PITANJA ZA II KOLOKVIJUM

FUNKCIJE

FUNKCIJE

1. Funkcije u jeziku C++ mogu se rekurzivno pozivati.

- **Tacno**
- **Netacno**

FUNKCIJE

1. Funkcije u jeziku C++ mogu se rekurzivno pozivati.

- **Tacno**
- **Netacno**

FUNKCIJE

2. Sluzbena rec inline:

- **ugraduje telo funkcije neposredno u kod na mestu pozivanja**
- **ubacuje citav objekat u jedan red**
- **izvrsava kod jednog reda**

FUNKCIJE

2. Sluzbena rec inline:

- **ugraduje telo funkcije neposredno u kod na mestu pozivanja**
- **ubacuje citav objekat u jedan red**
- **izvrsava kod jednog reda**

FUNKCIJE

3. Funkcija atoi():

- **konvertuje numericki string u ceo broj**
- **konvertuje ceo broj u string**
- **izracunava arc tg (arkus tangens) promenljive i**

FUNKCIJE

3. Funkcija atoi():

- **konvertuje numericki string u ceo broj**
- **konvertuje ceo broj u string**
- **izracunava arc tg (arkus tangens) promenljive i**

FUNKCIJE

4. Šta vraca funkcija tipa void?

- **Ne vraca nista**
- **Vrednost tipa void**
- **Vrednost prosledenih parametara**

FUNKCIJE

4. Šta vraca funkcija tipa void?

Ne vraca nista

- **Vrednost tipa void**
- **Vrednost prosledenih parametara**

FUNKCIJE

5. Program main moze da ima parametre.

- **Tacno**
- **Netacno**

FUNKCIJE

5. Program main moze da ima parametre.

- **Tacno**
- **Netacno**

FUNKCIJE

6. Ako je data funkcija double nadji(int x, int y, bool b) { ... }

- **ime funkcije je nadji**
- **povratna vrednost je int**
- **povratna vrednost je bool**
- **broj parametara funkcije je tri**

FUNKCIJE

6. Ako je data funkcija double nadji(int x, int y, bool b) { ... }

- **ime funkcije je nadji**
- **povratna vrednost je int**
- **povratna vrednost je bool**
- **broj parametara funkcije je tri**

FUNKCIJE

7. U jeziku C++ moguce je imati razlicite funkcije istog imena koje rade sa razlicitim brojem ili tipovima argumenata. To se zove _____ funkcija.

- **preklapanje (overloading)**
- **redefinisanje (overriding)**

FUNKCIJE

7. U jeziku C++ moguce je imati razlicite funkcije istog imena koje rade sa razlicitim brojem ili tipovima argumenata. To se zove _____ funkcija.

- **preklapanje (overloading)**
- **redefinisanje (overriding)**

FUNKCIJE

8. Lokalne promenljive su tipicno “automatske”:

- **gube vrednost nakon izvrsenja funkcije, kreiraju se prilikom svakog novog poziva**
- **ne gube vrednost nakon izvrsenja funkcije**

FUNKCIJE

8. Lokalne promenljive su tipicno “automatske”:

- **gube vrednost nakon izvrsenja funkcije, kreiraju se prilikom svakog novog poziva**
- **ne gube vrednost nakon izvrsenja funkcije**

FUNKCIJE

9. Lokalne promenljive su dostupne:

- **samo jednom delu programa**
- **celom programu**

FUNKCIJE

9. Lokalne promenljive su dostupne:

- **samo jednom delu programa**
- **celom programu**

FUNKCIJE

10. Promenljiva definisana u funkciji ima:

- **lokalni doseg (local scope)**
- **globalni doseg (global scope)**

FUNKCIJE

10. Promenljiva definisana u funkciji ima:

- **lokalni doseg (local scope)**
- **globalni doseg (global scope)**

FUNKCIJE

11. Blok koda odvaja se zagradama:

- **()**
- **[]**
- **{}**

FUNKCIJE

11. Blok koda odvaja se zagradama:

- ()
- []
- {}

FUNKCIJE

12. Ako funkcija ne vraća vrednost, koja ključna reč se može koristiti kao povratni tip (engl. return type)?

- **void**
- **int**
- **double**
- **float**
- **unsigned short**

FUNKCIJE

12. Ako funkcija ne vraća vrednost, koja ključna reč se može koristiti kao povratni tip (engl. return type)?

- **void**
- **int**
- **double**
- **float**
- **unsigned short**

FUNKCIJE

13. Zaglavlje funkcije main u C++ jeziku je:

- **Main(String[] args)**
- **Main(String args[])**
- **void main(String[] args)**
- **main(String[] args)**
- **int main()**

FUNKCIJE

13. Zaglavlje funkcije main u C++ jeziku je:

- **Main(String[] args)**
- **Main(String args[])**
- **void main(String[] args)**
- **main(String[] args)**
- **int main()**

FUNKCIJE

14. Argumenti funkcije se uvek nalaze unutar:

- **[ ] zagrada**
- **( ) zagrada**
- **{ } zagrada**
- **“ “ znakova**

FUNKCIJE

14. Argumenti funkcije se uvek nalaze unutar:

- **[ ] zagrada**
- **( ) zagrada**
- **{ } zagrada**
- **“ “ znakova**

FUNKCIJE

15. Šta je rezultat izvršenja datog kôda?

```
int f()
{
return 1;
}

int main()
{
cout << f() << endl;
return 0;
}
```

- **0**
- **1**
- **ništa**
- **1 0**
- **0 1**

FUNKCIJE

15. Šta je rezultat izvršenja datog kôda?

```
int f()
{
return 1;
}

int main()
{
cout << f() << endl;
return 0;
}
```

- **0**
- **1**
- **ništa**
- **1 0**
- **0 1**

FUNKCIJE

16. Dati rezultat izvršenja sledećeg koda.

```
void f()
{
cout << 1 << endl;
}

int main()
{
f();
return 0;
}
```

- **0**
- **1**
- **ništa**
- **1 0**
- **0 1**

FUNKCIJE

16. Dati rezultat izvršenja sledećeg koda.

```
void f()
{
cout << 1 << endl;
}

int main()
{
f();
return 0;
}
```

- **0**
- **1**
- **ništa**
- **1 0**
- **0 1**

FUNKCIJE

17. Svaki put kada se funkcija poziva, sistem skladišti parametre i lokalne promenljive u oblasti memorije, poznatu kao:

- **gomila (engl. heap)**
- **prostor za skladištenje**
- **stek**
- **niz**
- **nijedan od datih odgovora**

FUNKCIJE

17. Svaki put kada se funkcija poziva, sistem skladišti parametre i lokalne promenljive u oblasti memorije, poznatu kao:

- **gomila (engl. heap)**
- **prostor za skladištenje**
-  **stek**
- **niz**
- **nijedan od datih odgovora**

FUNKCIJE

18. Koje se sledeće funkcije mogu definisati kao void?

- **štampa kalendar za mesec dana za dati mesec i godinu.**
- **vraća kvadratni koren nekog broja.**
- **vraća proviziju prodaje, za dati iznos prodaje i stopu provizije.**
- **vraća vrednost bool pokazujući da li je broj paran**
- **štampa određeni karakter nekoliko puta**

FUNKCIJE

18. Koje se sledeće funkcije mogu definisati kao void?

- **štampa kalendar za mesec dana za dati mesec i godinu.**
- **vraća kvadratni koren nekog broja.**
- **vraća proviziju prodaje, za dati iznos prodaje i stopu provizije.**
- **vraća vrednost bool pokazujući da li je broj paran**
- **štampa određeni karakter nekoliko puta**

FUNKCIJE

19. Dati povratni tip metoda printGrade:

```
#include <iostream>
using namespace std;
// štampati ocenu za dati broj poena
_______ printGrade(double score)
{
if (score >= 90.0)
cout << 'A';
else if (score >= 80.0)
cout << 'B';
else if (score >= 70.0)
cout << 'C';
else if (score >= 60.0)
cout << 'D';
else
cout << 'F';
}
```

```
int main()
{
cout << "Enter a score: ";
double score;
cin >> score;
cout << "The grade is ";
printGrade(score);
return 0;
}
```

- **int**
- **double**
- **bool**
- **char**
- **void**

FUNKCIJE

19. Dati povratni tip metoda printGrade:

```
#include <iostream>
using namespace std;
// štampati ocenu za dati broj poena
_______ printGrade(double score)
{
if (score >= 90.0)
cout << 'A';
else if (score >= 80.0)
cout << 'B';
else if (score >= 70.0)
cout << 'C';
else if (score >= 60.0)
cout << 'D';
else
cout << 'F';
}
```

```
int main()
{
cout << "Enter a score: ";
double score;
cin >> score;
cout << "The grade is ";
printGrade(score);
return 0;
}
```

- **int**
- **double**
- **bool**
- **char**
- **void**

FUNKCIJE

20. Dati povratni tip metoda printGrade:

```
#include <iostream>
using namespace std;

_________ getGrade(double score)
{
if (score >= 90.0)
return 'A';
else if (score >= 80.0)
return 'B';
else if (score >= 70.0)
return 'C';
else if (score >= 60.0)
return 'D';
else
return 'F';
}
```

```
int main()
{
cout << "Enter a score: ";
double score;
cin >> score;

cout << "The grade is ";
cout << getGrade(score) << endl;
return 0;
}
```

- **int**
- **double**
- **bool**
- **char**
- **void**

FUNKCIJE

20. Dati povratni tip metoda printGrade:

```
#include <iostream>
using namespace std;

_________ getGrade(double score)
{
if (score >= 90.0)
return 'A';
else if (score >= 80.0)
return 'B';
else if (score >= 70.0)
return 'C';
else if (score >= 60.0)
return 'D';
else
return 'F';
}
```

```
int main()
{
cout << "Enter a score: ";
double score;
cin >> score;

cout << "The grade is ";
cout << getGrade(score) << endl;
return 0;
}
```

- **int**
- **double**
- **bool**
- **char**
- **void**

FUNKCIJE

21. Analizirati sledeći kôd:

```
#include <iostream>
using namespace std;

int f(int number)
{
// kôd koji nedostaje
}

int main()
{
cout << f(5) << endl;
return 0;
}
```

- **return "number";**
- **cout << number << endl;**
- **cout << "number" << endl;**
- **return number;**

Dati kôd koji nedostaje:

FUNKCIJE

21. Analizirati sledeći kôd:

```
#include <iostream>
using namespace std;

int f(int number)
{
// kôd koji nedostaje
}

int main()
{
cout << f(5) << endl;
return 0;
}
```

Dati kôd koji nedostaje:

- **return "number";**
- **cout << number << endl;**
- **cout << "number" << endl;**
-  **return number;**

FUNKCIJE

22. Kad se pozove funkcija sa parametrom, vrednost argumenta se prenosi na parametar. To se zove:

- **poziv funkcije**
- **poziv po vrednosti**
- **poziv po referenci**
- **poziv po imenu**
- **nijedan od ponuđenih odgovora**

FUNKCIJE

22. Kad se pozove funkcija sa parametrom, vrednost argumenta se prenosi na parametar. To se zove:

- **poziv funkcije**
- **poziv po vrednosti**
- **poziv po referenci**
- **poziv po imenu**
- **nijedan od ponuđenih odgovora**

FUNKCIJE

23. Dati rezultat izvršavanja sledećeg kôda:

```
#include <iostream>
using namespace std;

void maxValue(int value1, int value2, int max)
{
if (value1 > value2)
max = value1;
else
max = value2;
}

int main()
{
int max = 0;
maxValue(1, 2, max);
cout << "max is " << max << endl;
return 0;
}
```

- **max is 0**
- **max is 1**
- **max is 2**
- **max je nedefinisan**

FUNKCIJE

23. Dati rezultat izvršavanja sledećeg kôda:

```
#include <iostream>
using namespace std;

void maxValue(int value1, int value2, int max)
{
if (value1 > value2)
max = value1;
else
max = value2;
}

int main()
{
int max = 0;
maxValue(1, 2, max);
cout << "max is " << max << endl;
return 0;
}
```

- **max is 0**
- **max is 1**
- **max is 2**
- **max je nedefinisan**

FUNKCIJE

24. Analizirati dati kôd:

```
#include <iostream>
using namespace std;

int xfunction(int n, long t)
{
cout << "int";
return n;
}

long xfunction(long n)
{
cout << "long";
return n;
}

int main()
{
cout << xfunction(5);
return 0;
}
```

- **program prikazuje int i zatim 5.**
- **program prikazuje long i zatim 5.**
- **program se korektno izvršava, ali ne pokazuje ništa.**
- **program se ne kompilira zato što kompajler ne može da odredi koju funkciju treba pozvati.**

FUNKCIJE

24. Analizirati dati kôd:

```
#include <iostream>
using namespace std;

int xfunction(int n, long t)
{
cout << "int";
return n;
}

long xfunction(long n)
{
cout << "long";
return n;
}

int main()
{
cout << xfunction(5);
return 0;
}
```

- **program prikazuje int i zatim 5.**
- **program prikazuje long i zatim 5.**
- **program se korektno izvršava, ali ne pokazuje ništa.**
- **program se ne kompilira zato što kompajler ne može da odredi koju funkciju treba pozvati.**

FUNKCIJE

25. Analizirati dati kôd:

```
#include <iostream>
using namespace std;

int m(int num)
{
return num;
}
void m(int num)
{
cout << num;
}

int main()
{
cout << m(2);
return 0;
}
```

- **program generiše grešku kompilacije zato što dve funcije imaju isti potpis (engl. signature)**
- **program generiše grešku kompilacije zato što je druga po redu m funkcija definisana, ali nikad nije pozvana iz main() funkcije**
- **program se izvršava i štampa jednom 2**
- **program se izvršava i štampa dva puta 2**

FUNKCIJE

25. Analizirati dati kôd:

```
#include <iostream>
using namespace std;

int m(int num)
{
return num;
}
void m(int num)
{
cout << num;
}

int main()
{
cout << m(2);
return 0;
}
```

- **program generiše grešku kompilacije zato što dve funcije imaju isti potpis (engl. signature)**
- **program generiše grešku kompilacije zato što je druga po redu m funkcija definisana, ali nikad nije pozvana iz main() funkcije**
- **program se izvršava i štampa jednom 2**
- **program se izvršava i štampa dva puta 2**

FUNKCIJE

26. Koji od sledećih su validni prototipovi funkcija za funkciju koja vraća max vrednost kada su data dva broja?

- **int max(int num1, int num2);**
- **int max(int num1, int num2)**
- **int max(int, int);**
- **int max(num1, num2);**
- **max(int num1, int num2);**

FUNKCIJE

26. Koji od sledećih su validni prototipovi funkcija za funkciju koja vraća max vrednost kada su data dva broja?

- **int max(int num1, int num2);**
- **int max(int num1, int num2)**
- **int max(int, int);**
- **int max(num1, num2);**
- **max(int num1, int num2);**

FUNKCIJE

27. Koje su sledeće deklaracije funkcije netačne?

- **void t1(int x, int y = 0, int z);**
- **void t2(int x = 0, int y = 0, int z);**
- **void t3(int x, int y = 0, int z = 0);**
- **void t4(int x = 0, int y = 0, int z = 0);**

FUNKCIJE

27. Koje su sledeće deklaracije funkcije netačne?

- **void t1(int x, int y = 0, int z);**
- **void t2(int x = 0, int y = 0, int z);**
- **void t3(int x, int y = 0, int z = 0);**
- **void t4(int x = 0, int y = 0, int z = 0);**

FUNKCIJE

28. Dati rezultat izvršavanja sledećeg kôda?

```
inline void print(int i)
{
cout << i << endl;
}

int main()
{
print(1);
return 0;
}
```

- **0**
- **1**
- **2**
- **10**
- **ništa**

FUNKCIJE

28. Dati rezultat izvršavanja sledećeg kôda?

```
inline void print(int i)
{
cout << i << endl;
}

int main()
{
print(1);
return 0;
}
```

- **0**
- **1**
- **2**
- **10**
- **ništa**

FUNKCIJE

29. Varijabla definisana unutar funkcije se zove _____.

- **globalna varijabla**
- **varijabla funkcije**
- **varijabla bloka**
- **lokalna varijabla**

FUNKCIJE

29. Varijabla definisana unutar funkcije se zove _____.

- **globalna varijabla**
- **varijabla funkcije**
- **varijabla bloka**
- **lokalna varijabla**

FUNKCIJE

30. Dati vrednost k posle izvršavanja sledećeg bloka?

```
{
int k = 2;
}
```

- **0**
- **1**
- **2**
- **k nije definisan van bloka**

FUNKCIJE

30. Dati vrednost k posle izvršavanja sledećeg bloka?

```
{
int k = 2;
}
```

- **0**
- **1**
- **2**
- **k nije definisan van bloka**

FUNKCIJE

31. Dati rezultat izvršavanja datog kôda.

```
#include <iostream>
using namespace std;

int j = 1;
int main()
{
int i = 2;
cout << "i = " << i << " j = " << j << endl;
return 0;
}
```

- **i = 2 j = 1**
- **i = 1 j = 1**
- **i = 2 j = 2**
- **i = 1 j = 2**

FUNKCIJE

31. Dati rezultat izvršavanja datog kôda.

```
#include <iostream>
using namespace std;

int j = 1;
int main()
{
int i = 2;
cout << "i = " << i << " j = " << j << endl;
return 0;
}
```

- **i = 2 j = 1**
- **i = 1 j = 1**
- **i = 2 j = 2**
- **i = 1 j = 2**

FUNKCIJE

32. Dati rezultat izvršavanja datog kôda.

```
#include <iostream>
using namespace std;

int j = 1;

int main()
{
int i = 2;
int j = 2;
cout << "i = " << i << " j = " << j << endl;
return 0;
}
```

- **i = 2 j = 1**
- **i = 1 j = 1**
- **i = 1 j = 2**
- **i = 2 j = 2**

FUNKCIJE

32. Dati rezultat izvršavanja datog kôda.

```
#include <iostream>
using namespace std;

int j = 1;

int main()
{
int i = 2;
int j = 2;
cout << "i = " << i << " j = " << j << endl;
return 0;
}
```

- **i = 2 j = 1**
- **i = 1 j = 1**
- **i = 1 j = 2**
-  **i = 2 j = 2**

FUNKCIJE

33. Sledeći program poziva p() tri puta.
Dati rezultat zadnjeg poziva p().

```
#include <iostream>
using namespace std;

int j = 40;
void p()
{
int i = 5;
static int j = 5;

i++;
j++;
cout << "i = " << i << " j = " << j << endl;
}

int main()
{
p();
p();
p();
return 0;
}
```

- **i = 6 j = 6**
- **i = 6 j = 7**
- **i = 6 j = 8**
- **i = 6 j = 9**

FUNKCIJE

33. Sledeći program poziva p() tri puta.
Dati rezultat zadnjeg poziva p().

```
#include <iostream>
using namespace std;

int j = 40;
void p()
{
int i = 5;
static int j = 5;

i++;
j++;
cout << "i = " << i << " j = " << j << endl;
}

int main()
{
p();
p();
p();
return 0;
}
```

- **i = 6 j = 6**
- **i = 6 j = 7**
- **i = 6 j = 8**
- **i = 6 j = 9**

FUNKCIJE

34. Ako je parametar varijabla reference, ovaj parametar postaje drugo ime (alias) za originalnu varijablu. To se zove _____.

- **poziv funkcije**
- **prenos vrednošću (engl. pass by value)**
- **prenos referencom (engl. pass by reference)**
- **prenos po imenu**

FUNKCIJE

34. Ako je parametar varijabla reference, ovaj parametar postaje drugo ime (alias) za originalnu varijablu. To se zove _____.

- **poziv funkcije**
- **prenos vrednošću (engl. pass by value)**
-  **prenos referencom (engl. pass by reference)**
- **prenos po imenu**

FUNKCIJE

35. Dati rezultat izvršavanja sledećeg kôda (tema je reference).

```
#include <iostream>
using namespace std;

void f(int &p1, int p2)
{
p1++;
p2++;
}
```

```
int main()
{
int x1 = 1;
int x2 = 1;
f(x1, x2);
cout << "x1 = " << x1 << " x2 = " << x2;
return 0;
}
```

- **x1 = 1 x2 = 1**
- **x1 = 2 x2 = 2**
- **x1 = 1 x2 = 2**
- **x1 = 2 x2 = 1**

FUNKCIJE

35. Dati rezultat izvršavanja sledećeg kôda (tema je reference).

```
#include <iostream>
using namespace std;

void f(int &p1, int p2)
{
p1++;
p2++;
}
```

```
int main()
{
int x1 = 1;
int x2 = 1;
f(x1, x2);
cout << "x1 = " << x1 << " x2 = " << x2;
return 0;
}
```

- **x1 = 1 x2 = 1**
- **x1 = 2 x2 = 2**
- **x1 = 1 x2 = 2**
- **x1 = 2 x2 = 1**

FUNKCIJE

36. Neka je dato:

```
void nPrint(char ch, int n)
{
while (n > 0)
{
cout << ch;
n--;
}
}
```

- **aaaaa**
- **aaaa**
- **aaa**
- **nevažeći poziv**

Dati rezultat poziva nPrint ('a', 4)?

FUNKCIJE

36. Neka je dato:

```
void nPrint(char ch, int n)
{
while (n > 0)
{
cout << ch;
n--;
}
}
```

Dati rezultat poziva nPrint ('a', 4)?

- **aaaaa**
- **aaaa**
- **aaa**
- **nevažeći poziv**

FUNKCIJE

37. Šta je rezultat izvršavanja datog kôda (tema je reference)?

```
#include <iostream>
using namespace std;

void f(double &p)
{
p += 2;
}
```

```
int main()
{
double x = 1;
double y = 1;
f(x);
f(y);

cout << "x = " << x;
cout << " y = " << y << endl;
return 0;
}
```

- **x = 1 y = 1**
- **x = 2 y = 1**
- **x = 1 y = 2**
- **x = 2 y = 2**
- **x = 3 y = 3**

FUNKCIJE

37. Šta je rezultat izvršavanja datog kôda (tema je reference)?

```
#include <iostream>
using namespace std;

void f(double &p)
{
p += 2;
}
```

```
int main()
{
double x = 1;
double y = 1;
f(x);
f(y);

cout << "x = " << x;
cout << " y = " << y << endl;
return 0;
}
```

- **x = 1 y = 1**
- **x = 2 y = 1**
- **x = 1 y = 2**
- **x = 2 y = 2**
- **x = 3 y = 3**

FUNKCIJE

38. Analizirati sledeći kôd.

```
int max(const int &num1, int num2)
{
if (num1 > num2)
return num1;
else
return num2;
}
```

- **num1 je konstantni parametar i ne može biti promenjen**
- **num2 je parametar koji se prenosi po vrednosti (engl. pass-by-value parameter) i ne može biti promenjen**
- **num1 je je konstantni paramater i može biti promenjen u funkciji**
- **num2 je je parametar koji se prenosi po vrednosti i može biti promenjen u funkciji**

FUNKCIJE

38. Analizirati sledeći kôd.

```
int max(const int &num1, int num2)
{
if (num1 > num2)
return num1;
else
return num2;
}
```

- **num1 je konstantni parametar i ne može biti promenjen**
- **num2 je parametar koji se prenosi po vrednosti (engl. pass-by-value parameter) i ne može biti promenjen**
- **num1 je je konstantni paramater i može biti promenjen u funkciji**
- **num2 je je parametar koji se prenosi po vrednosti i može biti promenjen u funkciji**

FUNKCIJE

39. Pravilno deklarisana funkcija je:

- **main []{}**
- **main {}()**
- **main (){}**

FUNKCIJE

39. Pravilno deklarisana funkcija je:

- **main []{}**
- **main {}()**
-  **main (){}**

FUNKCIJE

40. Na koji način funkcija vraća vrednost na mesto odakle je pozvana?

- **Automatski**
- **Svojim imenom**
- **Preko parametara u zagradi**

FUNKCIJE

40. Na koji način funkcija vraća vrednost na mesto odakle je pozvana?

- **Automatski**
- 
- **Svojim imenom**
- **Preko parametara u zagradi**

FUNKCIJE

41. Funkcija koja se poziva iz programa main:

- **Mora biti deklarisana u programu main**
- **Mora biti deklarisana pre programa main**
- **Može biti deklarisana bilo gde u programu**

FUNKCIJE

41. Funkcija koja se poziva iz programa main:

- **Mora biti deklarisana u programu main**
-  **Mora biti deklarisana pre programa main**
- **Može biti deklarisana bilo gde u programu**

FUNKCIJE

42. Parametri se u funkciju mogu preneti:

- **Po vrednosti ili po adresi**
- **Po imenu, vrednosti ili adresi**
- **Po adresi, imenu ili referenci**

FUNKCIJE

42. Parametri se u funkciju mogu preneti:

Po vrednosti ili po adresi

- **Po imenu, vrednosti ili adresi**
- **Po adresi, imenu ili referenci**

FUNKCIJE

43. Definisanje različitih funkcija istog imena je:

- **Dozvoljeno**
- **Nije dozvoljeno**
- **Zavisi od prethodnog koda**

FUNKCIJE

43. Definisanje različitih funkcija istog imena je:

Dozvoljeno

- **Nije dozvoljeno**
- **Zavisi od prethodnog koda**

FUNKCIJE

44. Niz može da bude povratni tip funkcije.

- **Tačno**
- **Pogrešno**

FUNKCIJE

44. Niz može da bude povratni tip funkcije.

- **Tačno**
- **Pogrešno**

FUNKCIJE

45. Niz može da bude parametar funkcije.

- **Tačno**
- **Pogrešno**

FUNKCIJE

45. Niz može da bude parametar funkcije.

Tačno

- **Pogrešno**

FUNKCIJE

46. U jeziku C++ postoji funkcija koja ne mora da ima prototip.

- **Tačno**
- **Pogrešno**

FUNKCIJE

46. U jeziku C++ postoji funkcija koja ne mora da ima prototip.

Tačno

- **Pogrešno**

FUNKCIJE

47. Standardni način prenosa argumenata u funkciju je:

- **Po vrednosti**
- **Po adresi**

FUNKCIJE

47. Standardni način prenosa argumenata u funkciju je:

- **Po vrednosti**
- **Po adresi**

FUNKCIJE

48. Preklopljene funkcije mogu da se razlikuju samo po povratnom tipu.

- **Tačno**
- **Pogrešno**

FUNKCIJE

48. Preklopljene funkcije mogu da se razlikuju samo po povratnom tipu.

- **Tačno**
- **Pogrešno**

FUNKCIJE

49. Deklaracija funkcije void f(int a=10, int b); je ispravna.

- **Tačno**
- **Pogrešno**

FUNKCIJE

49. Deklaracija funkcije void f(int a=10, int b); je ispravna.

- **Tačno**
- **Pogrešno**

FUNKCIJE

50. Kada se funkciji argumenti prenose po vrednosti, ona može da ih izmeni.

- **Tačno**
- **Pogrešno**

FUNKCIJE

50. Kada se funkciji argumenti prenose po vrednosti, ona može da ih izmeni.

- **Tačno**
-  **Pogrešno**

FUNKCIJE

51. Kada se funkciji argumenti prenose po adresi, ona može da ih izmeni.

- **Tačno**
- **Pogrešno**

FUNKCIJE

51. Kada se funkciji argumenti prenose po adresi, ona može da ih izmeni.

Tačno

- **Pogrešno**

FUNKCIJE

52. Prototip funkcije sprečava njeno pozivanje sa neodgovarajućim brojem i(li) tipovima argumenata.

- **Tačno**
- **Pogrešno**

FUNKCIJE

52. Prototip funkcije sprečava njeno pozivanje sa neodgovarajućim brojem i(li) tipovima argumenata.

Tačno

- **Pogrešno**

FUNKCIJE

53. Naredba koja se koristi unutar tela funkcije za povratak iz funkcije i vraćanje rezultata je:

- **void**
- **return**
- **public**
- **static**

FUNKCIJE

53. Naredba koja se koristi unutar tela funkcije za povratak iz funkcije i vraćanje rezultata je:

- **void**
-  **return**
- **public**
- **static**

FUNKCIJE

54. Globalne promenljive su dostupne:

- **Samo jednom delu programa**
- **Celom programu**

FUNKCIJE

54. Globalne promenljive su dostupne:

- **Samo jednom delu programa**
-  **Celom programu**

FUNKCIJE

55. Opseg važenja globalne promenljive je:

- **Ceo program**
- **Funkcija main**
- **Ceo program posle mesta deklarisanja**

FUNKCIJE

55. Opseg važenja globalne promenljive je:

- **Ceo program**
- **Funkcija main**
-  **Ceo program posle mesta deklarisanja**

FUNKCIJE

56. Šta označava reč static ako se napiše ispred promenljive?

- **Promenljiva zadržava vrednost tokom izvršavanja programa**
- **Promenljiva se ne nalazi u memoriji**

FUNKCIJE

56. Šta označava reč static ako se napiše ispred promenljive?

Promenljiva zadržava vrednost tokom izvršavanja programa

- **Promenljiva se ne nalazi u memoriji**

FUNKCIJE

57. Oblast važenja lokalne promenljive je:

- **Ceo kod**
- **Blok u kojem je definisana**
- **Funkcija main**

FUNKCIJE

57. Oblast važenja lokalne promenljive je:

- Ceo kod
-  Blok u kojem je definisana
- Funkcija main

FUNKCIJE

58. Promenljiva koja je deklarisana unutar funkcije je:

- **Globalna**
- **Statička**
- **Lokalna**

FUNKCIJE

58. Promenljiva koja je deklarisana unutar funkcije je:

- **Globalna**
- **Statička**
-  **Lokalna**

FUNKCIJE

59. Statička globalna promenljiva je vidljiva samo u datoteci u kojoj je deklarisana.

- **Tačno**
- **Pogrešno**

FUNKCIJE

59. Statička globalna promenljiva je vidljiva samo u datoteci u kojoj je deklarisana.

Tačno

- **Pogrešno**

FUNKCIJE

60. Kada se poziva funkcija čiji je parametar referenca, ispred naziva stvarnog argumenta funkcije treba da bude &.

- **Tačno**
- **Pogrešno**

FUNKCIJE

60. Kada se poziva funkcija čiji je parametar referenca, ispred naziva stvarnog argumenta funkcije treba da bude &.

- **Tačno**

Pogrešno

FUNKCIJE

61. Funkcija nikako ne bi trebalo da vraća referencu na lokalnu promenljivu.

- **Tačno**
- **Pogrešno**

FUNKCIJE

61. Funkcija nikako ne bi trebalo da vraća referencu na lokalnu promenljivu.

Tačno

- **Pogrešno**

FUNKCIJE

62. Reference se mogu preusmeravati na druge objekte, isto kao pokazivači.

- **Tačno**
- **Pogrešno**

FUNKCIJE

62. Reference se mogu preusmeravati na druge objekte, isto kao pokazivači.

- **Tačno**

Pogrešno

FUNKCIJE

63. Korišćenje referenci olakšava prenos argumenata u funkciju:

- **Po vrednosti**
- **Po adresi**

FUNKCIJE

63. Korišćenje referenci olakšava prenos argumenata u funkciju:

- **Po vrednosti**

Po adresi

FUNKCIJE

64. Moguće je definisati niz referenci.

- **Tačno**
- **Pogrešno**

FUNKCIJE

64. Moguće je definisati niz referenci.

- **Tačno**
-  **Pogrešno**

FUNKCIJE

65. Ako je parametar funkcije konstantna referenca (const &), ona sme da promeni vrednost stvarnog argumenta sa kojim je pozvana.

- **Tačno**
- **Pogrešno**

FUNKCIJE

65. Ako je parametar funkcije konstantna referenca (const &), ona sme da promeni vrednost stvarnog argumenta sa kojim je pozvana.

- **Tačno**

Pogrešno

KRAJ!