

1. U paketu hr.fer.oopj definiran je razred R:

```
public class R {  
    private int x;  
    private int y;  
  
    public R (int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
  
    public String toString() { return x + ", " + y; }
```

Potom je u metodi main nekog drugog razreda napisano:

```
public class Main {  
    public static void main (String[] args) {  
        R[] r = new R[3];  
        for (int i = 0; i < r.length; i++) {  
            r[i] = new R(i, i);  
        }  
        r[1].x++;  
        System.out.println(r[0]);  
        System.out.println(r[1]);  
    }  
}
```

a) program će se srušiti tijekom izvođenja main

b) ispis 1,0 pa ispis 1,0

c) ispis 0,0 pa ispis 2,1

d) program se zbog pogrešaka u metodi main neće izvršiti

5. Ako u nekoj baznoj klasi postoji članska (nestatička) metoda `calculatePrice`, kolika mora biti metoda `calculatePrice` iz uvedene klase kojom se nadopunjava `calculatePrice` iz bazne klase?

a) Statička

b) void

☒ c) Članska (nestatička)

6. Što će se dogoditi kada neka klasa želi implementirati dva sučelja koja imaju metodu istog imena i istog broja argumenata?

a) izradit će se dvije iste metode s naznakom sučelja kojeg implementiraju

b) prevoditelj će javiti grešku jer klasa ne može implementirati oba sučelja

c) program se neće moći prevesti

☒ d) Izradit će se jedinstvena zajednička implementacija tih dviju metoda

7. Ako u klasi imamo sledeću deklaraciju atributa / članke varijable:

```
private String name;
```

te klasa imamo samo podrazumevani konstruktor koji je stvorio prevoditelj.

Na koju vrijednost će se inicijalizirati `name`:

a) na neku slučajnu vrijednost

b) 0

c) `"\0"`

d) `""`

☒ e) null

8. Ako u nekoj baznoj klasi postoji metoda `calculatePrice` vidljivosti `protected`, koje SVE vidljivosti može biti metoda `calculatePrice` iz uvedene klase kojom se nadopunjava `calculatePrice` iz bazne klase?

a) private

☒ b) public

c) ništa od navedenog

☒ d) protected

9. Ako u nekoj baznoj klasi postoji metoda `calculatePrice` vidljivosti `private`, koje SVE vidljivosti može biti metoda `calculatePrice` iz navedene klase kojom se nadjačava `calculatePrice` iz baze klase?

a) `public`

b) ništa od navedenog

c) `private`

d) `protected`

10. Za koja od označenih pridruživanja će prevoditelj prijaviti pogrešku?

```
interface I1 {}
```

```
interface I2 {}
```

```
class C1 implements I1 {}
```

```
class C2 extends C1 implements I2 {}
```

```
public class Main {
```

```
    public static void main (String [] args) {
```

```
        C1 obj1 = new C1 ();
```

```
        C2 obj2 = new C2 ();
```

```
        obj2 = obj1; // (1)
```

```
        obj1 = obj2; // (2)
```

```
        I1 a = obj1; // (3)
```

```
        I2 b = obj2; // (4)
```

```
        I3 c = obj1; // (5)
```

```
    }
```

```
}
```

a) `obj2 = obj1; (1)`

b) `I2 c = obj1; (5)`

11. Što će ispisati sledeći program:

```
public class R {  
    final int x = 0;  
    public R(int x) {  
        this.x = x;  
    }  
    public static void main (String[] args) {  
        R r1 = new R(1);  
        R r2 = new R(2);  
        System.out.println (r1.x);  
        System.out.println (r2.x);  
    }  
}
```

a) 12

12. Što će biti rezultat prevodenja i pokretanja programa?

```
interface A {  
    public int m();  
}  
class B implements A {  
    public int m() {  
        return 1;  
    }  
}  
class C extends B {  
    int m() {  
        return 2;  
    }  
}  
public class Main {  
    public static void main (String[] args) {  
        A ref1 = new C();  
        B ref2 = new B();  
        System.out.print (ref1.m());  
        System.out.println (ref2.m());  
    }  
}
```

a) Program se neće uspešno prevesti

13. U paketu hr.fer.oopj definiran je razred R kako sledi:

```
public class R {  
    public double x;  
    public double y;  
  
    public R() {  
        _____  
    }  
  
    public R(double x, double y) {  
        this.x = x;  
        this.y = y;  
    }  
}
```

Što će od ponuđenoga, napisano na mjestu označenom poduglavcima, osigurati da se prvim pretpostavljenog konstruktora u konačnici obavi inicijalizacija objekta postavljanjem članske varijable x na 3 te članske varijable y na 5?

d) this.x = 3; this.y = 5;

14. Hoće li se sledeci kod koji definira razred R provesti bez greske?

```
package hr.fer.oopj;
```

```
public class R {  
    private double x;  
    private double y;  
  
    public double getX() { return this.x; }  
    public double getY() { return y; }  
    public void update(double x, double y) {  
        this.x = x; y = y;  
    }  
}
```

b) Hoće, razred nema definiran konstruktor, ali će mu prevodilac automatski dodati podrazumijevani konstruktor.

15. Koji od navedenih redaka su ispravni?

```
interface Geometric {
```

```
    double PI = 3.14;
```

```
    double getArea();
```

```
}
```

```
public class Main {
```

```
    public static void main (String[] args) {
```

```
        Geometric g1; // (1)
```

```
        Geometric g2 = new Geometric; // (2)
```

```
        Geometric[] gArray1 = new Geometric [10]; // (3)
```

```
        Geometric[] gArray2; // (4)
```

```
        Object o = new Double (Geometric.PI); // (5)
```

```
    }
```

```
}
```

☒ a) (1)

☒ d) (4)

b) (2)

☒ e) (5)

☒ c) (3)

16. Što je od navedenog točno ako u programskom kodu piše

```
class A extends B
```

```
class C extends A
```

☒ a) Klasa A je izvedena iz klase B

f) Klasa B je izvedena iz klase A

b) Klasa B je bazna klasa za klasu C

c) Klasa C je bazna klasa za klasu B

d) Klasa B je izvedena iz klase A

☒ e) Klasa B je bazna klasa za klasu A

17. Izvedena klasa u istom paketu kao i bazna klasa može postupiti onim varijablama i metodama bazne klase koje su označene s

a) private

☒ b) public

☒ c) nemaju modifikator vidljivosti

☒ d) protected

17.b) izvedena klasa u RAZLIČITOM PAKETU

a) private

☒ b) public

c) nemaju m.v.

☒ d) protected

18. Što od navedenog vrijedi za apstraktnu metodu?

☒ a) a.m. nema tipelo

b) mod. vidljivosti a.m. mora biti public

☒ c) metoda koja je označava ključ. riječju abstract je apstraktna metoda

d) za a.m. ne treba navesti tip povratne vrijednosti

e) ako su sve metode u a. klasi apstraktne tada ih ne možemo označiti ključnom riječju abstract

19. Što od navedenog vrijedi za poziv apstraktne metode?

Konstruktor se može pozvati a) u bilo kojoj liniji konstruktora podklase

☒ b) u prvoj liniji konstruktora podklase

b) iz bilo koje metode podklase

d) ništa od navedenog

20. Želja programera je napisati programski kod razreda P i R, pri čemu razred R treba nasljeđivati razred P. Pri tome je pisao sljedeće.

```
public class P {  
    protected P() {  
    }  
}
```

```
public class R extends P { }
```

Koje su od sljedećih tvrdnji točne?

☒ a) Oba razreda imaju konstruktore, i zbog nasljeđivanja, konstruktor razreda R će najprije pozvati konstruktor razreda P

b) Razred R ne deklarira konstruktor pa se kod neće izvršiti

c) Razred P nikada ne nasljeđuje

☒ d) Razred P nasljeđuje razred java.lang.Object

Što će se dogoditi kada neka klasa želi implementirati dva sučelja koja imaju metodu istog imena i istog broja argumenata?

a Izradit će se dvije iste metode s naznakom sučelja kojeg implementiraju

b Prevoditelj će javiti grešku jer klasa ne može implementirati dva sučelja

c Program se neće moći prevesti

d Izradit će se jedinstvena zajednička implementacija tih dviju metoda

Ako u nekoj baznoj klasi postoji članska (nestatička) metoda calculatePrice, kakva mora biti metoda calculatePrice iz izvedene klase kojom se nadjačava calculatePrice iz bazne klase?

a Statička

b void

c Ništa od navedenog

d Članska (nestatička)

Hrana, piće i odjeća su artikli, pa se mogu objediniti u isto polje.

```
Item[] items = new Item[3];  
items[0] = new Beverage("23", "Coca cola", 10, 2);  
items[1] = new Food("777", "CaoCao", 2.5, LocalDate.of(2016,5, 11));  
items[2] = new Cloth("045", "Simple T-shirt", 350, 54);
```

Kako se ispravno poziva metoda getSize() klase Cloth?

a

System.out.println((Cloth.getSize(items[2]));

b

System.out.println(items[2].getSize());

c

System.out.println(((Cloth) items[2]).getSize());

d

System.out.println(((Cloth)items[0]).getSize());

Što će biti rezultat prevođenja i pokretanja programa?

```
interface A {  
    int m();  
}  
  
class B implements A {  
    public int m() {  
        return 1;  
    }  
}  
  
class C extends B {  
    public int m() {  
        return 2;  
    }  
}  
  
public class Main {  
    public static void main(String[] args) {  
        A ref1 = new C();  
        B ref2 = (B) ref1;  
        System.out.println(ref2.m());  
    }  
}
```

a

Program se neće uspješno prevesti.

b

Program će se prevesti i ispisat će 2 tijekom izvođenja.

c

Program će se prevesti i ispisat će 1 2 tijekom izvođenja.

d

Program će se prevesti i ispisat će 1 tijekom izvođenja.

e

Program će se prevesti, ali neće ispisati ništa i/ili će se srušiti tijekom izvođenja.

Što će ispisati sljedeći program:

```
public class Element {
    int i;

    public Element(int i) {
        this.i = i;
    }
    public Element(Element e) {
        this.i = e.i;
    }
}

public class R {
    Element e1;
    Element e2;

    public R(Element e1, Element e2){
        this.e1 = e1;
        this.e2 = new Element(e2);
    }

    public static void main(String[] args) {

        Element e1 = new Element(1);
        Element e2 = new Element(1);

        R r1 = new R(e1,e2);
        e1.i++;
        e2.i++;
        R r2 = new R(e1,e2);

        System.out.println(r1.e1.i + " " + r1.e2.i);
        System.out.println(r2.e1.i + " " + r2.e2.i);
    }
}
```



1 12 2



2 12 2



2 22 2



1 22 1

U paketu hr.fer.oopj definiran je razred R kako slijedi:

```
public class R {  
    private int x;  
    private int y;  
  
    public R(int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
  
    public String toString() { return x+", "+y; }  
}
```

Potom je u metodi main nekog drugog razreda napisano:

```
R[] r = new R[3];  
for(int i = 0; i < r.length; i++) {  
    r[i] = new R(i,i);  
}  
r[1].x++;  
System.out.println(r[0]);  
System.out.println(r[1]);
```

Što će biti rezultat pokretanja programa?

a program će se srušiti tijekom izvođenja metode main

b ispis 1,0 pa ispis 1,0

c ispis 0,0 pa ispis 2,1

d program se zbog pogrešaka u metodi main neće prevesti

Koji od navedenih redaka **nisu** ispravni i uzrokuju pogrešku pri prevođenju programa?

```
interface I1 { }  
class C1 implements I1 { }  
public class Main {  
    public static void main(String[] args) {  
        I1 a = new C1();           // (1)  
        C1 b = new C1();           // (2)  
        b = a;                     // (3)  
        a = b;                     // (4)  
        a = new I1();              // (5)  
    }  
}
```

a I1 a = new C1(); // (1)

b a = new I1(); // (5)

c C1 b = new C1(); // (2)

d b = a; // (3)

e a = b; // (4)

Koji od navedenih redaka **nisu** ispravni i uzrokuju pogrešku pri prevođenju programa?

```
interface I1 { }  
class C1 implements I1 { }  
public class Main {  
    public static void main(String[] args) {  
        I1 a = new C1();           // (1)  
        C1 b = new C1();           // (2)  
        b = a;                     // (3)  
        a = b;                     // (4)  
        a = new I1();              // (5)  
    }  
}
```

a

I1 a = new C1(); // (1)

b

a = new I1(); // (5)

c

C1 b = new C1(); // (2)

d

b = a; // (3)

e

a = b; // (4)

Što ispisuje sljedeći program?

```
package hr.fer.oop.lab2.example;
public class StrangeSum {
    public final int Ssum1(int x, int y) {
        return x + y + 10;
    }

    public int Ssum2(int x, int y) {
        return x + y + 10;
    }
}
```

```
package hr.fer.oop.lab2.example;
public class TwoNumbers extends StrangeSum {
    @Override
    public int Ssum2(int x, int y) {
        return super.Ssum2(x, y) + 15;
    }

    public static void main(String[] args) {
        TwoNumbers ss = new TwoNumbers();
        int result = ss.Ssum2(1, 1);
        System.out.println(result);
    }
}
```



27



17



Ništa od navedenog



12

S kojim od navedenih načina se može definirati apstraktna metode method1?

a

```
abstract method1();
```

b

```
public abstract void method1();
```

c

```
public abstract void method1;
```

d

```
abstract void method1(){};
```

e

```
public virtual void method1();
```

U paketu hr.fer.oopj definiran je razred R kako slijedi:

```
public class R {  
    public int x;  
    public int y;  
  
    public R(int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
  
    public String toString() { return x+", "+y; }  
}
```

Potom je u metodi main nekog drugog razreda napisano:

```
R r1 = new R(2,3);  
R r2 = new R(2,3);  
R r3 = new R(3,2);  
System.out.println(r1==r2);  
System.out.println(r1==r3);
```

Što će biti rezultat pokretanja programa?

a

ispis true pa ispis true

b

program se zbog pogrešaka u metodi main neće prevesti

c

ispis false pa ispis false

d

ispis true pa ispis false

Što ispisuje sljedeći program?

```
package hr.fer.oop.lab2.example;

public class StrangeSum {
    public final int Ssum1(int x, int y) {
        return x + y + 10;
    }
    public int Ssum2(int x, int y) {
        return x + y + 10;
    }
}
```

```
package hr.fer.oop.lab2.example;

public class TwoNumbers extends StrangeSum{
    @Override
    public int Ssum1(int x, int y) {
        return x + y + 15;
    }
    public static void main(String[] args) {
        TwoNumbers ss = new TwoNumbers();
        int result = ss.Ssum1(1, 1);
        System.out.println(result);
    }
}
```



12



Ništa od navedenog

Koliko je objekata, primjeraka klase B spremno za uništavanje (garbage collection)

```
public class B {  
    B next;  
    static B head;  
    public static void main(String[] args) {  
        B b = new B();  
        b.next = new B();  
        B.head = new B();  
        b = B.head; //Line 8  
        System.out.println("End");  
    }  
}
```

a 1

b 2

c 0

d 3

Što od navedenog vrijedi za apstraktnu klasu

a

ništa od navedenog

b

mora imati barem jednu apstraktnu metodu

c

primjerak potklase se može castati na tip ove klase

d

s konstruktorom se može kreirati primjerak (instanca) ove klase

Što radi metoda equals u klasi Object:



uspoređuje da reference pokazuju na isti objekt



uspoređuje da su reference spremljene u istu varijablu



uspoređuje da objekti imaju isti reprezentaciju sadržaja



uspoređuje da reference pokazuju na objekte koji imaju isti sadržaj



uspoređuje da su objekti spremljeni u istu varijablu

U paketu hr.fer.oopj definiran je razred R kako slijedi:

```
public class R {  
    public int x;  
    public int y;  
  
    public R(int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
  
    public String toString() { return x+", "+y; }  
}
```

Potom je u metodi main nekog drugog razreda napisano:

```
R r1 = new R(2,3);  
R r2 = new R(2,3);  
R r3 = new R(3,2);  
System.out.println(r1.equals(r2));  
System.out.println(r1.equals(r3));
```

Što će biti rezultat pokretanja programa?

a program se zbog pogrešaka u metodi main neće prevesti

b ispis true pa ispis false

c ispis true pa ispis true

d ispis false pa ispis false

U paketu hr.fer.oopj definiran je razred R kako slijedi:

```
public class R {  
    public int x;  
    public int y;  
  
    public R(int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
  
    public String toString() { return x+", "+y; }  
  
    public boolean equals(Object o) {  
        R other = (R)o;  
        return this.x==other.x;  
    }  
}
```

Potom je u metodi main nekog drugog razreda napisano:

```
R r1 = new R(2,3);  
R r2 = new R(2,3);  
R r3 = new R(2,4);  
System.out.println(r1.equals(r2));  
System.out.println(r1.equals(r3));
```

Što će biti rezultat pokretanja programa?

a

ispis true pa ispis false

b

ispis true pa ispis true

c

program se zbog pogrešaka u metodi main neće prevesti

d

ispis false pa ispis false

Što vrijedi za višestruko nasljeđivanje?

- a u Javi je moguće višestruko nasljeđivanje
- b višestruko nasljeđivanje je klasa implementira više sučelja
- c višestruko nasljeđivanje je kada neka klasa može naslijediti više klasa
- d u Javi nije moguće višestruko nasljeđivanje
- e višestruko nasljeđivanje je kada klasa naslijedi klasu koja je naslijedila neku drugu klasu

Želja programera je napisati programski kod razreda P i R, pri čemu razred R treba naslijediti razred P. Pri tome je napisao sljedeće.

```
public class P {  
}  
  
public class R inherits P {  
}
```

Koje su od sljedećih tvrdnji točne?

a

Deklaracija razreda P nije ispravna jer ako se koristi nasljeđivanje, onda treba eksplicitno napisati da on nasljeđuje razred `java.lang.Object`

b

Da bi razred R naslijedio razred P, umjesto `inherits` treba stajati `implements`

c

Oba razreda u byte-kodu imaju konstruktore

d

Da bi razred R naslijedio razred P, umjesto `inherits` treba stajati `extends`

e

Razredi P i R ne deklariraju konstruktore pa se kod zbog toga neće prevesti

Što je od navedenog točno ako u programskom kodu piše

```
class A extends B
```

a

B je podklasa nadklase A

b

Klasa B je bazna klasa za klasu A

c

Klasa A je bazna klasa za klasu B

d

Klasa A je izvedena iz klase B

e

Klasa B je izvedena iz klase A

S kojim od navedenih načina se može definirati apstraktna metode `method1`?

a

```
abstract method1();
```

b

```
public abstract void method1;
```

c

```
abstract void method1(){};
```

d

```
public abstract void method1();
```

e

```
public virtual void method1();
```