```
// Klasa0.java
public class Klasa0 {
	static {
		int x = 5;
	}

	static int x, y;

	public static void main(String args[]) {
		x--;
		System.out.println(x + " " + y);
		metoda();
		System.out.println(x + " " + y);
		System.out.println(++x + x++);
		System.out.println(++Klasa0.x);
	}

	public static void metoda() {
		y = ++x;
	}
}
// Klasa1.java

public class Klasa1 {
	int i = 0;

	public static void main(String argv[]) {
	}

	Klasa1() {
		top: while (i < 2) {
			System.out.println(i);
			i++;
			continue top;
		}
	}
}
// Klasa2.java

class Klasa2{

	static double i = 1;
	static int j = 2;
	int x = 3;
	static int y = 6;

	public static void main(String args[]){
		metoda();
		System.out.println(i + j);
		System.out.println(x + i);
		metoda2();
		System.out.println(i + y);
		System.out.println(x + j);
	}

	public static int metoda(){
		return (int)i + --y + (j++);
	}

	public static double metoda2(){
		return  j++ - --i;
	}
};
;;


// Klasa4.java
public class Klasa4 {
	int j = 1;

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

```
		Klasa4 a = new Klasa4();
		a.metoda();
	}
	public static void metoda(){
		char digit = 'a';
		for (int i = 0; i < 10; i++){
			switch (digit){
				case 'x':
				{
					int j = 0;
					System.out.println(j);
				}
				default:
				{
					int j = 100;
					System.out.println(j);
				}
			}
		}
		int i = j;
		System.out.println(i);
	}
}


// Klasa5.java
public class Klasa5 {
	int x = 0, y = 0;
	Klasa5(int a, int b){
		x = a;
		y = b;
	}
	protected int zbir(){
		return x + y;
	}
	protected int razlika(){
		return x - y;
	}
	public static void main(String s[]){
		Klasa5 b = new Klasa5(1,2);
		Klasa6 c = new Klasa6();
		System.out.println(b.razlika());
		System.out.println(c.razlika());
	}
}

class Klasa6 extends Klasa5{
	public int zbir(){
		return y+x;
	}
	public int razlika(){
		return y-x;
	}
}




// Klasa3.java
public class Klasa3{
	static int x = 3;

	public static void main(String[] args) {
		new Klasa3();
	}

	Klasa3() {
		Klasa3(2);
	}

	Klasa3(int x) {
```

```
                System.out.println(x);
        }
}

// Klasa7.java
public class Klasa7 {
        public static void main(String[] args) {
                Klasa7 e = new Klasa7();
                Klasa8 f = new Klasa8();
                try {
                        f.metoda();
                        e.metoda();
                }catch (Exception t) {
                        System.out.println("catch 1");
                }finally{
                        System.out.println("finally");
                }
        }
        void metoda() throws CE1 {
                        throw new CE2("Error 2");
        }
}
class Klasa8 extends Klasa7{
        void metoda(){
                try{
                        throw new CE1();
                }catch (CE1 e) {
                        System.out.println("catch 2");
                }
        }

}
class CE1 extends Exception {
        public CE1() {
                System.out.println("CE1 - 1");
        }
        public CE1(String s) {
                System.out.println(s);
        }
}
class CE2 extends CE1 {
        public CE2() {
                System.out.println("CE2 - 1");
        }
        public CE2(String s) {
                System.out.println("CE2 - 2");
        }
}



//Tablet.java
interface Gadget {
void doStuff();
}
abstract class Electronic {
void getPower() { System.out.print("plug in "); }
}
public class Tablet extends Electronic implements Gadget {
void doStuff() { System.out.print("show book "); }
public static void main(String[] args) {
new Tablet().getPower();
new Tablet().doStuff();
}
}

//Bottom2.java

class Top {
public Top(String s) { System.out.print("B"); }
}
public class Bottom2 extends Top {
```

```
public Bottom2(String s) { System.out.print("D"); }
public static void main(String [] args) {
new Bottom2("C");
System.out.println(" ");
}
}

//Clidlet.java
class Clidder {
private final void flipper() \{ System.out.println("Clidder"); }
}
public class Clidlet extends Clidder {
public final void flipper(){ System.out.println("Clidlet"); }
public static void main(String [] args) {
new Clidlet().flipper();
}
}

//Frodo.java
       public class Frodo extends Hobbit
    public static void main(String[] args) {
    int myGold = 7;
    System.out.println(countGold(myGold, 6));
    }
    }
    class Hobbit {
    int countGold(int x, int y) { return x + y; }
    }
//Hawk.java
class Bird {
    \{ System.out.print("b1 "); }
    public Bird() { System.out.print("b2 "); }
    }
    class Raptor extends Bird {
    static { System.out.print("r1 "); }
    public Raptor() { System.out.print("r2 "); }
    { System.out.print("r3 "); }
    static { System.out.print("r4 "); }
    }
    class Hawk extends Raptor {
    public static void main(String[] args) {
    System.out.print("pre ");
    new Hawk();
    System.out.println("hawk ");
    }
    }
//DogShow.java
class Dog {
    public void bark() \{ System.out.print("woof "); }
    }
    class Hound extends Dog {
    public void sniff() { System.out.print("sniff "); }
    public void bark() { System.out.print("howl "); }
    }
    public class DogShow {
    public static void main(String[] args) { new DogShow().go(); }
    void go() {
    new Hound().bark();
    ((Dog) new Hound()).bark();
    ((Dog) new Hound()).sniff();
    }
    }


// dodati naziv fajla === _____________
class Building {
    Building() { System.out.print("b "); }
    Building(String name) {
    this(); System.out.print("bn " + name);
    }
    }
    public class House extends Building {
```

```
    House() { System.out.print("h "); }
    House(String name) {
    this(); System.out.print("hn " + name);
    }
    public static void main(String[] args) { new House("x "); }
    }

//dodati naziv fajla === _____________________
class Mixer {
    Mixer() {
}
    Mixer(Mixer m) {
        m1 = m;
       }
    Mixer m1;
    public static void main(String[] args) {
    Mixer m2 = new Mixer();
    Mixer m3 = new Mixer(m2); m3.go();
    Mixer m4 = m3.m1; m4.go();
    Mixer m5 = m2.m1; m5.go();
    }
    void go() {
       System.out.print("hi ");
       }




class Fizz {
    int x = 5;\par
    public static void main(String[] args) {
    final Fizz f1 = new Fizz();
    Fizz f2 = new Fizz();
    Fizz f3 = FizzSwitch(f1,f2);
    System.out.println((f1 == f3) + " " + (f1.x == f3.x));\par
  }
    static Fizz FizzSwitch(Fizz x, Fizz y) {
    final Fizz z = x;
    z.x = 6;
    return z;
    }
}

       public class Wind{
    int id;
    Wind(int i){
       id = i;
       }
    public static void main(String[] args){
    new Wind(3).go();
    }
    void go() \{\par
    Wind w1 = new Wind(1);\par
    Wind w2 = new Wind(2);\par
    System.out.println(w1.id + " " + w2.id);\par
   }
}




public class Ouch {
    static int ouch = 7;
    public static void main(String[] args){
    new Ouch().go(ouch);
     System.out.print(" " + ouch);
   }
```

```
    void go(int ouch) {
    ouch++;
    for(int ouch = 3; ouch < 6; ouch++);
    System.out.print(" " + ouch);
   }
}

public class Happy {
    int id;\par
    Happy(int i) {
 id = i;
}
    public static void main(String[] args){
    Happy h1 = new Happy(1);
    Happy h2 = h1.go(h1);
    System.out.println(h2.id);
    }
    Happy go(Happy h) {
    Happy h3 = h;
    h3.id = 2;
    h1.id = 3;
    return h1;
    }
}




class Beta {
}
    class Alpha {
    static Beta b1;
    Beta b2;
    }
    public class Tester {
    public static void main(String[] args){
    Beta b1 = new Beta(); Beta b2 = new Beta();
    Alpha a1 = new Alpha(); Alpha a2 = new Alpha();
    a1.b1 = b1;
    a1.b2 = b1;
    a2.b2 = b2;
    a1 = null;
b1 = null;
b2 = null;--- nacrtati heap i odrediti koliko objekata ce biti pocisceno
}
 }

public class Telescope {
    static int magnify = 2;
    public static void main(String[] args) {
    go();
    }
    static void go() {
    int magnify = 3 ;
    zoomIn();
    }
    static void zoomIn() {
    magnify *= 5;
    zoomMore(magnify);
    System.out.println(magnify);
   }
    static void zoomMore(int magnify) {
    magnify *= 7; } }

     public class Dark {
      int x = 3;
    public static void main(String[] args) {
     new Dark().go1();
    }
     void go1() {
```

```
    int x;
     go2(++x);
    }
    void go2(int y) {
    int x = ++y;
     System.out.println(x);
   }
}

       class Hexy {
    public static void main(String[] args) {
    int i = 42;
    String s = (i<40)?"life":(i>50)?"universe":"everything";
    System.out.println(s);
    }
    }


    public class Main{
    public static void main(String[] args){
    Thread.sleep(500);
    }
    }




    public class Dog {
    String name;
    Dog(String s) { name = s; }
    public static void main(String[] args) {
    Dog d1 = new Dog("Boi");
    Dog d2 = new Dog("Tyri");
    System.out.print((d1 == d2) + " ");
    Dog d3 = new Dog("Boi");
    d2 = d1;
    System.out.print((d1 == d2) + " ");
    System.out.print((d1 == d3) + " ");
    }
    }

    class Feline {
    public static void main(String[] args) {
    long x = 42L;
    long y = 44L;
    System.out.print(" " + 7 + 2 + " ");
    System.out.print(foo() + x + 5 + " ");
    System.out.println(x + y + foo());
    }
    static String foo() { return "foo"; }
    }



    public class SpecialOps {
    public static void main(String[] args) {
    String s = "";
    boolean b1 = true;
    boolean b2 = false;
    if((b2 = false) | (21%5) > 2) s += "x";
     if(b1 || (b2 == true)) s += "y";
     if(b2 == true) s += "z";
     System.out.println(s);
     }
     }


    interface Vessel { }
    interface Toy { }
    class Boat implements Vessel { }
     class Speedboat extends Boat implements Toy { }
     public class Tree {
```

```
 public static void main(String[] args) {
 String s = "0";
 Boat b = new Boat();
 Boat b2 = new Speedboat();
 Speedboat s2 = new Speedboat();
 if((b instanceof Vessel) && (b2 instanceof Toy)) s += "1";
 if((s2 instanceof Vessel) && (s2 instanceof Toy)) s += "2";
 System.out.println(s);
 }
 }


public class Mutant {
public static void main(String[] args) {
StringBuilder sb = new StringBuilder("abc");
String s = "abc";
sb.reverse().append("d");
s.toUpperCase().concat("d");
System.out.println("." + sb + ". ." + s + ".");
}
}


import java.util.*;
public class Sequence {
public static void main(String[] args) {
ArrayList<String> myList = new ArrayList<String>();
myList.add("apple");
myList.add("carrot");
myList.add("banana");
myList.add(1, "plum");
System.out.print(myList);
}
}


public class Tailor {
public static void main(String[] args) {
byte[][] ba = {{1,2,3,4}, {1,2,3}};
System.out.println(ba[1].length + " " + ba.length);
}
}

class Dozens {
 int[] dz = {1,2,3,4,5,6,7,8,9,10,11,12};
}
 public class Eggs {
  public static void main(String[] args) {
   Dozens [] da = new Dozens[3];
da[0] = new Dozens();
 Dozens d = new Dozens();
da[1] = d;
d = null;
da[1] = null;
 // do stuff
 }
 }


public class Hedges {
public static void main(String[] args) {
String s = "JAVA";
s = s + "rocks";
s = s.substring(4,8);
s.toUpperCase();
System.out.println(s);
}
}
class Plane {
static String s = "-";
public static void main(String[] args) {
new Plane().s1();
```

```
        System.out.println(s);
        }
        void s1() {
        try { s2(); }
        catch (Exception e) { s += "c"; }
        }
        void s2() throws Exception {
        s3(); s += "2";
        s3(); s += "2b";
        }
        void s3() throws Exception {
        throw new Exception();
        }
        }


        class Emu {
        static String s = "-";
        public static void main(String[] args) {
        try {
        throw new Exception();
        } catch (Exception e) {
        try {
        try { throw new Exception();
        } catch (Exception ex) { s += "ic "; }
        throw new Exception(); }
        catch (Exception x) { s += "mc "; }
        finally { s += "mf "; }
        } finally { s += "of "; }
        System.out.println(s);
        } }


public class Ebb {
        static int x = 7;
         public static void main(String[] args) {
          String s = "";
         for(int y = 0; y < 3; y++) {
        x++;
        switch(x) {
         case 8: s += "8 ";
        case 9: s += "9 ";
        case 10: { s+= "10 "; break; }
         default: s += "d ";
         case 13: s+= "13 ";
         }
         }
         System.out.println(s);
         }
         static { x++; }
         }


        public class Circles {
         public static void main(String[] args) {
        int[] ia = {1,3,5,7,9};
        for(int x : ia) {
         for(int j = 0; j < 3; j++) {
        if(x > 4 && x < 8) continue;
         System.out.print(" " + x);
         if(j == 1) break;
          continue;
        }
         continue;
         }
         }
         }
```

```
public class OverAndOver {
static String s = "";
 public static void main(String[] args) {
 try {
 s += "1";
 throw new Exception();
 } catch (Exception e) { s += "2";
 } finally { s += "3"; doStuff(); s += "4";
 }
 System.out.println(s);
 }
 static void doStuff() { int x = 0; int y = 7/x; }
 }

 public class Wind {
  public static void main(String[] args) {
foreach:
 for(int j=0; j<5; j++) {
 for(int k=0; k< 3; k++) {
 System.out.print(" " + j);
if(j==3 && k==1) break foreach;
 if(j==0 || j==2) break;
 }
 }
 }
 }

public class Kico {
public static void main(String...  args){
Integer i =  15 ;
int j = i ;
while(i -- > 0 ) ;
System.out.println("i = " + i + " j = " + j );
}
}

 class MyThread extends Thread {
 public static void main(String [] args) throws Exception{
 MyThread t = new MyThread();
 Thread x = new Thread(t);
Thread y = new Thread(t) ;
y.start() ;
y.join() ;
 x.start();
}
 public void run() {
 for(int i=0;i<3;++i) {
 System.out.print(i + "..");
 }
 }
 }

class MyThread extends Thread {
MyThread() {
System.out.print("MyThread ");
}
public void run() {
System.out.print("bar ");
}
public void run(String s) {
System.out.print("baz ");
}
}
public class TestThreads {
public static void main (String [] args) {
Thread t = new MyThread() {
public void run() {
System.out.print("foo ");
}
};
```

```
t.start();
} } -- crtati niti

public class ThreadDemo {
synchronized void a() { actBusy(); }
static synchronized void b() { actBusy(); }
static void actBusy() {
try {
Thread.sleep(1000);
} catch (InterruptedException e) {}
}
public static void main(String[] args) {
final ThreadDemo x = new ThreadDemo();
final ThreadDemo y = new ThreadDemo();
Runnable runnable = new Runnable() {
public void run() {
int option = (int) (Math.random() * 4);
switch (option) {
case 0: x.a(); break;
case 1: x.b(); break;
case 2: y.a(); break;
case 3: y.b(); break;
}
}
};
Thread thread1 = new Thread(runnable);
Thread thread2 = new Thread(runnable);
thread1.start();
thread2.start();
}
}

public class TwoThreads {
static Thread laurel, hardy;
public static void main(String[] args) {
laurel = new Thread() {
public void run() {
System.out.println("A");
try {
hardy.sleep(1000);
} catch (Exception e) {
System.out.println("B");
}
System.out.println("C");
}
};
hardy = new Thread() {
public void run() {
System.out.println("D");
try {
laurel.wait();
} catch (Exception e) {
System.out.println("E");
}
System.out.println("F");
}
};
laurel.start();
hardy.start();
}
}

public class Leader implements Runnable {
public static void main(String[] args) {
Thread t = new Thread(new Leader());
t.start();
 System.out.print("m1 ");
 t.join();
 System.out.print("m2 ");
 }
 public void run() {
 System.out.print("r1 ");
```

```
 System.out.print("r2 ");
 }
 }

import java.io.*;
class Player {
Player() { System.out.print("p"); }
}
class CardPlayer extends Player implements Serializable {
CardPlayer() { System.out.print("c"); }
public static void main(String[] args) {
CardPlayer c1 = new CardPlayer();
try {
FileOutputStream fos = new FileOutputStream("play.txt");
ObjectOutputStream os = new ObjectOutputStream(fos);
os.writeObject(c1);
os.close();
FileInputStream fis = new FileInputStream("play.txt");
ObjectInputStream is = new ObjectInputStream(fis);
CardPlayer c2 = (CardPlayer) is.readObject();
is.close();
} catch (Exception x ) { }
}
}

import java.io.*;
class Keyboard { }
public class Computer implements Serializable {
private Keyboard k = new Keyboard();
public static void main(String[] args) {
Computer c = new Computer();
c.storeIt(c);
}
void storeIt(Computer c) {
try {
ObjectOutputStream os = new ObjectOutputStream(
new FileOutputStream("myFile"));
os.writeObject(c);
os.close();
System.out.println("done");
} catch (Exception x) {System.out.println("exc"); }
}
}

import java.io.*;
public class TestSer {
public static void main(String[] args) {
SpecialSerial s = new SpecialSerial();
try {
ObjectOutputStream os = new ObjectOutputStream(
new FileOutputStream("myFile"));
os.writeObject(s); os.close();
System.out.print(++s.z + " ");
ObjectInputStream is = new ObjectInputStream(
new FileInputStream("myFile"));
SpecialSerial s2 = (SpecialSerial)is.readObject();
is.close();
System.out.println(s2.y + " " + s2.z);
} catch (Exception x) {System.out.println("exc"); }
}
}
class SpecialSerial implements Serializable {
transient int y = 7;
static int z = 9;
}
}

public class Mainy{

       int id;

       double[] doubles = new double[10000000];
```

```
        long[] longs = new long[6000000];
        Mainy[] mys = new Mainy[3];
        Mainy m;
        Dumb d;
        public Mainy(Main m){
               this.m = m;
               if(m!=null)
          id = m.id + 1;
    }

        public static void main(String[] args){
               Mainy m0 = new Mainy(null);
               Mainy m1 = new Mainy(m0);
               Mainy m2 = new Mainy(m1);
               Mainy m3 = new Mainy(m2);
               Mainy m4 = m3;
               Mainy m5 = new Mainy(m0);

               Mainy[] ms = new Mainy[3];
               ms[0] = new Mainy(m2);
               ms[1] = new Mainy(m3);
               ms[2] = new Mainy(m2);
               m1 = m2 = m3 = m5 = null;
               System.gc();   //1               //pozivanje  garbage collectora
               ms[0] = new Mainy(ms[0]);
               ms[1] = new Mainy(ms[1]);
               ms[2] = new Mainy(ms[2]);
               ms[2].m = ms[0];
               m3 = new Mainy(ms[0] = null);
               ms[0] = new Mainy(ms[0]);
               System.gc();  //2
               }
        }

class Dumb {
  float[] flaots = new float[20000000];
  int[] ints = new int[40000000];
```

```
}


1.
public class ThreadA extends Thread {
    String name;
    public ThreadA(String name) {
        this.name = name;
    }
    public void run() {
        new Thread(){
               public void run(){
                            for (int i = 0; i < 10; i++){
                                   System.out.println(name + "2: " + i);
                            }
                     }
        }.start();
        System.out.println(name);
    }

    public static void main(String args[]) throws InterruptedException{
        System.out.println("Pocetak programa");
        ThreadA a = new ThreadA("A");
        ThreadB b = new ThreadB("B");
        ThreadC c = new ThreadC("C", a);
        a.start();
        b.start();
        a.join();
        b.join();
        System.out.println("Kraj programa");
    }
}

class ThreadB extends ThreadA implements Runnable{
    public ThreadB(String name) {
        super(name);
    }
}

class ThreadC extends ThreadB{
    Thread t;
    public ThreadC(String name, Thread t) {
        super(name);
        this.t = t;
        start();
    }
    public void run(){
        try {
               t.join();
        } catch (InterruptedException e) {
               e.printStackTrace();
        }
        for(int i=0; i<10; i++){
               System.out.println(name + "1: " + i);
        }
    }

}

2.
public class ThreadD extends Thread {
    private String name;
    public ThreadD(String name) {
        this.name = name;
    }
    public void run() {
        Runnable r = new Runnable() {
               public void run() {
                      for(int i=0; i<10; i++){
                             System.out.println(name + "1: " + i);
                      }
               }
```

```
        };
        new Thread(r).start();

for(int i=0; i<10; i++){
              System.out.println(name + "2: " + i);

        }
    }
    public static void main(String args[]){
        ThreadD a = new ThreadD("A");
        ThreadD b = new ThreadD("B");
        a.start();
        b.start();
    }
}
3.
public class ThreadE extends Thread {
    private String name;
    public ThreadE(String name) {
        this.name = name;
    }
    public void run() {
        new Runnable() {
              public void run() {
                    for(int i=0; i<10; i++){
                          System.out.println(name + "1: " + i);
                    }
              }
        }.start();

        for (int i = 0; i < 10; i++){
                    System.out.println(name + "2: " + i);
        }
    }
    public static void main(String args[]){
        ThreadE a = new ThreadE("A");
        ThreadE b = new ThreadE("B");
        a.start();
        b.start();
    }
}


// Peta.java
interface PrviI extends DrugiI, TreciI {
    void metoda();
}
interface DrugiI {
    void metoda();
}
interface TreciI {
    void metodaI3();
}

class Peta implements PrviI, TreciI {
    public void metoda() {
        System.out.println("metoda");
    }
    public void metodaI3() {
        System.out.println("metoda I3");
    }
}

class Dvanaesta{
    public static void main(String [] args) {
        Peta e = new Peta();
        PrviI i = new Peta();
        DrugiI i2 = e;
        i.metoda();
        e.metodaI3();
        i.metodaI3();
        i2.metoda();
```

```
    }
}

// H.java
package paketA;

interface I {
    void metoda();
}

class H implements I {
    static H h = new H();
    public static void main(String[] args) {
        H h = new H();
        h.metoda();
    }

    void metoda() {
        System.out.println("abcdef");
    }
}
```

```
//A.java
package paketA;

class A {
    static int i;
    A() {
        ++i;
    }

    private int metoda() {
        return ++i;
    }
};

class B extends A {

    B() {
        i++;
    }

    int metoda() {
        return (i + 3);
    }
};

class X extends B {

    public static void main(String ka[]) {
        X x = new X();
        A a = new A();
        a = (A) x;
        System.out.println(a.metoda());
    }
};

// A.java
public class A {
    int j = 1;
```

```
    public static void main(String args[]){
        metoda();
        A a = new A();
        a.metoda();
    }
    public static void metoda(){
        char digit = 'a';
        for (int i = 0; i < 10; i++){
               switch (digit){
                      case 'x':
                      {
                             int j = 0;
                             System.out.println(j);
                      }
                      default:
                      {
                             int j = 100;
                             System.out.println(j);
                      }
               }
        }
        int i = j;
        System.out.println(i);
    }
}


// B.java
public class B {
    int x = 0, y = 0;
    B(int a, int b){
        x = a;
        y = b;
    }
    protected int zbir(){
        return x + y;
    }
    protected int razlika(){
        return x - y;
    }
    public static void main(String s[]){
        B b = new B(1,2);
        C c = new C();
        System.out.println(b.razlika());
        System.out.println(c.razlika());
    }
}

class C extends B{
    public int zbir(){
        return y+x;
    }
    public int razlika(){
        return y-x;
    }
}


// D.java
public class D implements I3{
    public void metoda3() {
        System.out.println(3);
    }
    public void metoda() {
        System.out.println(1);
    }
    public void metoda2() {
        System.out.println(2);
    }
    public static void main(String args[]){
        D d = new D();
        E e = new E();
```

```
        d.metoda();
        e.metoda();
    }
}

class E implements I{
    public void metoda() {
        System.out.println(11);
    }
}

interface I{
    void metoda();
}

abstract interface I2{
    abstract void metoda2();
}

interface I3 extends I, I2{
    void metoda3();
}


//F.java
public class F {
    private int i = 0;
    private long y = 0;

    long metoda(){
        return i + y;
    }
    F napravi(){
        return new F();
    }
    public static void main(String args[]){
        F f = new F();
        G g = new G();
        F f1 = f.napravi();
        F f2 = g.napravi();
        System.out.println(f1.metoda());
        System.out.println(f2.metoda());
    }
}

class G extends F{
    int i = 1;
    long z = 1;

    protected long metoda(){
        z = super.metoda();
        return i + z;
    }
    G napravi(){
        return new G();
    }
}

import java.util.*;

public class M1{
int id;
M1 m1;
M2 m2;
char[] nizChar = new char[10_000_000];
int[] nizInt = new int[5_000_000];
M2 mArray[][] = new M2[3][2];

public M1(int id)
{
       System.out.println("M1: " + id);
 this.id = id;
```

```
}

public M1(M1 m1,int id,M2 m2)
{
	System.out.println("M1: " + id);
	this.m1 = m1;
	this.m2 = m2;
	this.id = id;
}


@Override//kad se pozove gc on pozove finallize
protected void finalize()
{
System.out.println(id + "finallize");
}

public static void main (String[] args)
{

M1 m10 = new M1(10);
M2 m21 = new M2(m10,21);
M1 m11 = new M1(m10,11,m21);
M2 m22 = new M2(null,22);
M1 m12 = new M1(null,12,m22);

m12.mArray[0][0] = m22;
m12.mArray[1][1] = new M2(23);
m12.mArray[2][1] = new M2(24);
m12.mArray[1] = null;
System.gc();
m10.mArray[1][0] = new M2(25);
m11.m2 = m10.mArray[1][0];
m11 = null;
System.gc();
m10.mArray[2][2] = new M2(new M1(1000),26);
System.gc();
}//zatvoren main

}


class M2
{
float[] f = new float[2_500_000];
M1 m1 = new M1(0);

private int id2 = 0;

public M2(M1 m1,int id2)
{
System.out.println("M2: " + id2);
this.m1 = m1;
this.id2 = id2;
}

public M2(int id2)
{
System.out.println("M2: " + id2);
this.id2 = id2;
}


@Override//kad se pozove gc on pozove finallize
protected void finalize()
{
System.out.println(id2 + "finallize");
}

}
```

```
public class VanjskaKlasa
{
  public static String vanjskiString;

  static{
       System.out.println("Staticki blok, VANJSKE klase");
  };


  public class UnutrasnjaKlasa
  {
   public static String o;
       static{
       System.out.println("Staticki blok, UNUTRASNJE klase");
       }

  }

  public static void main(String[] args)
  {
       VanjskaKlasa vanjskaKlasa = new VanjskaKlasa();
       VanjskaKlasa.UnutrasnjaKlasa unutrasnjaKlasa = vanjskaKlasa. new UnutrasnjaKlasa();
  }

}
```

*testirati serijalizaciju klase kod koje osnovna klasa ne implementira serializable (pozivace se podrazumevani konstruktor osnovne klase)