

Programski jezici 2 – II dio

Termin: 23. 02. 2023. godine

Napomena: Vrijeme trajanja ispita je 60 minuta.

1. Analizirati kod u sljedećim primjerima i utvrditi da li se može kompajlirati i izvršiti. Ako kod nije moguće kompajlirati ili izvršiti, označiti „problematične“ linije koda i navesti razloge. Ako se kod može kompajlirati i izvršiti, napisati izlaze. Po potrebi detaljno obrazložiti. 7 x (5)

```
a) // A1.java

public class A1 {
    private A1 a1;
    static{
        System.out.println("A1-S");
    }
    {
        System.out.println("A1-N");
    }
    public A1() {
        System.out.println("A1");
    }
    public A1(A1 a1){
        System.out.println("A1(A1)");
        this.a1 = a1;
        System.out.println("---");
        new A2(a1);
    }
    void metoda(){
        System.out.println("metoda A1");
    }
    public static void main(String[] args) {
        A1 a1 = new A1();
        a1.metoda();
        A4 a4 = new A4();
        a4.metoda();
        a4.metoda2();
        A3 a3 = new A3();
        A2 a2 = new A2(a1);
        a3.metoda();
        a2.metoda();
    }
}

class A2 extends A1 {
    A1 a1 = new A4();
    static{
        System.out.println("A2-S");
    }
    {
        System.out.println("A2-N");
    }
    public A2() {
        this(new A1());
        System.out.println("A2");
    }
    public A2(A1 a1){
        this.a1 = a1;
        System.out.println("A2(A1)");
    }
    private void metoda2(){
        System.out.println("metoda2 A2");
    }
}
```

```

class A3 extends A2 implements Serializable {
    static{
        System.out.println("A3-S");
    }
    {
        System.out.println("A3-N");
    }
    public A3() {
        System.out.println("A3");
    }
    public A3(A2 a2) {
        this();
        System.out.println("A3(A2)");
    }
    public A3(A2 a2, A1 a1) {
        this(a2);
        System.out.println("A3(A2, A1)");
    }
    void metoda(){
        System.out.println("metoda A3");
    }
    void metoda2(){
        System.out.println("metoda2 A3");
    }
}
class A4 extends A3 {
    private A1 a = new A1();
    private A2 a2 = new A2(new A1(new A1(new A1(new A2()))));
    Serializable a3 = new A3(a2, a1);
    static{
        System.out.println("A4-S");
    }
    {
        System.out.println("A4-N");
    }
    public A4() {
        a2.metoda();
        System.out.println("A4");
        a.metoda();
        ((A1) a3).metoda();
    }
    protected void metoda(){
        System.out.println("metoda A4");
    }
}

b) // B1.java

public class B1 {
    protected B1(){ System.out.println("B1"); }
    double x = 0;
    public static void main(String[] args) throws Exception {
        new B1().metoda();
        B2 b2 = new B2(3.5);
        B3 b3 = new B3(b2);
        if(b2.equals(b3)){
            B1 b[][] = {{b2, b3}, {b2}};
            for (B1[] tmp : b) {
                for(B1 b1: tmp) {
                    System.out.println(b1.metoda());
                }
            }
        }else
            new B3(b2).metoda();
        B1 b1 = (B1) new B1().clone();
        System.out.println(b1.metoda().metoda());
    }
}

```

```

        public B2 metoda(){
            return new B2();
        }

        @Override
        public int hashCode() {
            return (int) x;
        }

        @Override
        public boolean equals(Object obj) {
            B1 tmp = (B1) obj;
            if(tmp.hashCode()==this.hashCode())
                return true;
            return false;
        }
    }

    class B2 extends B1 {
        int x = 1;
        public B2(){
            System.out.println("B2");
        }
        B2(B2 b2){
            System.out.println("B2 - 1");
        }
        B2(double x){
            x = x;
            System.out.println("B2 - 2");
        }

        public B3 metoda(){
            return new B3(this);
        }

        void setX(int x){
            x = x;
        }

        @Override
        public String toString() {
            return super.toString() + " : " + x;
        }
    }

    class B3 extends B2 implements BI {
        int x = 0;
        B3(B2 b2){
            System.out.println("B3");
            x = b2.x;
            b2.setX(x);
        }

        public B3 metoda(){
            return new B3(new B2());
        }
    }

    interface BI {
        B2 metoda();
        Object clone() throws CloneNotSupportedException;
        int hashCode();
    }

```

c) // C1.java

```
public class C1 {
    C2 c2 = new C2();
    C1() {
        System.out.println("C1");
    }
    public static void main(String[] args) {
        C1 c1 = new C1();
        try {
            c1.metoda();
            System.out.println("main 1");
        } catch (CE2 e) {
            System.out.println("main 2: " + e);
        } catch (CE1 e) {
            System.out.println("main 3: " + e);
        } catch (Error e) {
            System.out.println("main 4: " + e);
        } catch (Throwable e) {
            System.out.println("main 5: " + e);
        } finally {
            System.out.println("finally 1");
        }
        C3 c3 = new C3();
        c3.metoda();
    }

    void metoda() throws Throwable {
        try {
            c2.metoda();
            System.out.println("C1: metoda()");
        } finally {
            System.out.println("finally 2");
        }
    }
}

class C2 {
    C3 c3;
    C2() {
        System.out.println("C2");
    }
    void metoda() throws CE1 {
        System.out.println("C2: metoda()");
        c3.metoda();
    }
}

class C3 {
    C3() {
        System.out.println("C3");
    }
    C3(String s) {
        this();
        System.out.println("C3 " + s);
    }
    protected void metoda() {
        try{
            System.out.println("C3: metoda()");
            throw new CE2("CE2");
        } catch (CE2 e) {
            System.out.println("C3: catch");
        } finally{
            System.out.println("finally 3");
        }
    }
}
```

```

class CE1 extends Exception {
    CE1(String s) {
        super(s);
        System.out.println("CE1: " + s);
    }
}

class CE2 extends CE1 {
    CE2(String s) throws CE1 {
        super(s);
        System.out.println("CE2: " + s);
        throw new CE1(s);
    }
}

```

d) // D1.java

```

public class D1 {

    public static void main(String[] args) {
        String test = "Faculty of Electrical Engineering!";
        String res = exec(t -> {
            String result = "";
            for (int i = t.length() - 1; i >= 0; i--) {
                result += t.charAt(--i);
            }
            return result;
        }, test);
        System.out.println(res);
        res = exec(new D2()::exec, test);
        System.out.println(res);
        res = new DI() {
            public String exec(String s) {
                return s.toLowerCase();
            }
        }.exec(test, 3);
        System.out.println(res);
    }

    static String exec(DI sf, String s) {
        return sf.exec(s);
    }
}

class D2 {
    public String exec(String s) {
        String result = "";
        for (int i = s.length() - 1; i >= 0; i--) {
            result += s.charAt(i);
        }
        return result;
    }
}

interface DI {
    public String exec(String s);
    default String exec(String s, int i) {
        return s.substring(i);
    }
}

```

e) // E1.java

```
public class E1 extends Thread{
    public static void main(String x[]) throws InterruptedException {
        System.out.println("one");
        E2 niz[] = {new E3("A") , new E3("B"), new E2(), new E3("D"),
new E3("E")};
        System.out.println("two");
        for (E2 e : niz) {
            System.out.println(e.id);
            if (e instanceof E3){
                new Thread(e).start();
            } else{
                e.start();
                e.join();
            }
        }
        System.out.println("three");
    }
}
class E2 extends Thread {
    String id;
    public E2() { this("C"); }
    E2(String id) {
        System.out.println("E2(): " + id);
        this.id = id;
        if(id.equals("E")) {
            new E3("F").start();
        }
    }
    public void run() {
        for (int i = 1; i < 6; i++) {
            try {
                sleep(1);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println(id + ": " + i);
        }
        new Thread(new Runnable() {
            public void run() {
                for(int i=0; i<10; i++){
                    System.out.println(id + "1: " + i);
                }
            }
        }).start();
        new Thread(){
            public void run(){
                System.out.println(id + ": Thread start...");
                synchronized (this) {
                    try {
                        super.join();
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                    for (int i = 0; i < 10; i++){
                        System.out.println(id + "2: " + i);
                    }
                }
            }
        }.start();
    }
}
class E3 extends E2 implements Runnable {
    E3(String id) {
        super(id);
        System.out.println("E3()");
    }
}
```

```

f) // F1.java
public class F1 {
    public static void main(String[] args) throws Exception {
        F5 f5 = new F5();
        F4 f4 = new F4();
        ObjectOutputStream o = new ObjectOutputStream(new
FileOutputStream("F.out"));
        o.writeObject(f5);
        o.writeObject(f4);
        o.close();
        ObjectInputStream in = new ObjectInputStream(new
FileInputStream("F.out"));
        F5 f55 = (F5) in.readObject();
        System.out.println(f55.a);
        System.out.println(f55.b);
        System.out.println(f55.f32.x);
        System.out.println(f55.f32.y);
        F4 f44 = (F4) in.readObject();
        System.out.println(f44.a);
        System.out.println(f44.b);
        System.out.println(f44.f31.x);
        System.out.println(f44.f31.y);
        in.close();
    }
}

class F2 {
    transient int a = 1;
    transient int b = 2;

    public F2() {
        System.out.println("F2 Constructor");
    }
}

class F31 implements Serializable {
    transient String x = "abc";
    String y = null;
    public F31() {
        System.out.println("F31 Constructor");
        y = "def";
    }
}

class F32 implements Serializable {
    String x = "ghi";
    transient String y = null;
    private void writeObject(ObjectOutputStream out) throws IOException {
        System.out.println("F32.writeObject");
        y = "jkl";
        out.defaultWriteObject();
    }
    public F32() {
        System.out.println("F32 Constructor");
    }
}

```

```

class F4 implements Externalizable {
    int a = 1;
    transient int b = 2;
    transient F31 f31 = new F31();

    public F4() {
        System.out.println("F4 Constructor");
    }
    public void writeExternal(ObjectOutput out) throws IOException {
        out.writeInt(a);
        out.writeInt(b);
        out.writeObject(f31);
        System.out.println("F4.writeExternal");
    }

    public void readExternal(ObjectInput in) throws IOException,
        ClassNotFoundException {
        System.out.println("F4.readExternal");
    }
}

class F5 extends F2 implements Serializable {
    transient long a = 32767;
    transient int b = 1;
    transient F32 f32 = new F32();
    public F5() {
        System.out.println("F5 Constructor");
    }
    private void writeObject(ObjectOutputStream out) throws IOException {
        System.out.println("F5.writeObject");
        out.writeLong(a);
        out.writeInt(b);
        out.writeObject(f32);
    }

    private void readObject(ObjectInputStream in) throws IOException,
        ClassNotFoundException {
        System.out.println("F5.readObject");
        b = in.readInt();
        a = in.readLong();
        f32 = (F32) in.readObject();
    }
}

```


g) G1.java

```
public class G1 {
    public static void main(String[] args) {
        G3 g3 = new G3();
        GT1<G3, G2> gt1 = new GT1<G3, G2>();
        GTI<Long> gt2 = new GT1<Integer, Long>();
        G2 g2 = new G2("g2");
        GTI<Integer> gt3 = new GT1<>();
        gt3.method(2);
        g3.method(2f);
        gt3.method();
        g3.add(gt3.method());
        g3.method2(3);
        gt1.method(g2);
        System.out.println(gt1.t.method());
        gt1.t.method("gt1");
        System.out.println(gt1.t.method());
        g2.method("g22");
        System.out.println(g3.method());
        System.out.println(gt1.t.method());
        System.out.println(gt2.method());
        System.out.println(gt3.method());
    }
}

class GT1<T, T3> implements GTI<T3> {
    T3 t;
    public void method(T3 t) {
        this.t = t;
    }

    public T3 method() {
        System.out.println("Class: " + t.getClass());
        return null;
    }
}

class G2 extends Exception {
    String x = "G2";
    G2(String x){
        x = x;
    }
    void method(String a){
        x = a;
    }
    String method(){
        return x;
    }
}

class G3 extends GT1<Integer, Float> implements GTI2<Integer>{
    public void add(Integer i){
        System.out.println("Class 2: " + t.getClass());
        t += i;
    }
    public void method2(Integer t) {
        this.t += t++;
    }
}

interface GTI<T2> {
    public void method(T2 t);
    public T2 method();
}

interface GTI2<T1> {
    public void method2(T1 t);
}
```

2. Za programski kod sa slike napisati izlaz, te kreirati odgovarajuće memorijske reprezentacije koje obuhvataju stanja *stacka*-a i *heap*-a nakon izvršavanja linija koda označenih brojevima 1, 2 i 3. Pretpostaviti sljedeće: da je maksimalna veličina *heap*-a 900 MB i da će *garbage collector* biti pokrenut odmah po pozivu *System.gc()* i u trenutku kada je na *heap*-u nema dovoljno prostora za smještanje novih objekata . (6)

```

public class M1 extends M2 {
    static int MEM = 0;
    static int counter = 0;
    int id;
    M1 m;
    double dm[] = new double[2_500_000];    // 40 MB
    long lm[] = new long[1_250_000];        // 5 MB
    int im[] = new int[5_000_000];          // 10 MB
    public M1(M1 m, int id){
        counter = id+1;
        System.out.println("M1 " + id);
        if(m!=null)
            System.out.println(" - m -> " + m.id);
        else
            System.out.println(" - m -> null");
        this.m = m;
        this.id = id;
    }
    protected void finalize(){
        System.out.println(this.getClass().getName() + " finalize");
    }
    public static void main(String args[]){
        M1 m0 = null;
        M1 m1 = new M1(m0, 1);
        M1 m2 = new M1(m1, 2);
        M1 m3 = new M1(m2, 3);
        M1 array[][][] = new M1[2][3][2];
        for (int i = 0; i < array.length; i++)
            for (int j = 0; j < array[i].length; j++)
                for(int k=0; k < array[i][j].length; k++){
                    array[i][j][k]=(i+j+k)%2==0?new
M1(test()==1?m2:m3,counter):null;
                    System.out.println("array["+ i +"]"+"["+ j
+"]"+"["+ k +"]" + test());
                }
        System.gc();                                // 1
        for (int i = 0; i < array.length; i++)
            for (int j = 0; j < array[i].length; j++){
                System.out.println("array["+ i +"]"+"["+ j +"]");
                array[i][j][0] = null;
            }
        array[0][1][1].m = null;
        array[1][1][0].m = null;
        array[1][0][1].m = null;
        System.gc();                                // 2
        array[0][1] = null;
        array[1][0] = null;
        System.gc();                                // 3
    }
    private static int test() {
        return counter%2==0?0:1;
    }
}
class M2{
    float fm[] = new float[12_500_000]; // 100 MB
    public M2() {
        System.out.println("M2");
    }
    protected void finalize(){
        System.out.println(this.getClass().getName() + " finalize");
    }
}

```

3. Analizirati kod u sljedećim primjerima i utvrditi da li se može kompajlirati i izvršiti. Ako kod nije moguće kompajlirati ili izvršiti, označiti „problematične“ linije koda i navesti razloge. Ako se kod može kompajlirati i izvršiti, napisati izlaze 6 x 1.

Napomena: boduju se samo odgovori koji su u potpunosti tačni.

a) (1)

```
public class T1 {

    public static void main(String[] args) {
        T11 cube = method(() -> new T11());
        System.out.println(cube.toString());
    }

    public static T11 method(IF<T11> i) {
        return i.get();
    }
}

class T11 {
    public T11() {
        System.out.println("T11 constructor...");
    }
}

interface IF<T> {
    T get();
}
```

b) (1)

```
public class T2 {
    public static void main(String arg[]) {
        int i = 4;
        for (; i <= 12; i+=2) {
            i += i++;
            i -= 1;
            i++;
            i += 1;
            i = i++;
        }
        System.out.println(--i);
    }
}
```

c) (1)

```
public class T3 {
    public static void main(String[] args) throws InterruptedException {
        Runnable r = () -> {
            Thread.sleep(1);
            int sum = 0;
            for (int i = 0; i < 5; i++) {
                sum += i;
            }
            System.out.println(sum);
        };
        new Thread(r).start();
    }
}
```

d) (1)

```
public class T4 {
    public static void main(String[] args) {
        label:
        for (int i = 0; i < 2; i++) {
            for (int j = 0; j < 12; j++) {
                if (j == 4)
                    continue;
                if (j == 7)
                    continue label;
                System.out.println(i + ", " + j);
            }
        }
    }
}
```

e) (1)

```
public class T5 {
    public static void main(String[] args) {
        List<Integer> numbers = Arrays.asList(100, 200, 30, 10, 20,
        50, 100, 400, 200, 500, 300);
        Optional<Integer> result = numbers.stream().distinct().
            filter(e -> e > 100).reduce((a, b) -> a + b);
        System.out.println(result.get());
        numbers.stream().distinct().sorted().
            collect(Collectors.toList()).forEach(System.out::println);
        Stream.iterate(2, e -> e + 2).peek(e -> {
            System.out.print("peek");
        }).limit(20).forEach(System.out::println);
    }
}
```

f) (1)

```
public class T6 {
    public static void main(String[] args) {
        int c1 = 0, c2 = 0;
        switch (c1) {
            case 0:
                System.out.println("c1 is 0");
                switch (c2) {
                    case 0:
                        System.out.println("c2 is 0");
                        break;
                    case 1:
                        System.out.println("c2 is 1");
                        break;
                }
            case 1:
                System.out.println("c1 is 1");
                break;
            default:
                System.out.println("c1 is unknown");
                break;
        }
    }
}
```

g) (1)

```
public class T7 {
    private String str;
    public T7(String str) {
        this.str = str;
    }
    public String toString() {
        return this.str;
    }
    public static void main(String args[]) {
        T7 h1 = new T7("2");
        String s1 = new String("1");
        Object arr[] = new Object[2];
        arr[0] = h1;
        arr[1] = s1;
        Arrays.sort(arr);
        for (Object o : arr) {
            System.out.print(o + " ");
        }
    }
}
```

h) (1)

```
public class T8 {
    public static void main(String[] args) {
        String array[] = { "a", "qwerty", "abc", "def", "test" };
        Arrays.sort(array, new MyComparator());
        for (String e : array) {
            System.out.print(e);
        }
    }
}

class MyComparator implements Comparator<String> {
    @Override
    public int compare(String s1, String s2) {
        return s2.compareTo(s1);
    }
}
```

i) (1)

```
public enum T9 {
    A (1), B (2), C (3), D (4), E (4);

    int val;

    T9(int a){
        val = a;
    }
    int val2(){
        return val*val;
    }
    public static void main(String[] args) {
        for(T9 t5: T9.values()){
            System.out.println(t5);
            System.out.println(t5.ordinal());
            System.out.println(t5.val2());
        }
    }
}
```