Objektovo-orientované programovanie 2017/18 **A**

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

Skúška – 8. jún 2018

(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. (2 b) Súčasťou riešenia problému je aj určitá funkčná časť, ktorá v rôznych prípadoch jeho uplatnenia môže byť iná. Ktorý návrhový vzor by ste použili?

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

2. (1 b) V jazyku C++ sa použitie šablón (templates) rieši

(a) v čase vykonávania
(b) generovaním špeciálneho kódu pre každý zadaný parameter
(c) ponechaním kódu bez úprav
(d) generovaním spoločného kódu pre všetky zadané parametre
(e) zlúčením všetkých šablón

3. (1 b) V jazyku C# (menný priestor – angl. namespace)

(a) možno definovať viac menných priestorov v jednom súbore a prvky z rôznych súborov môžu byť súčasťou jedného menného priestoru
(b) prvky z rôznych súborov môžu byť súčasťou jedného menného priestoru, ak sa nachádzajú v spoločnom adresári
(c) možno definovať viac menných priestorov v jednom súbore, ale prvky z rôznych súborov nemôžu byť súčasťou jedného menného priestoru
(d) prvky z rôznych súborov môžu byť súčasťou jedného menného priestoru, ale nemožno definovať viac menných priestorov v jednom súbore
(e) v jednom adresári možno definovať iba jeden menný priestor

4. (1 b) Java podporuje perzistenciu prostredníctvom

(a) agregácie
(b) radializácie
(c) serializácie
(d) modularizácie
(e) synchronizácie

5. (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() {
        System.out.print("Bx");
    }
    public static void y() {
        A.y();
        System.out.print("By");
    }
}
class C extends B {
    public void x() {
        super.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();
        A o3 = new A();
        B o4 = new B();
        C o5 = new C();

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

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

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

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

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

6. (1 b) Generickosť v Jave umožňuje predovšetkým to, aby zoskupenia

(a) boli automaticky ukladané na disk
(b) mohli byť špecializované pre akýkoľvek typ údajov pri inštanciácii
(c) fungovali rýchlejšie
(d) mohli byť špecializované pre akýkoľvek typ údajov pri dedení
(e) mohli uchovať väčší počet objektov

7. (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';
        b = 'a';
    }
    public synchronized void b() {
        synchronized(this) {
            if (a != b)
                System.out.println("!");
            a = 'b';
            b = 'b';
        }
    }
    public static synchronized 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 výkričník nikdy nevypíše (uveďte ich v zápise: `Trieda.metóda()`)?

8. (1 b) Na to, aby v jazyku AspectJ bolo možné vykonať kód pred metódou

(a) jej názvu musí predchádzať predponá before
(b) pred jej názvom treba použiť kľúčové slovo before
(c) treba ju premiestniť do aspektu
(d) treba ju označiť anotáciou @before
(e) nie je potrebné do nej zasahovať

9. (3 b) Trieda, ktorá reprezentuje špeciálnu hru, je odvodená od triedy, ktorá reprezentuje všeobecnú hru. Metóda na pridanie hráča je v špeciálnej hre prekonaná tak, že povoľuje pridanie hráča s akýmkoľvek počtom bodov, kým vo všeobecnej hre počet bodov musí byť väčší ako nula (možné je aj záporné hodnotenie). Týmto sa predpoklady a dôsledky týchto metód zoslabujú, zosilňujú alebo sa nemenia? Je týmto dodržaný Liskovej princíp substitúcie (LSP)?

Odpovedzte vo forme: *predpoklady / dôsledky / 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*.

10. (2 b) Čo všetko sa vypíše prostredníctvom príkazov System.out.print() po spustení nasledujúceho programu v Jave (po jeho úspešné alebo neúspešné ukončenie)?

```
class E extends Exception {}

class M {
    public void c(char c) throws E {
        if (c == 'a')
            System.out.print("A");
        else
            throw new E();
    }
    public void m(char c) {
        System.out.print("M");

        try {
            c(c);
        } catch (E e) {
            System.out.print("E");
        } finally {
            System.out.print("F ");
        }
    }
    public static void main(String[] args) {
        new M().m('b');
        new M().m('b');
        new M().m('a');
    }
}
```

11. (2 b) Súčasťou grafického používateľského rozhrania počítačovej hry je aj tlačidlo t (objekt typu JButton), v súvislosti s ktorým sa v hre vyskytuje nasledujúci kód v Jave:

```
class GameWindow extends JFrame {
    Player player;
    ...
    class DecreaseEnergy implements ActionListener() {
        public void actionPerformed(ActionEvent e) {
            if (player.hasShield())
                player.setEnergy(player.getEnergy() − 1);
            else
                player.setEnergy(player.getEnergy() − 2);
        }
    }
    ...
    t.addActionListener(new DecreaseEnergy());
    ...
}
```

Z hľadiska flexibility objektovo-orientovaného návrhu by bolo najdôležitejšie

(a) vyčleniť kód v metóde actionPerformed() do implementácie logiky hry
(b) odvodiť zodpovedajúce triedy z triedy Player a použiť polymorfizmus namiesto príkazu **if**
(c) zmeniť triedu DecreaseEnergy na anonymnú alebo použiť lambda výraz na priradenie metódy actionPerformed() k tlačidlu t
(d) pomenovať tlačidlo t výstižnejšie
(e) aplikovať modifikátor prístupu **private** na atribút player

12. (1 b) V jazyku C++ možno vytvoriť nový objekt

(a) iba z tried, ktoré neobsahujú virtuálne funkcie
(b) nie z triedy, ale iba zo šablóny
(c) výlučne definovaním premennej typu danej triedy
(d) pomocou operátora **new** ako v Jave alebo definovaním premennej typu danej triedy
(e) výlučne pomocou operátora **new** ako v Jave

(vyplňte tlačeným písmom)

Priezvisko:

Meno:

13. (10 b) V informačnom systéme fakulty sa okrem iného vedie evidencia o predmetoch. Jestvujú dva druhy predmetov: prednášané predmety a projektové predmety. Každý druh predmetu je implementovaný vlastnou triedou. Pre každý predmet sa eviduje názov, anotácia a počet kreditov, ale niektoré údaje sú špecifické podľa druhu predmetu. Prednášané predmety majú prednášateľa a garanta, kým projektové predmety nemajú prednášateľa, ale iba garanta. Pri prednášaných predmetoch sa eviduje počet hodín prednášok a cvičení týždenne, kým pri projektových nie.

Systém poskytuje možnosť výpisu súhrnných informácií o predmetoch v úplnej a skrátenej forme. Pri úplnej forme sa vypíšu všetky informácie, pričom sa nevypisujú položky irelevantné vzhľadom na druh daného predmetu. Pri skrátenej forme sa vypíše názov predmetu a k tomu pri prednášaných predmetoch meno prednášateľa, kým pri projektových predmetoch meno garanta. V ďalších verziách sa očakáva pridanie nových foriem výpisov.

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 2017/18 **A**
doc. Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška – 8. jún 2018
30

1 e

2 b

3 a

4 c

5 BxCxCy BxAy AxAy BxAyBy BxCxAyBy (akceptovaná je aj odpoveď bez medzier)

6 b

7 A.run(), B.run(), C.b() a C.main()

8 e

9 zoslabujú sa / nemenia sa / dodržaný

10 MEF MEF MAF (akceptovaná je aj odpoveď bez medzier)

11 a

12 d

V poslednej otázke mal byť aplikovaný vzor Visitor. Druhy predmetov by boli v role Elementu, a druhy výpisov v role Visitora.