

Objektovo-orientované programovanie

doc. Ing. Valentino Vranić, PhD., ÚISI FIIT STU

Test – 4. apríl 2017

A

Priezvisko:	tlačeným písmom
Meno:	

1b	
2b	

Test trvá 30 minút.

Odpovede na otázky vpíšte do tabuľky. Hodnotia sa len odpovede v tabuľke. Pri otázkach s výberom odpovede je len jedna možnosť správna.

V prípade opravy jasne vyznačte odpoveď, ktorá platí. Každá správna odpoveď má hodnotu vyznačenú v otázke. Nesprávna, nejednoznačná alebo neúplná odpoveď má hodnotu 0 bodov. Postup riešenia sa nehodnotí.

Poškodený list nebude uznaný.

1	
2	
3	
4	
5	
6	
7	
8	
9	

1. (1 b) Opätovné využitie štruktúry a správania triedy v inej triede s možnosťou rozšírenia štruktúry a zmeny správania sa v objektovo-orientovanom programovaní označuje ako

- (a) agregácia
- (b) dedenie
- (c) integrácia
- (d) segregácia
- (e) generickosť

2. (1 b) Zachytenie výnimky v Jave

- (a) ohrozuje integritu programu
- (b) automaticky zastavuje program
- (c) automaticky opravuje vzniknutú chybu
- (d) signalizuje výnimočnú situáciu klientskemu kódu
- (e) umožňuje jej spracovanie a pokračovanie v programe

3. (1 b) Aká je hlavná výhoda použitia anonymných tried?

- (a) zrýchlenie programu
- (b) zmenšenie programu
- (c) zvýšenie bezpečnosti
- (d) lepšie vyjadrenie zámeru
- (e) šetrenie pamäte

4. (1 b) Ktorý návrhový vzor implementuje tento kód v Jave (každá trieda a rozhranie vo vlastnom súbore)?

```
public interface I {  
    void m();  
}
```

```
public class M implements I {  
    List<I> p;  
    public m() {  
        ...  
        for (I e : p)  
            e.m();  
    }  
}
```

```
}  
  
public class S implements I {  
    public m() { ... }  
    ...  
}
```

- (a) Visitor
- (b) Observer
- (c) MVC
- (d) Composite
- (e) Strategy

5. (1 b) Zabezpečenie vykonania určitej akcie po kliknutí na tlačidlo sa v rámcoch na zabezpečenie grafického používateľského rozhrania, akými sú JavaFX a Swing, sa dosahuje

- (a) pridaním tejto akcie vo forme prijímača/spracovateľa (listener/handler) udalosti kliknutia registrovaného pre dané tlačidlo
- (b) sledovaním, či je tlačidlo kliknuté v nekonečnej slučke vo vlastnej niti a následným vykonaním akcie
- (c) špeciálnou jazykovou konštrukciou na zachytenie udalosti kliknutia a následným vykonaním akcie, ak sa zachytená udalosť týka príslušného tlačidla
- (d) synchronizáciou udalosti v hlavnej metóde programu
- (e) sledovaním, či je tlačidlo kliknuté v nekonečnej slučke v niti na odosielanie/vykonanie udalostí (event dispatching thread) a následným vykonaním akcie

6. (1 b) Daný je nasledujúci kód v Jave:

```
while (getObject(o)) {  
    if (o instanceof A)  
        ((A) o).opa();  
    else if (o instanceof B)  
        ((B) o).opb();  
    else  
        ;  
}
```

Tento kód porušuje princíp otvorenosti a uzavretosti, lebo

- (a) pribúdaním ďalších tried, ktoré treba zobrať do úvahy, slučka **while** sa stane neprehľadnou
- (b) používa operátor **instanceof** namiesto metódy **isInstance()**
- (c) ak pribudne ďalšia trieda, ktorú treba zobrať do úvahy, treba zmeniť slučku **while**
- (d) triedy a objekty sú na rozdielnej úrovni abstrakcie a netreba ich porovnávať
- (e) nedefinovaním akcie pre **else** nie sú uzavreté všetky možnosti

7. (1 b) Synchronizovať nestatickú metódu v Jave znamená zabezpečiť

- (a) rýchlejšie vykonávanie tejto metódy
- (b) zamknutie aktuálneho objektu (**this**) kým sa táto metóda nevykoná
- (c) rovnomerné prepínanie nití pri vykonávaní tejto metódy
- (d) zamknutie objektu triedy (**class**) kým sa táto metóda nevykoná
- (e) nemennosť tejto metódy v odvodených triedach

8. (1b) Daná je trieda

```
public class C implements Serializable {
    public String id;
    private List<C> l = new ArrayList<>();

    public C(String id) {
        this.id = id;
    }
    public void m(C... c) {
        for (C e : c)
            l.add(e);
    }
}
```

Dané sú inštancie triedy C:

```
C a = new C("a");
C b = new C("b");
C c = new C("c");
C d = new C("d");
b.m(a, d);
c.m(a, b, c, d);
d.m(b);
```

Pre serializáciu týchto inšancií platí, že

- (a) najviac inšancií sa korektne serializuje, ak sa za začiatok serializácie zvolí objekt b
- (b) je jedno, ktorý objekt sa zvolí za začiatok serializácie, lebo **private** zabráni serializácii atribútu l
- (c) najviac inšancií sa korektne serializuje, ak sa za začiatok serializácie zvolí objekt d
- (d) najviac inšancií sa korektne serializuje, ak sa za začiatok serializácie zvolí objekt a
- (e) najviac inšancií sa korektne serializuje, ak sa za začiatok serializácie zvolí objekt c

9. (2b) Čo sa vypíše po spustení nasledujúceho programu v Jave?

```
abstract class P {
    void m(N d) {
        System.out.print("p");
    }
}

interface N {
    void op(A e);
    void op(B e);
}

class A extends P {
    public void m(N d) {
        super.m(d);
        d.op(this);
    }
}

class B extends P {
    public void m(N d) {
        super.m(d);
        d.op(this);
    }
}

class X implements N {
    public void op(A e) {
        System.out.print("ax ");
    }
    public void op(B e) {
        System.out.print("bx ");
    }
}

class Y implements N {
    public void op(A e) {
        System.out.print("ay ");
    }
    public void op(B e) {
        System.out.print("by ");
    }
}

public class M {
    public static void main(String[] args) {
        B o1 = new B();
        P o2 = new A();
        A o3 = new A();
        P o4 = new B();

        o1.m(new X());
        o2.m(new Y());
        o3.m(new X());
        o4.m(new Y());
    }
}
```

Objektovo-orientované programovanie

doc. Ing. Valentino Vranić, PhD., ÚISI FIIT STU

Test – 4. apríl 2017

A

10 b

1 b

2 e

3 d

4 d

5 a

6 c

7 b

8 e

9 pbx pay pax pby