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

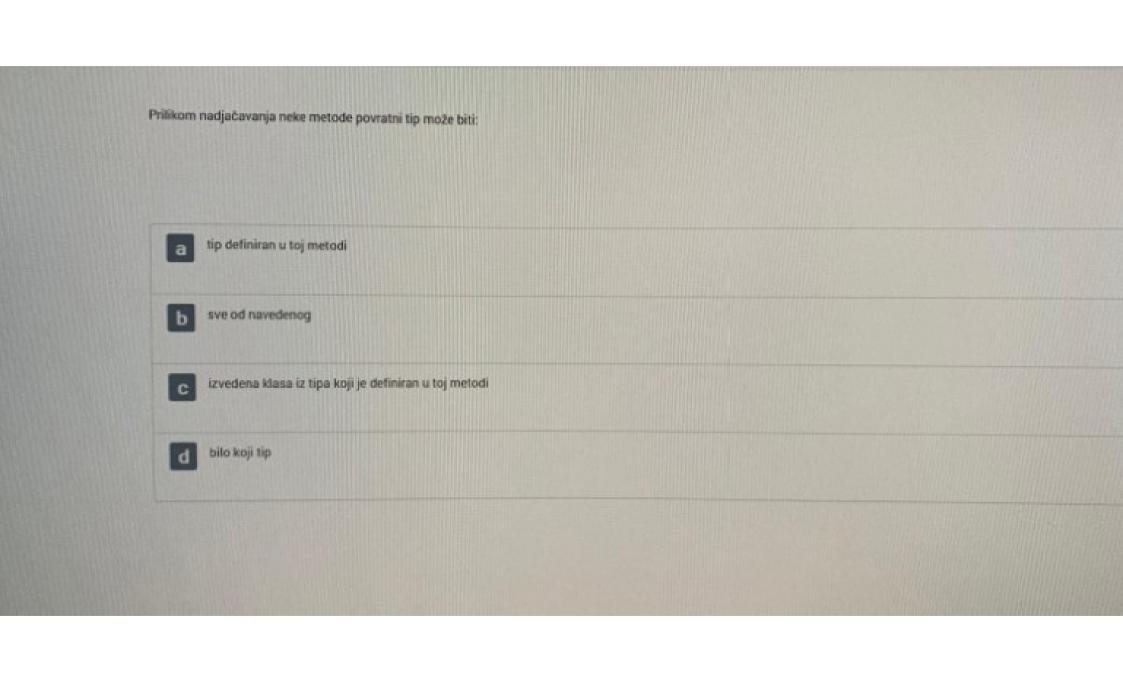
```
public class R {
  public double x;
  public double y;

private R() {
  }

  public R(double x, double y) {
    this.x - x;
    this.y - y;
  }
}
```

U metodi main smještenoj u razredu X u istom paketu želimo deklarirati lokalnu varijablu r i pridružiti joj jedan novostvoreni objekt razreda R. Koji će od ponuđenih izraza to učiniti?

- a Rr = new R;
- b R r new R(3.14, 6.28);
- R r malloc(sizeof(R));
- d R r = new R();



Koliko je objekata, primjeraka klase B spremno za uništavanje (garbage collection) u trenutku nakon što se izvrši naredba napisana u liniji 9?

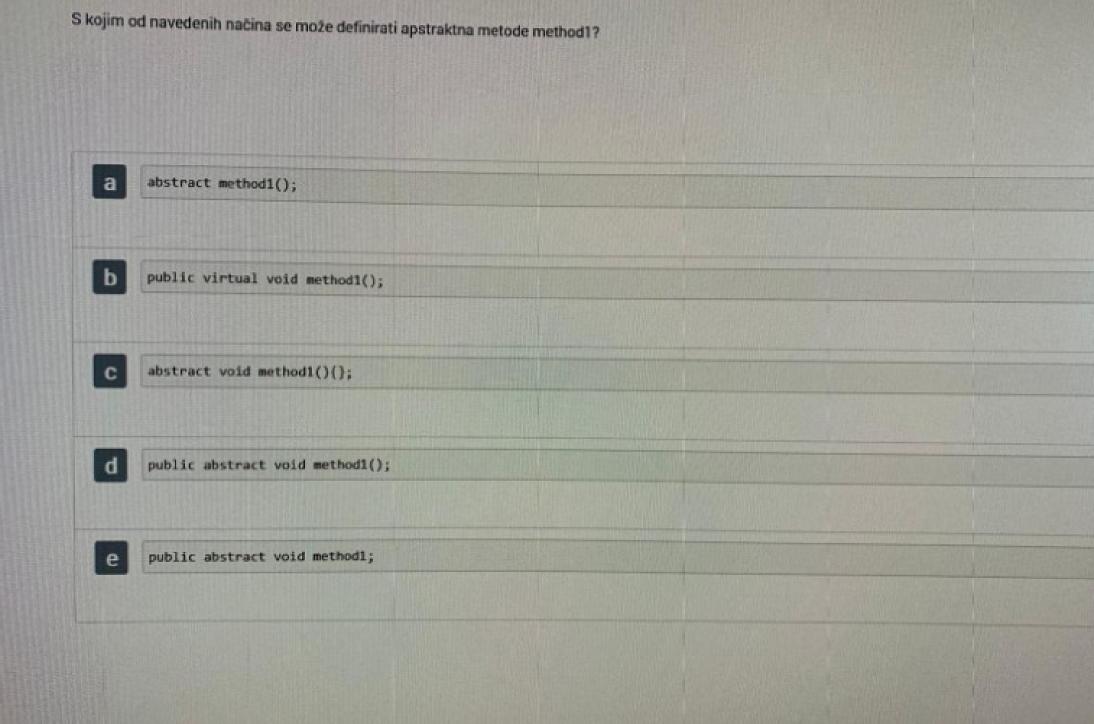
```
public class B {
    B next;
    static B head;
    public static void main(String[] args) {
        B b = new B();
        b.next - new B();
        B.head = new B();
        B pom = B.head;
        B.head = b; //Line 9
        System.out.println("End");
    }
}
```

a 3

b

C

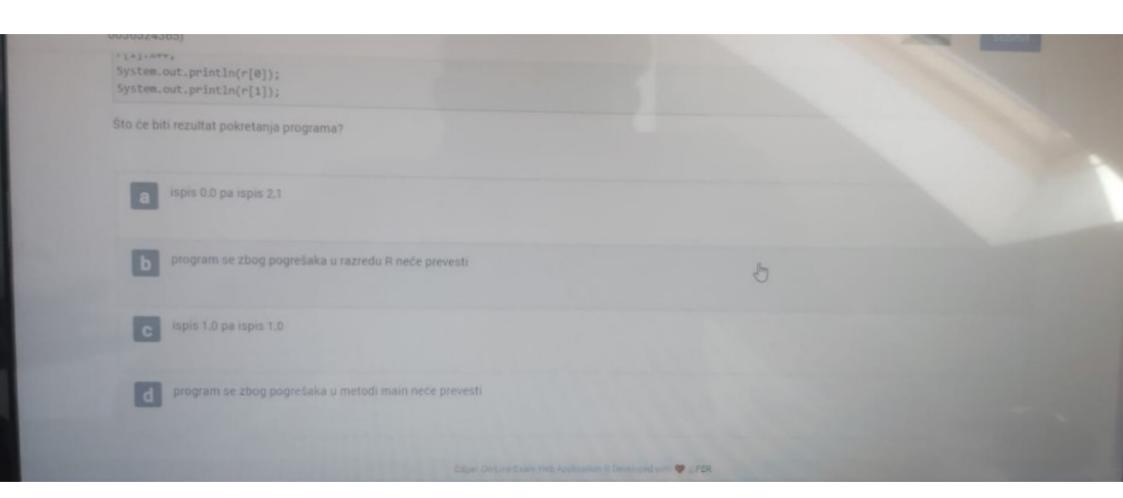
d

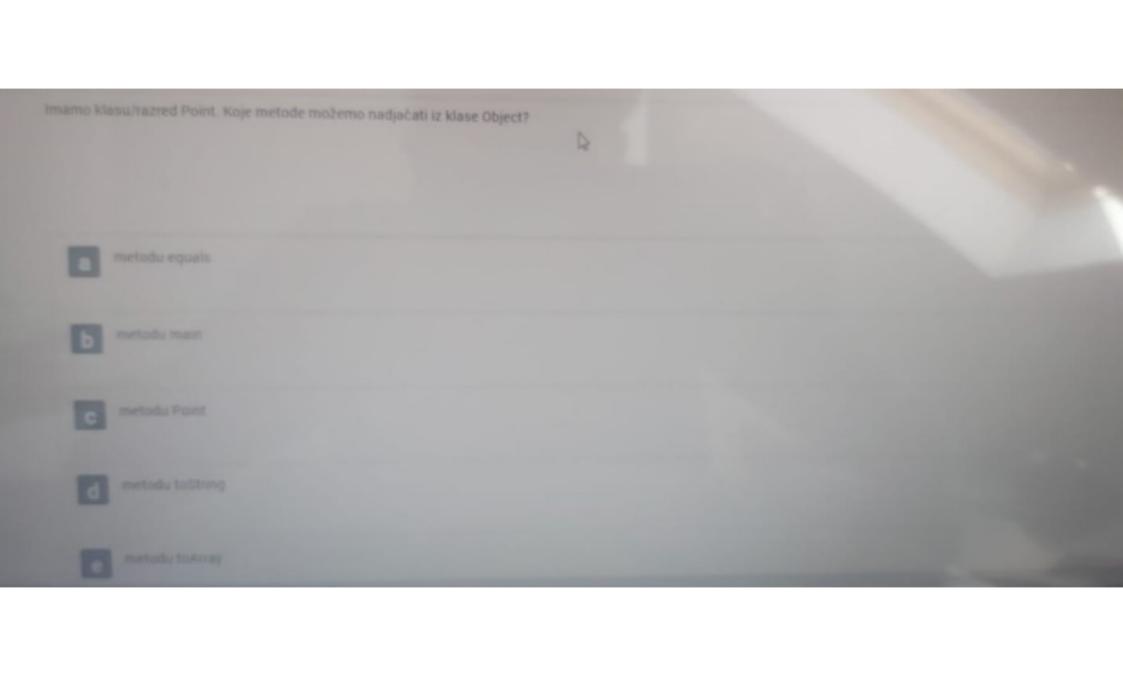


```
interface A {
   int m();
}
class B implements A {
   public int m() {
      return 1;
   }
}
class C extends B {
   public int m() {
      return 2;
   }
}
public class Main {
   public static void main(String[] args) {
      A ref1 = new C();
      B ref2 = (B) ref1;
      System.out.println(ref2.m());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

- Program se neće uspješno prevesti.
- Program će se prevesti, ali neće ispisati ništa i/ili će se srušiti tijekom izvođenja.
- Program će se prevesti i ispisat će 1 tijekom izvođenja.
- Program ĉe se prevesti i ispisat ĉe 1 2 tijekom izvođenja.
- e Program će se prevesti i ispisat će 2 tijekom izvođenja.

```
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[] r = new R[3];
  for(int i = 0; i < r.length; i++) {
   r[i] = 1 = 0 ? new R(i,i) : r[i-1];
  r[1].x++;
  System.out.println(r[8]);
  System:out.println(r[1]):
```





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



- a nista od navedenog
- b mora imati barem jednu apstraktnu metodu
- s konstruktorom se može kreirati primjerak (instanca) ove klase
- primjerak potklase se može castati na tip ove klase

Referenci tipa P mozemo pridijeliti vrijednost pohranjeno u referencu tipa R bez da radimo eksplicitno ukalupljivanje pisanjem (P) ispred reference to a R

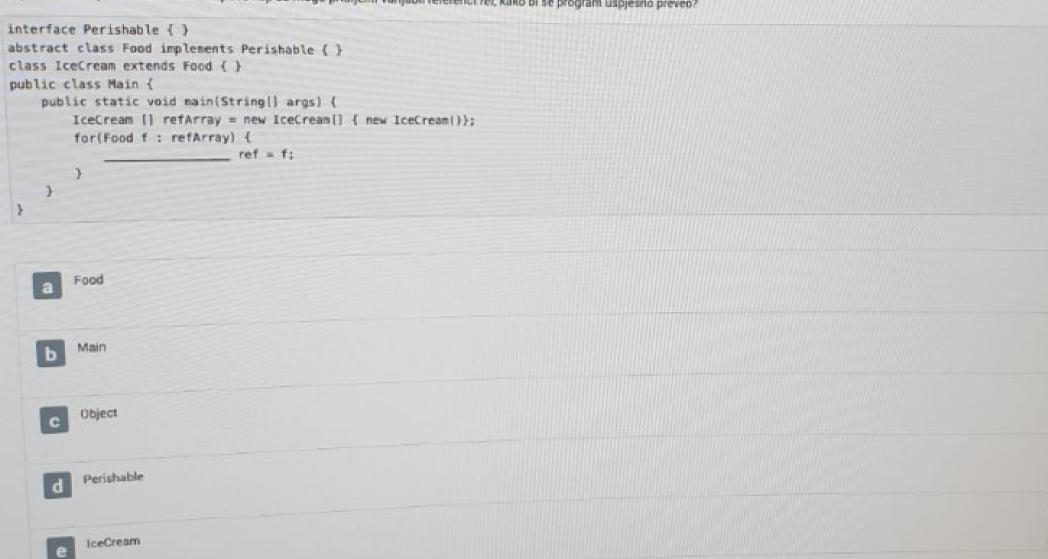
Razmotrite sljedeći kod i označite točne tvrdnje.

```
public class P {
    private int a;
    private P() {}
    public P(int a) {
        this.a = a;
    }
}

public class R extends P {
    private int b;
    public R() {}
    public R(int a) {
        this(a,5);
    }
    public R(int a, int b) {
        this.b=b;
        super(a);
    }
}
```

- Pretpostavljeni konstruktor razreda R je neispravan jer u njemu nismo napisali super();
- Pretpostavljeni konstruktor razreda R je neispravan jer pokušava pozvati nadređeni konstruktor koji mu nije vidljiv
- Konstruktor R(int a, int b) je neispravan jer konsti super, a ako se on koristi, onda mora biti prva naredba u tijelu konstruktora
- d Razred P ne može imati dva konstruktora
- Konstruktor R(int a) je neispravan jer je u njemu umjesto this(a,5) trebalo stajati super(a,5)

Od ponuđenih tipova odaberite sve tipove koji se mogu pridijeliti varijabli referenci ref, kako bi se program uspješno preveo?



Od ponuđenih tipova odaberite sve tipove koji se mogu pridijeliti varijabli referenci ref, kako bi se program uspješno preveo? interface Perishable { } abstract class Food implements Perishable () class IceCream extends Food { } public class Main { public static void main(String[] args) { IceCream [] refArray = new IceCream()); for(Food f : refArray) { __ref = f; Food Main Object Perishable IceCream

Koliko je objekata, primjeraka klase B spremno za uništavanje (garbage collection) u trenutku nakon što se izvrši naredba napisana u liniji 87

```
public class B {
    8 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");
    }
}
```



12

```
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 Ssuml(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);
          Ništa od navedenog
```

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

```
public class R {
  public double x:
  public double y;
  public R() (
    this(3,5);
   public R(int x) {
    y = 3;
     this(x,x):
   public R(int x, int y) {
     this((double)x,(double)y);
    public R(double x, double y) {
      this.x = x;
      this.y = y;
```

Što će biti rezultat prevođenja napisanog izvornog koda u byte-kod?

- Prevodilac će prijaviti pogrešku kod konstruktora R(int x) а
- Prevodilac će prijaviti pogrešku da ne mogu postojati dva konstruktora koja oba primaju dva argumenta b
- Prevodilac će prijaviti pogrešku kod konstruktora R(int x, int y) С
- Kod če se uredno prevesti i nastat će class-datoteka

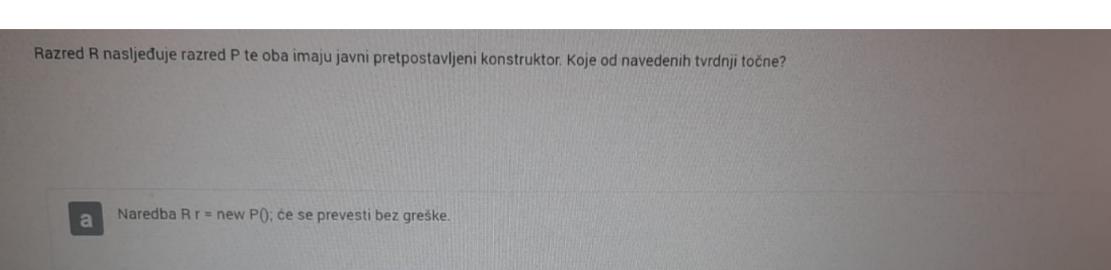
Razmotrite sljedeći kod i označite točne tvrdnje.

```
public class P {
    private int a;
    private P() {}
    public P(int a) {
        this.a = a;
    }
}
```

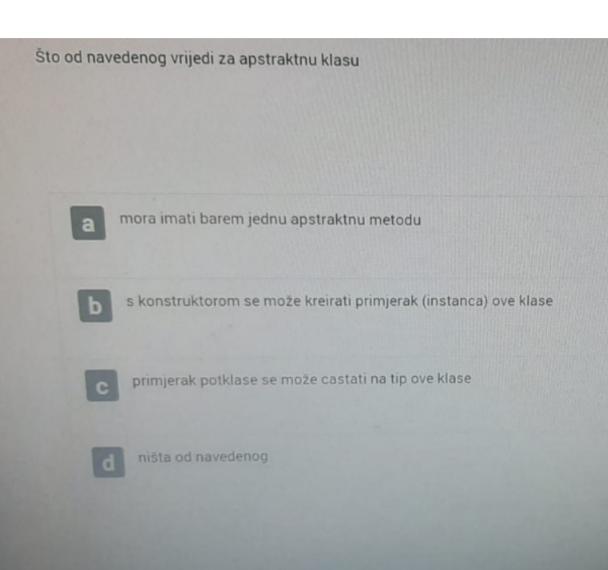
```
public class R extends P {
    private int b;
    public R() {}
    public R(int a) {
        this(a,5);
    }
    public R(int a,int b) {
        this.b=b;
        super(a);
    }
}
```

- Konstruktor R(int a, int b) je neispravan jer koristi super, a ako se on koristi, onda mora biti prva naredba u tijelu konstruktora
- Pretpostavljeni konstruktor razreda R je neispravan jer u njemu nismo napisali super();
- C Konstruktor R(int a) je neispravan jer je u njemu umjesto this(a,5) trebalo stajati super(a,5)
- Razred P ne može imati dva konstruktora
- Pretpostavljeni konstruktor razreda R je neispravan jer pokušava pozvati nadređeni konstruktor koji mu nije vidljiv

74 1 × 10 - 1



- Naredba P p = new R(); će se prevesti bez greške.
- Prevodilac će prilikom prevođenja naredbe R r = new P(); prijaviti grešku.
- Na svaki primjerak razreda P może se gledati kroz referencu tipa R
 - Referenci tipa P ne możemo pridijeliti vrijednost pohranjenu u referencu tipa R bez da radimo eksplicitno ukalupljivanje pisanjem (P) ispred reference tipa i



Što je od navedenog točno ako u programskom kodu piše class A extends B class C extends A

- a Klasa A je specijalizacija klase B
- b Klasa C je specijalizacija klase A
- c Klasa B je generalizacija klase A
- d Klasa C je generalizacija klase A
- Klasa B je specijalizacija klase A

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

```
public class R {
  public double x;
  public double y;

public R() {
  }

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

Ŝto ĉe od ponuđenoga, napisano na mjestu označenom podvlakama, osigurati da se pozivom pretpostavljenog konstruktora u konačnici obavi inicijalizacija objekta postavljanjem članske varijable x na 3 te članske varijable y na 5?

a

```
public K(double x, double y) {
   this.x = x;
   this.y = y;
}
```

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

a R(3,5);

b this.3.5;

c this(3,5);

d this("3","5");

```
U paketu hr.fer.oopj definiran je razred R kako slijedi:
 public class R {
   public double x;
   public double y;
   public R() {
      this(3,5);
    public R(int x) {
      y = 3;
       this(x,x);
     public R(int x, int y) (
       this((double)x,(double)y);
      public R(double x, double y) {
       this.x - x;
        this.y - y;
    Sto će biti rezultat prevođenja napisanog izvornog koda u byte kod?
              Prevodilac će prijaviti pogrešku da ne mogu postojati dva konstruktora koja oba primaju dva argumenta
               Prevoditac če prijaviti pogrešku kod konstruktora R(int x, int y)
               Kod če se uredno prevesti i nastat če class-datoteka
          d Prevodilac će prijaviti pogrešku kod konstruktora R(int x)
```

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; }

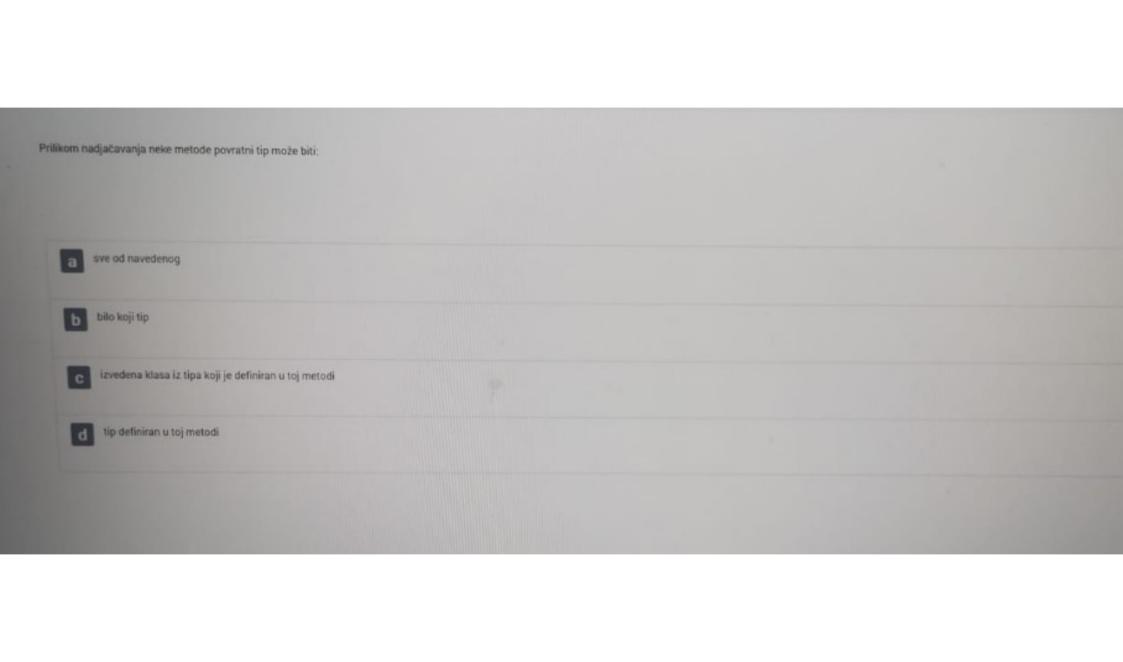
public boolean equals(Object o) {
  R other = (R)o;
  return this.x==other.x && this.y==other.y;
}
```

Potom je u metodi main nekog drugog razreda napisano:

```
R ri = 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(r2));
```

Sto če biti rezultat pokretanja programa?

- a ispis true pa ispis true
- b ispis true pa ispis false
- program se zbog pogrešaka u metodi main neće prevesti
- d ispis false pa ispis false



Deklarirani su sljedeći tipovi:			
<pre>interface I1 { } interface 12 { } class C1 { } class C2 { }</pre>		Q.	
Označite sve dozvoljene deklara	cije.		
a interface I4 exten	ds 11, 12 { }		
b class C4 extends (1 implements 11, 12 { }		
C class C3 implemen	s n, n ()		
d interface 15 impl	ements II ()		
e interface 13 ext	ends 11 ()		

