Objektovo-orientované programovanie 2023/24 **A**
prof. Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška – 11. jún 2024

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) Viacnásobné dedenie v jazyku C++
(a) slúži na zníženie počtu tried
(b) nahrádza mechanizmus friend
(c) môže prispieť k lepšiemu oddeleniu záležitostí
(d) je to isté, čo opakované dedenie
(e) predstavuje alternatívu k virtuálnym funkciám

2. (1 b) Princíp otvorenosti a uzavretosti hovorí, že
(a) každý prúd údajov je potrebné po otvorení aj zatvoriť
(b) kód má byť otvorený pre zmeny, ale uzavretý pre rozšírenie
(c) kód má byť zároveň aj otvorený, aj uzavretý pre úpravy
(d) každý prúd údajov okrem štandardného vstupu a výstupu je potrebné po otvorení aj zatvoriť
(e) kód má byť otvorený pre rozšírenie, ale uzavretý pre zmeny

3. (1 b) V jazyku C++ prekonávanie sa použitím špeciálneho kľúčového slova
(a) môže v prípade, že nie je potrebné, deaktivovať
(b) musí v prípade potreby aktivovať
(c) musí v prípade potreby aktivovať, a následne sa môže deaktivovať
(d) musí v prípade, že nie je potrebné, deaktivovať
(e) musí v prípade potreby aktivovať, a následne sa musí aj deaktivovať

4. (1 b) To, že sa namiesto objektu jednej triedy dá použiť objekt inej triedy znamená, že medzi týmito triedami je vzťah

(a) agregácie
(b) integrácie
(c) segregácie
(d) dedenia
(e) generickosti

5. (1 b) V jazyku C# zástupca (delegate) reprezentuje
(a) funkciu zodpovedajúcich typov parametrov a návratovej hodnoty
(b) ukazovateľ na objekt
(c) vlastnosť zodpovedajúceho typu
(d) hocijakú funkciu
(e) hocijakú vlastnosť

6. (1 b) Výraz call(abc∗(..)) v jazyku AspectJ znamená
(a) vyvolanie metódy call() pred všetkými metódami, ktorých názov začína na abc
(b) zachytenie volania všetkých metód, ktorých názov začína na abc
(c) vyvolanie prvej metódy, ktorej názov začína na abc
(d) vyvolanie všetkých metód, ktorých názov začína na abc
(e) zachytenie prvého volania metódy, ktorej názov začína na abc

7. (2 b) Daný je nasledujúci program v Jave:

```
class C implements Runnable {
    M m;

    public C(M m) {
        this.m = m;
    }
    public void run() {
        for (int i = 0; i < 100000; i++) {
            m.p1();
            m.p2();
            m.p3();
        }
    }
}

public class M {
    private int a = 1, b = 1;

    public void p1() {
        if (a != b)
            System.out.print("1");
        a = 1;
        b = 1;
    }

    public void p2() {
        if (a != b)
            System.out.print("2");
        a = 2;
        b = 2;
    }

    public void p3() {
        synchronized(this) {
            if (a != b)
                System.out.print("3");
            a = 3;
            b = 3;
        }
    }

    public static void main(String[] args) {
        M m = new M();
        new Thread(new C(m)).start();
        new Thread(new C(m)).start();
    }
}
```

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

8. (2 b) Pre ktorý návrhový vzor je charakteristická notifikácia určitých objektov pri zmene stavu iného objektu?

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

9. (2 b) Čo všetko sa vypíše prostredníctvom príkazov System.out.print() po spustení nasledujúceho programu v Jave (po jeho úspešné alebo neúspešné ukončenie)?

```
public class E {
    public void c(int n) throws Exception {
        if (n == 0)
            throw new Exception();
    }

    public void m(int n) {
        try {
            c(n);
        } catch (Exception e) {
            System.out.print("E");
        } finally {
             System.out.print(n);
        }
    }

    public static void main(String[] args) {
        E e = new E();
        e.m(2);
        e.m(0);
        e.m(2);
    }
}
```

10. (2 b) Grafické používateľské rozhranie počítačovej hry je vytvorené pomocou rámca JavaFX. Jeho súčasťou je aj tlačidlo t, v súvislosti s ktorým sa v hre vyskytuje nasledujúci kód:

```
t.setOnAction(e -> {
    if (player.hasShield())
        player.setEnergy(player.getEnergy() - 1);
    else
        player.setEnergy(player.getEnergy() - 2);
});
```

Primárny problém tohto kódu z hľadiska objektovo-orientovaného návrhu je to, že

(a) aplikačná logika figuruje v používateľskom rozhraní
(b) spracovateľ udalostí nebol realizovaný prostredníctvom anonymnej triedy
(c) nebol použitý polymorfizmus
(d) nebolo použité zapuzdrenie
(e) neobsahuje komentár

11. (3 b) Čo všetko sa vypíše prostredníctvom príkazov System.out.print() po spustení nasledujúceho programu v Jave?

```
class C {
    public void f() {
        System.out.print("C");
    }
}
class D extends C {
    public void f() {
        super.f();
        System.out.print("D");
    }
}
class E extends D {
    public void f() {
        super.f();
        System.out.print("E");
    }
}
class F extends E {
    public void f() {
        super.f();
        System.out.print("F");
    }
}
public class M {
    public static void main(String[] args) {
        F o1 = new F();
        C o2 = new E();
        E o3 = new E();
        D o4 = new F();

        ((E) o1).f();
        System.out.print(" ");

        ((D) o2).f();
        System.out.print(" ");

        ((C) o3).f();
        System.out.print(" ");

        ((C) o4).f();
        System.out.print(" ");
     }
}
```

12. (3 b) Metóda garantuje, že po výpočte vráti celé kladné číslo. Metóda, ktorá ju prekonáva, vracia celé číslo, ktoré môže byť aj záporne.

Týmto sa dôsledky a predpoklady pôvodnej metódy zoslabujú, zosilňujú alebo nemenia? Je týmto dodržaný Liskovej princíp substitúcie (LSP)? Je v tomto prípade vhodne použité dedenie?

Odpovedzte vo forme: *predpoklady / dôsledky / LSP / dedenie*. Položky *dôsledky* a *predpoklady* nahraďte jednou z možností *zoslabujú sa*, *zosilňujú sa* alebo *nemenia sa*. Položku *LSP* nahraďte jednou z možností *dodržaný* alebo *nedodržaný*. Položku *dedenie* nahraďte jednou z možností *áno* alebo *nie*.

Priezvisko:

Meno:

13. (10 b) Virtuálny priestor pozostáva z poprepájaných buniek. Bunka môže byť nečleniteľná alebo členiteľná. Členiteľná bunka môže obsahovať ďalšie nečleniteľné a členiteľné bunky. Bunky môžu byť prepojené bez ohľadu na to, či sú nečleniteľné alebo členiteľné. Pri každej bunke je možné získať zoznam buniek, do ktorých možno z tejto bunky vstúpiť, čo sú bunky, s ktorými je daná bunka priamo prepojená, a – v prípade členiteľnej bunky – bunky, z ktorých daná bunka bezprostredne pozostáva (prvá úroveň).

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 prvkov nie je potrebné uvádzať.

V implementácii sa sústreďte na aplikačnú logiku – používateľské rozhranie nie je predmetom otázky. *Samotné efekty (ohňostroj, dúha atď.) iba naznačte – neimplementujte ich.*

Identifikujte explicitne prvky, ktorými sú modelované a implementované roly aplikovaného návrhového vzoru. 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 2023/24 **A**
prof. Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška – 11. jún 2024
30

1 c

2 e

3 b

4 d

5 a

6 b

7 M.p1() a M.p2()

8 c

9 2E02

10 a

11 CDEF CDE CDE CDEF (akceptovaná je aj odpoveď bez medzier)

12 nemenia sa / zoslabujú sa / nedodržaný / nie

13 Vhodný je vzor Composite. Bunka ako taká by bola v role Component (rozhranie alebo abstraktná trieda), členiteľná bunka v role Composite, a nečleniteľná bunka v role Leaf (triedy odvodené od triedy reprezentujúcej bunku). Operácia výpisu alebo vrátení spojení mala byť implementovaná tak, aby sa dala uplatniť transparentne aj nad členiteľnou, aj nad nečleniteľnou bunkou (mala byť prítomná už v bunke, a v členiteľnej a nečleniteľnej bunke prekonaná).

Aj členiteľná, aj nečleniteľná bunka mali obsahovať evidenciu (napr. zoznam alebo pole) buniek, s ktorými sú spojené. Členiteľná bunka mala navyše obsahovať evidenciu buniek, z ktorých pozostáva. Operácia výpisu alebo vrátení spojení pri nečleniteľnej bunke mala vrátiť len bunky, s ktorými je daná bunka spojená, kým pri členiteľnej bunke k tomu mali pribudnúť aj bunky, z ktorých táto pozostáva.