mozilla Firefox - Pofuk, Robert (00364... 9:43

AHycO

Pregledavanje rezultata provjere: ASP Blic 2

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10

Broj mogućih točnih odgovora: 1 Odgovor Točan odgovor

Mogući broj bodova: 0,50

Ostvareni broj bodova: 0,50

Koja od sljedećih nizova naredbi u pseudokodu će zamijeniti vrijednost varijabli A i B pomoću stoga:

a) stavi(A); stavi(Pom); stavi(B); stavi(Pom); skinii(A); stavi(Pom);skinii(B);
b) stavi(B); skinii(A);
c) stavi(A); skinii(B);
d) stavi(A); stavi(B); skinii(A); skinii(B);
e) stavi(A); skinii(B); stavi(B); skinii(A);

Kraj pregledavanja

Internet 100%

Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

AHYCO

Provjera: ASP Blic 2

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10 11 12

c e c e b

25 m 16 s

Ocenjivanje provjere

Broj mogućih točnih odgovora: 1

Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha i 0 u slučaju neuspjeha i ima prototip
int dodaj (int element, Stog *stog);
što će biti na stogu nakon obavljanja sljedeće naredbe, uz pretpostavku da je stog bio prazan, da na njega stane najmanje 3 elementa i da stog raste s lijeva na desno?
dodaj (dodaj (dodaj (5, &stog)+2, &stog), &stog);

- a) 5 3 4
- b) 5 3 1**
- c) 1 1 1
- d) 5 1 1
- e) 1 3 5

Done

start zad34 - Paint Glavni izbornik - Mozilla... Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

http://ahyco.fer.hr/Provjere/Provjera.aspx

AHYCO

Provjera: ASP Blic 2

Robert Pofuk Odjava

Redni broj pitanja 1 2 3 4 5 6 7 8 9 10 11 12

c e

27 m 59 s

Ocenjivanje provjere

Broj mogućih točnih odgovora: 1

Koja od sljedećih tvrdnji vezanih uz rekurziju JE istinita?

- a) rekurzija koristi isključivo globalne varijable za prijenos vrijednosti
- b) maksimalni broj poziva rekurzije obmuto je proporcionalan broju registara procesora
- c) ne mora postojati slučaj koji se rješava bez rekurzije
- d) rekurzija je istovremeni poziv dva ili više potprograma
- e) za pohranu podataka i povratak iz rekurzije koristi se stog**

Done

start zad1 - Paint Glavni izbornik - Mozilla... Pofuk, Robert (0036441381) (ASP Blic 2) - Mozilla Firefox

9:44

Ako funkcija stavljanja na stog vraća 1 u slučaju uspjeha a 0 u slučaju neuspjeha i ima prototip

```
int dodaj (int element, Stog *stog);
```

a funkcija skidanja sa stoga vraća vrijednost elementa s vrha ili -1 ako je stog prazan i ima prototip

```
int skinji (int *element, Stog *stog);
```

što će biti na stogu nakon obavljanja sljedećih naredbi, uz pretpostavku da je stog bio prazan i da stog raste s lijeva na desno:

```
dodaj (5, &stog);
dodaj (dodaj(skinji(&element, &stog), &stog), &stog);
```

a)	5
b)	stog će biti prazan
c)	5 0
d)	-1 1
e)	5 1

Done

start zad5 - Paint Glavni izbornik - Mozilla Firefox Pofuk, Robert (00364... HR 9:48

Redni broj pitanja **2** Odgovor
 Broj mogućih točnih odgovora: 1 Točan odgovor
 Mogući broj bodova: 0,50
 Ostvareni broj bodova: -0,10

Kraj pregledavanja

Što će se ispisati funkcijom:

```
void ispis (cvor *korijen) {
    printf ("%c", korijen->element);
    if (korijen->lijevo && korijen->desno) {
        ispis (korijen->desno);
        ispis (korijen->lijevo);
    }
}
```

za stablo na slici pozivom funkcije
`ispis (korijen);`
 ako je korijen u trenutku poziva pokazivač na korijen stabla?

1 2 3 4 5 6 7 8 9 10
 a b e b a b e b e c
 a e e b a b e a e c

a)	521634
b)	134625
c)	123456
d)	12364
e)	13462

Redni broj pitanja

<

3

>

Odgovor
Točan odgovorBroj mogućih točnih odgovora: 1
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

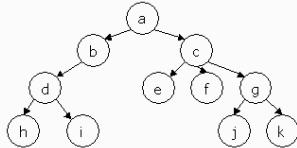
a b e b a b e b e c

a e e b a b e a e c



Kraj pregledavanja

Koja od sljedećih tvrdnji NIJE istinita?



Slika (odnosi se samo na dva ponuđena odgovora):

- a) Binarno stablo koje je visine k i ima $2^k - 1$ elemenata naziva se puno (*full*) binarno stablo
- b) Maksimalni broj čvorova binarnog stabla na k -toj razini jednak je 2^{k-1}
- c) Maksimalni broj čvorova binarnog stabla dubine k jednak je $2^k - 1$ za $k > 0$
- d) U stablu sa slike listovi su: (h, i, e, f, j, k)
- e)** Svi čvorovi sa stabla na slici su istog stupnja

Redni broj pitanja

<

4

>

Odgovor
Točan odgovorBroj mogućih točnih odgovora: 1
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

a b e b a b e b e c

a e e b a b e a e c



Kraj pregledavanja

Predložena je funkcija za pronađak čvora sa zadanim cijelobrojnom šifrom u jednostruko povezanoj linearnoj listi:

```
cvor *trazi(cvor *glava, int sifra){
    if (glava->sifra == sifra) return glava;
    if (glava->sljed)
        return trazi(glava->sljed, sifra);
    else
        return NULL;
}
```

Koja od sljedećih tvrdnji (koje se odnose na predloženu funkciju) je ispravna?

- a) Funkcija nije ispravna: pretražuje sve čvorove liste osim prvog!
- b)** Funkcija nije ispravna: ne radi na praznu listu (uzrokuje pogrešku kod poziva!)
- c) Funkcija nije ispravna: pretražuje sve čvorove liste osim zadnjeg!
- d) Funkcija nije ispravna: ukoliko čvor sa zadanim šifrom ne postoji u listi, funkcija neće "nikad" završiti
- e) Funkcija je ispravna i vraća pokazivač na čvor sa zadanim šifrom ili NULL ako čvor sa zadanim šifrom ne postoji u listi.

Redni broj pitanja

<

5

>

Odgovor
Točan odgovorBroj mogućih točnih odgovora: 1
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

a b e b a b e b e c

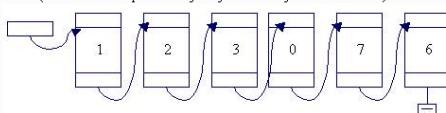
a e e b a b e a e c



Kraj pregledavanja

Što će vratiti prišljena funkcija za zadatu jednostruko povezanu linearnu listu:

Slika (unutar čvorova prikazana je vrijednost varijable element):



```
int f(cvor *glava){
    if (glava){
        if (glava->element > 3)
            return glava->element + f(glava->sljed);
        else
            return f(glava->sljed);
    }else{
        return 0;
    }
}
```

- a)** 13
- b) 0
- c) 6
- d) 19
- e) 16

Redni broj pitanja

<

6

>

Odgovor

Točan odgovor

Broj mogućih točnih odgovora: 1
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

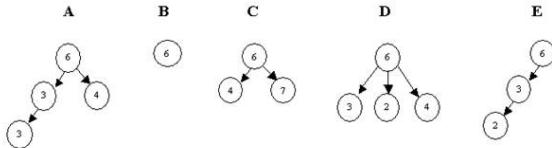
a b e b a b e b e c

a e e b a b e a e c

g g g g g g g g g g

Kraj pregledavanja

Koja od prikazanih stabala su gomile:



a) A

b) A, B

c) A, B, C, E

d) A, B, E

e) B

Redni broj pitanja

<

1

>

Odgovor

Točan odgovor

Broj mogućih točnih odgovora: 1
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

a b e b a b e b e c

a e e b a b e a e c

g g g g g g g g g g

Kraj pregledavanja

Koliko razina ima potpuno binarno stablo koje sadrži 100 čvorova i koliki je broj čvorova na posljednjoj razini?

a) broj razina=7 broj čvorova=37

b) broj razina =6 broj čvorova=50

c) broj razina =7 broj čvorova=50

d) broj razina =6 broj čvorova=64

e) broj razina =7 broj čvorova=64

Redni broj pitanja

<

7

>

Odgovor

Točan odgovor

Broj mogućih točnih odgovora: 1
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

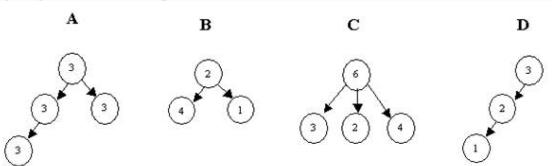
a b e b a b e b e c

a e e b a b e a e c

g g g g g g g g g g

Kraj pregledavanja

Koja od prikazanih stabala su gomile:



a) C

b) B

c) niti jedno od prikazanih stabala nije gomila

d) A, B, C, D

e) A

Redni broj pitanja **8** Odgovor
Broj mogućih točnih odgovora: 1 Točan
Mogući broj bodova: 0,50 odgovor
Ostvareni broj bodova: -0,10

Koji od ponuđenih ispisa gomile po razinama je ispravan ako je gomila formirana za ulazni niz 50 5 7 10 13 1 8 algoritmom čija je složenost za najgori slučaj $O(n \log n)$?

a)	50 13 8 5 10 1 7
b)	50 13 10 8 7 5 1
c)	50 13 7 5 10 1 8
d)	50 5 7 10 13 1 8
e)	50 13 8 10 7 5 1

Kraj pregledavanja

Redni broj pitanja **9** Odgovor
Broj mogućih točnih odgovora: 1 Točan
Mogući broj bodova: 0,50 odgovor
Ostvareni broj bodova: 0,50

Neka jednostruko povezana lista sadrži čvorove sa zapisima o osobama:

```
struct s {
    int sifraOsobe;
    int godinaStaza;
    float placaOsobe;
    struct s *sljed;
};

typedef struct s cvor;

Zadana je funkcija f:
float f(cvor *p, int g, int *br) {
    if (p) {
        if (p->godinaStaza >= g)
            ++(*br);
        return p->placaOsobe + f(p->sljed, g, br);
    }
    else
        return f(p->sljed, g, br);
} else {
    return 0.0;
}
}
```

Kakve vrijednosti sadrže varijable br i p nakon poziva funkcije f u sljedećem programskog odsječku:

```
...
cvor *glava; int br = 0;
...
// glava pokazuje na pocetak liste u trenutku poziva funkcije
p = f(glava, 10, &br);
...
```

- | | |
|-----------|---|
| a) | br = broj osoba u listi kojima je plaća veća ili jednaka 10, p = zbroj plaća svih osoba u listi kojima je plaća veća ili jednaka 10 |
| b) | br = broj osoba s 10 ili više godina radnog staža, p = prosjek plaća osoba s 10 ili više godina radnog staža |
| c) | br = broj osoba u listi, p = zbroj plaća svih osoba u listi |
| d) | br = broj osoba u listi, p = prosjek plaća svih osoba u listi |
| e) | br = broj osoba s 10 ili više godina radnog staža, p = zbroj plaća osoba s 10 ili više godina radnog staža |

Redni broj pitanja

< 10 >

Odgovor

Broj mogućih točnih odgovora: 1
Točan odgovor
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

a b e b a b e b e c

a e e b a b e a e c

○○○○○○○○○○○○○○

Kraj pregledavanja

Neka jednostruko povezana lista sadrži čvorove sa sljedećim tipom zapisa:

```
struct s {  
    int broj;  
    struct s *sljed;  
};  
typedef struct s cvor;  
Što radi funkcija f, ako je poziv funkcije f(glava, 7, &br)?  
cvor *f(cvor *p, int broj, int *br) {  
    if (p) {  
        ++(*br);  
        if (p->broj == broj) {  
            return p;  
        } else {  
            return f(p->sljed, broj, br);  
        }  
    } else {  
        return NULL;  
    }  
}
```

Napomena:

Varijabla glava je pokazivač na prvi čvor u listi, a varijabla br je deklarirana i inicijalizirana kao: int br = 0)

- a) Funkcija vraća pokazivač na početni čvor u listi, te broj čvorova u listi koji sadrži broj 7
- b) Funkcija vraća pokazivač na posljednji čvor u listi koji sadrži broj 7, te redni broj tog čvora u listi
- c) Funkcija vraća pokazivač na prvi čvor u listi koji sadrži broj 7, te redni broj tog čvora u listi
- d) Funkcija vraća pokazivač na prvi čvor u listi, te broj čvorova u listi koji sadrže broj 7
- e) Funkcija vraća pokazivač na početni čvor u listi, te redni broj čvora u listi koji sadrži broj 7

Redni broj pitanja

< 1 >

Odgovor

Broj mogućih točnih odgovora: 1
Točan odgovor
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

a b e b a b e b e c

a e e b a b e a e c

○○○○○○○○○○○○○○

Kraj pregledavanja

Što je razina koja sadrži 100 čvorova u binarnom stablu?

- a) broj razina=7 broj čvorova=37
- b) broj razina =6 broj čvorova=50
- c) broj razina =7 broj čvorova=50
- d) broj razina =6 broj čvorova=64
- e) broj razina =7 broj čvorova=64

Redni broj pitanja

< 2 >

Odgovor

Broj mogućih točnih odgovora: 1
Točan odgovor
Mogući broj bodova: 0,50
Ostvareni broj bodova: -0,10

1 2 3 4 5 6 7 8 9 10

a b e b a b e b e c

a e e b a b e a e c

○○○○○○○○○○○○○○

Kraj pregledavanja

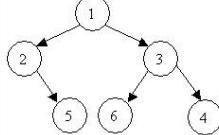
Što će se ispisati funkcijom:

```
void ispis (cvor *korijen) {  
    printf ("%c", korijen->element);  
    if (korijen->lijevo && korijen->desno) {  
        ispis (korijen->desno);  
        ispis (korijen->lijevo);  
    }  
}
```

za stablo na slici pozivom funkcije

ispis (korijen);

ako je korijen u trenutku poziva pokazivač na korijen stabla?

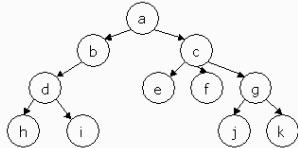


- a) 521634
- b) 134625
- c) 123456
- d) 12364
- e) 13462

Redni broj pitanja **3** Odgovor
 Broj mogućih točnih odgovora: 1 Točan odgovor
 Mogući broj bodova: 0,50 Osvuđeni broj bodova: 0,50

Kraj pregledavanja

Koja od sljedećih tvrdnji NIJE istinita?



Slika (odnosi se samo na dva ponuđena odgovora):

- a) Binarno stablo koje je visine k i ima $2^k - 1$ elemenata naziva se puno (*full*) binarno stablo
- b) Maksimalni broj čvorova binarnog stabla na k -toj razini jednak je 2^{k-1}
- c) Maksimalni broj čvorova binarnog stabla dubine k jednak je $2^k - 1$ za $k > 0$
- d) U stablu sa slike listovi su: (h, i, e, f, j, k)
- e)** Svi čvorovi sa stabla na slici su istog stupnja

Redni broj pitanja **4** Odgovor
 Broj mogućih točnih odgovora: 1 Točan odgovor
 Mogući broj bodova: 0,50 Osvuđeni broj bodova: 0,50

Kraj pregledavanja

Predložena je funkcija za pronađak čvora sa zadanom cijelobrojnom šifrom u jednostruko povezanoj linearnoj listi:

```

cvor *trazi(cvor *glava, int sifra){
    if (glava->sifra == sifra) return glava;
    if (glava->sljed)
        return trazi(glava->sljed, sifra);
    else
        return NULL;
}
  
```

Koja od sljedećih tvrdnji (koje se odnose na predloženu funkciju) je ispravna?

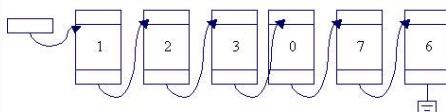
- a) Funkcija nije ispravna: pretražuje sve čvorove liste osim prvog!
- b)** Funkcija nije ispravna: ne radi na praznu listu (uzrokuje pogrešku kod poziva)
- c) Funkcija nije ispravna: pretražuje sve čvorove liste osim zadnjeg!
- d) Funkcija nije ispravna: ukoliko čvor sa zadanom šifrom ne postoji u listi, funkcija neće "nikad" završiti
- e) Funkcija je ispravna i vraća pokazivač na čvor sa zadanom šifrom ili NULL ako čvor sa zadanom šifrom ne postoji u listi.

Redni broj pitanja **5** Odgovor
 Broj mogućih točnih odgovora: 1 Točan odgovor
 Mogući broj bodova: 0,50 Osvuđeni broj bodova: 0,50

Kraj pregledavanja

Što će vratiti prišljena funkcija za zadatu jednostruko povezanu linearnu listu:

Slika (unutar čvorova prikazana je vrijednost varijable element):



```

int f(cvor *glava){
    if (glava){
        if (glava->element > 3)
            return glava->element + f(glava->sljed);
        else
            return f(glava->sljed);
    }else{
        return 0;
    }
}
  
```

- a)** 13
- b) 0
- c) 6
- d) 19
- e) 16

Redni broj pitanja

<

6

>

Odgovor

Točan
odgovor

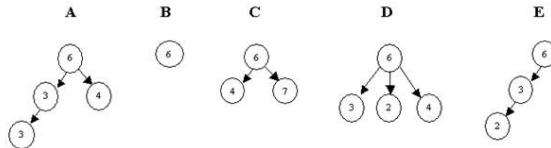
Broj mogućih točnih odgovora: 1

Mogući broj bodova: 0,50

Ostvareni broj bodova: 0,50

Kraj pregledavanja

Koja od prikazanih stabala su gomile:



a) A

D) A, B

c) A, B, C, E

d) A, B, E

e) B

Redni broj pitanja

<

7

>

Odgovor

Točan
odgovor

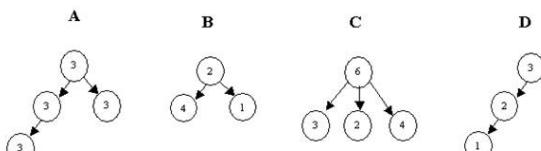
Broj mogućih točnih odgovora: 1

Mogući broj bodova: 0,50

Ostvareni broj bodova: 0,50

Kraj pregledavanja

Koja od prikazanih stabala su gomile:



a) C

b) B

c) niti jedno od prikazanih stabala nije gomila

d) A, B, C, D

e) A

Redni broj pitanja

<

8

>

Odgovor

Točan
odgovor

Broj mogućih točnih odgovora: 1

Mogući broj bodova: 0,50

Ostvareni broj bodova: -0,10

Kraj pregledavanja

Koji od ponuđenih ispisu gomile po razinama je ispravan ako je gomila formirana za ulazni niz 50 5 7 10 13 1 8 algoritmom čija je složenost za najgori slučaj $\mathcal{O}(n \log n)$?

a) 50
13 8
5 10 1 7

b) 50
13 10
8 7 5 1

c) 50
13 7
5 10 1 8

d) 50
5 7
10 13 1 8

e) 50
13 8
10 7 5 1

Redni broj pitanja

<

9

>

Odgovor
Točan odgovor

Broj mogućih točnih odgovora: 1
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

a b e b a b e b e c

a e e b a b e a e c



Kraj pregledavanja

Neka jednostruko povezana lista sadrži čvorove sa zapisima o osobama:

```
struct s {
    int sifraOsobe;
    int godinaStaza;
    float placaOsobe;
    struct s *sljed;
};

typedef struct s cvor;

Zadana je funkcija f:
float f(cvor *p, int g, int *br) {
    if (p) {
        if(p->godinaStaza >= g) {
            ++(*br);
            return p->placaOsobe + f(p->sljed, g, br);
        }
        else {
            return f(p->sljed, g, br);
        }
    } else {
        return 0.;
    }
}
```

Kakve vrijednosti sadrže varijable br i p nakon poziva funkcije f u sljedećem programskog odsječku:

```
...
cvor *glava; int br = 0;
...
// glava pokazuje na pocetak liste u trenutku poziva funkcije
p = f(glava, 10, &br);
...
```

- a) $br = \text{broj osoba u listi kojima je plaća veća ili jednaka } 10, p = \text{zbroj plaća svih osoba u listi kojima je plaća veća ili jednaka } 10$
- b) $br = \text{broj osoba s } 10 \text{ ili više godina radnog staža, } p = \text{prosjek plaća osoba s } 10 \text{ ili više godina radnog staža}$
- c) $br = \text{broj osoba u listi, } p = \text{zbroj plaća svih osoba u listi}$
- d) $br = \text{broj osoba u listi, } p = \text{prosjek plaća svih osoba u listi}$
- e)** $br = \text{broj osoba s } 10 \text{ ili više godina radnog staža, } p = \text{zbroj plaća osoba s } 10 \text{ ili više godina radnog staža}$

Redni broj pitanja

<

10

>

Odgovor
Točan odgovor

Broj mogućih točnih odgovora: 1
Mogući broj bodova: 0,50
Ostvareni broj bodova: 0,50

1 2 3 4 5 6 7 8 9 10

a b e b a b e b e c

a e e b a b e a e c



Kraj pregledavanja

Neka jednostruko povezana lista sadrži čvorove sa sljedećim tipom zapisa:

```
struct s {
    int broj;
    struct s *sljed;
};

typedef struct s cvor;

Što radi funkcija f, ako je poziv funkcije f(glava, 7, &br)?
cvor *f(cvor *p, int broj, int *br) {
    if (p) {
        ++(*br);
        if (p->broj == broj) {
            return p;
        } else {
            return f(p->sljed, broj, br);
        }
    } else {
        return NULL;
    }
}
```

Napomena:

Varijabla glava je pokazivač na prvi čvor u listi, a varijabla br je deklarirana i inicijalizirana kao `int br = 0`)

- a) Funkcija vraća pokazivač na početni čvor u listi, te broj čvorova u listi koji sadrži broj 7
- b) Funkcija vraća pokazivač na posljednji čvor u listi koji sadrži broj 7, te redni broj tog čvora u listi
- c)** Funkcija vraća pokazivač na prvi čvor u listi koji sadrži broj 7, te redni broj tog čvora u listi
- d) Funkcija vraća pokazivač na prvi čvor u listi, te broj čvorova u listi koji sadrže broj 7
- e) Funkcija vraća pokazivač na početni čvor u listi, te redni broj čvora u listi koji sadrži broj 7