Objektovo-orientované programovanie 2018/19
doc. Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška – opravný termín – 26. jún 2019

(vyplňte tlačeným písmom)
Priezvisko:

Meno:

1 b	
2 b	
3 b	

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	

Skúška trvá 70 minút.
Odpovede na otázky 1–12 vpíšte do tabuľky. Pri týchto otázkach sa hodnotia len odpovede v tabuľke (bez postupu). Odpoveď musí byť jednoznačná a čitateľná, inak má hodnotu 0 bodov.
V otázkach s ponúknutými odpoveďami je len jedna možnosť správna – do tabuľky vpíšte len písmeno, ktorým je označená odpoveď, ktorú vyberáte.
Odpoveď na otázku 13 píšte výlučne na list, na ktorom sa nachádza jej znenie.
Poškodený list nebude uznaný.

1. (1 b) V jazyku C++ virtuálne funkcie slúžia na zabezpečenie

(a) virtuálnej reality
(b) fiktívnej funkcionality
(c) generickosti
(d) polymorfizmu
(e) preťaženia

2. (1 b) Analogickým mechanizmom k šablónam (templates) v jazyku C++ v Jave

(a) sú anonymné triedy
(b) je generickosť
(c) nejestvuje
(d) sú rozhrania
(e) RTTI

3. (1 b) V jazyku C# udalosti

(a) jestvujú ako mechanizmus jazyka a možno ich použiť aj mimo GUI
(b) nepredstavujú mechanizmus jazyka
(c) nemožno implementovať
(d) jestvujú ako mechanizmus jazyka, ale nemožno ich použiť mimo GUI
(e) sú to isté čo mechanizmus delegátov

4. (1 b) Pri serializácií skupiny objektov do jedného súboru je kľúčové

(a) aby sa medzi objektmi v skupine nevyskytovali cirkulárne prepojenia
(b) aby objekty v skupine neboli príliš veľké
(c) aby sa žiaden objekt v skupine neodkazoval na iné jej objekty
(d) aby atribúty objektov v skupine mali viditeľnosť **public**
(e) aby v skupine bol objekt, z ktorého sa dá dostať k jej ostatným objektom

5. (1 b) Je možné V Jave pristúpiť k triede definovanej v inom balíku bez importovania tohto balíka?

(a) áno, ale trieda musí byť uložená v tom istom adresári
(b) áno
(c) áno, ale nedá sa od nej dediť
(d) nie
(e) áno, ale trieda musí byť v tom istom súbore

6. (1 b) V jazyku AspectJ je možné kódom mimo metódy (bez zásahu do jej kódu a volaní)

(a) nahradiť ju úplne iným kódom vrátane vynechania jej vykonania
(b) len znemožniť alebo povoliť jej vykonanie, vrátane jej opakovaného vykonania
(c) nahradiť ju len dostatočne podobným kódom
(d) len zmeniť návratovú hodnotu
(e) nahradiť ju úplne iným kódom, ale nie je možné vynechať jej vykonanie

7. (2 b) Daná je trieda A v Jave:

```
class A {
   void z() throws Q {
      ...
   }
   void m() throws R {
      try {
         z();
      } catch (Q q) {
         ...
      }
      ...
   }
}
```

Z tohto kódu možno usúdiť, že

(a) metóda m() vždy vyhadzuje výnimku typu R
(b) metóda m() môže vyhodiť výnimku typu R a Q
(c) metóda m() vždy vyhadzuje výnimku typu Q
(d) metóda m() môže vyhodiť výnimku typu R
(e) metóda m() môže vyhodiť výnimku typu Q

8. (2 b) Ak sa v programe vyskytuje nasledujúci kód:

```
void aktivujPredmet(Object o) {
   if (MagickyPrsten.class.isInstance(o))
      ((MagickyPrsten)o).zasviet();
   else if (CarovnyPrutik.class.isInstance(o))
      ((CarovnyPrutik)o).vycaruj();
   else if ...
}
```

ten program porušuje

(a) pravidlá dedenia
(b) princíp zapuzdrenia
(c) princíp otvorenosti a uzavretosti
(d) pravidlá polymorfizmu
(e) Liskovej princíp substitúcie

9. (2 b) Daný je nasledujúci program v Jave:

```
class A extends Thread {
    C c;
    public A(C c) {
        this.c = c;
    }
    public synchronized void run() {
        for (int i = 0; i < 9999; i++)
            c.a();
    }
}
class B extends Thread {
    C c;
    public B(C c) {
        this.c = c;
    }
    public synchronized void run() {
        for (int i = 0; i < 9999; i++)
            c.b();
    }
}
class C {
    private char a = 'a', b = 'a';

    public synchronized void a() {
        if (a != b)
            System.out.println("a");
        a = 'a';
        b = 'a';
    }
    public synchronized void b() {
        synchronized(this) {
            if (a != b)
                System.out.println("b");
            a = 'b';
            b = 'b';
        }
    }
    public synchronized void c() {
        System.out.println("c");
    }
    public static void main(String[] args) {
        C c = new C();
        new A(c).start();
        new B(c).start();
    }
}
```

Pri ktorých metódach je možné odstrániť modifikátor **synchronized**, aby stále bolo zaručené, že sa nič nikdy nevypíše (uveďte ich v zápise: `Trieda.metóda()`)?

10. (2 b) V danom softvérovom riešení je potrebné zastrešiť rôzne spôsoby riešenia toho istého problému. Ktorý návrhový vzor by ste použili?

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

11. (3 b) Čo sa vypíše po spustení nasledujúceho programu v Jave?

```
class A {
    public void x() {
        System.out.print("Ax");
    }
    public static void y() {
        System.out.print("Ay");
    }
}
class B extends A {
    public void x() {
        super.x();
        System.out.print("Bx");
    }
    public static void y() {
        A.y();
        System.out.print("By");
    }
}
class C extends B {
    public void x() {
        System.out.print("Cx");
    }
    public static void y() {
        System.out.print("Cy");
    }
}
class M {
    public static void main(String[] args) {
        B o1 = new C();
        A o2 = new B();
        C o3 = new C();
        A o4 = new B();
        B o5 = new B();

        ((B) o1).x();
        ((B) o1).y();
        System.out.print(" ");

        o2.x();
        o2.y();
        System.out.print(" ");

        ((C) o3).x();
        ((C) o3).y();
        System.out.print(" ");

        o4.x();
        o4.y();
        System.out.print(" ");

        ((A) o5).x();
        ((A) o5).y();
    }
}
```

12. (3 b) Metóda garantuje, že po výpočte vráti celé číslo. Metóda, ktorá ju prekonáva, vracia len celé záporne čísla. Týmto sa predpoklady a dôsledky tejto metódy zoslabujú, zosilňujú alebo sa nemenia? Aký to má vplyv na dodržanie Liskovej princípu substitúcie?

Odpovedzte vo forme: P: *predpoklady* / D: *dôsledky* / L: *LSP*. Položky *predpoklady* a *dôsledky* nahraďte jednou z možností *zoslabujú sa*, *zosilňujú sa* alebo *nemenia sa*. Položku *LSP* nahraďte jednou z možností *dodržaný* alebo *nedodržaný*. Nezabudnite uviesť aj písmená P, D a L.

(vyplňte tlačeným písmom)

Priezvisko:

Meno:

13. (10 b) Postava v počítačovej hre môže u seba mať rôzne zbrane, ktoré môže použiť. Zbrane sa pri použití prejavia, len ak postava má dostatok energie na ich aktiváciu. Na výstrel z pištole treba viac než 10% energie, kým na výstrel z luku treba viac než 50% energie. V hre v budúcnosti pribudnú ďalšie druhy zbraní, ktorých použitie tiež bude závisieť od energie postavy. Hra indikuje akcie prostredníctvom výpisov typu „výstrel z luku“ alebo „na výstrel z luku nestačí energia“.

Navrhnite a implementujte v Jave zodpovedajúce objektovo-orientované riešenie zohľadňujúce princípy objektovo-orientovaného programovania. Využite pritom najvhodnejší z návrhových vzorov Strategy, Observer, Visitor a Composite.

Základný návrh predložte vo forme náčrtu diagramu tried v UML, ktorý bude obsahovať najvýznamnejšie vzťahy, operácie a atribúty. Zoberte pritom do úvahy návrhový vzor. Viditeľnosť nie je potrebné uvádzať.

V implementácii sa sústreďte na aplikačnú logiku – GUI nie je predmetom otázky. Taktiež, použité algoritmy nemusia byť optimálne.

Identifikujte explicitne prvky, ktorými sú modelované a implementované roly aplikovaného návrhového vzoru, a vysvetlite, prečo ste ho aplikovali. Poskytnite príklad použitia, v ktorom vytvoríte príslušné objekty a spustíte ich interakciu.

Odpoveď bude hodnotená podľa nasledujúceho kľúča:

- zabezpečenie základnej funkčnosti – 4 b
- kvalita a flexibilita objektovo-orientovaného návrhu – 6 b

Objektovo-orientované programovanie 2018/19
doc. Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška – opravný termín – 26. jún 2019
30

1 d

2 b

3 a

4 e

5 b

6 a

7 d

8 c

9 A.run(), B.run(), C.b() a C.c()

10 c

11 CxAyBy AxBxAy CxCy AxBxAy AxBxAy

12 P: nemenia sa / D: zosilňujú sa / L: dodržaný

V poslednej otázke mal byť aplikovaný vzor Visitor. Postava je v role Element, pričom operácia accept() predstavuje aktiváciu zbrane. Zbrane sú v role Visitor, pričom ich skutočný prejav je daný operáciou visit(). Na pridanie ďalšej zbrane stačí od rozhrania alebo triedy, ktoré implementuje všeobecnú zbraň, odvodiť ďalšiu triedu a implementovať operáciu visit().