Objektovo-orientované programovanie **A**

Ing. Valentino Vranić, PhD., ÚISI FIIT STU

Skúška — 4. jún 2008

Priezvisko:	
Meno:	

1 b	
2 b	
3 b	

	a	b	c	d	e
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					

Skúška trvá najviac 100 minút. V otázkach 1–16 je len jedna možnosť správna. Vyznačte svoju odpoveď krížikom do veľkej tabuľky (malú tabuľku nevypĺňajte). Hodnotia sa len odpovede vyznačené v tabuľke. V prípade opravy jasne vyznačte ktorú odpoveď vyberáte. Každá správna odpoveď má hodnotu vyznačenú v otázke. Nesprávna odpoveď, vyznačenie viac odpovedí alebo nejednoznačné vyznačenie má hodnotu 0 bodov. Postup riešenia sa pre otázky 1–16 nehodnotí. Odpovede na otázky 17 a 18 píšte na prídavný list. Na ňom tiež uveďte svoje priezvisko a meno.

1. (1 b) Prúd údajov (stream) v Java API sa obaľuje, aby

(a) bolo ľahšie zachytiť výnimku IOException
(b) bolo vôbec možné pracovať s prúdom údajov
(c) prístup k prúdu údajov mohol byť realizovaný operáciami vyššej úrovne
(d) nedošlo k strate údajov v dôsledku vysokej rýchlosti ich prenosu
(e) bolo možné prúd údajov presmerovať

2. (1 b) Prúd údajov (stream) v Java API sa zatvára

(a) príkazom IOStream.close()
(b) jeho metódou close()
(c) príkazom System.close()
(d) prečítaním posledného údaja v ňom
(e) zavolaním jeho deštruktora

3. (1 b) Dá sa v Jave niť vytvoriť implementáciou rozhrania Runnable?

(a) nie
(b) áno, ale len ako démon
(c) áno, ale len atomická
(d) áno, ale len synchronizovaná
(e) áno, ale na spustenie nite sa musí použiť trieda Thread

4. (1 b) Na rozdiel od Javy jazyk C++

(a) má explicitné deštruktory
(b) automaticky vytvára objekty
(c) automaticky ruší nereferencované objekty
(d) má explicitné konštruktory
(e) má implicitné konštruktory

5. (2 b) Okno v rámci Swing sa dá vytvoriť

(a) zavolaním konštruktora triedy JFrame
(b) zavolaním metódy newWindow() triedy JFrame
(c) zavolaním statickej metódy EventQueue.newWindow()
(d) zavolaním konštruktora triedy SwingWindow
(e) zavolaním statickej metódy SwingUtils.newWindow()

6. (2 b) V diagrame sekvencií jazyka UML horizontálne šípky s plnou čiarou označujú

(a) agregáciu
(b) návrat hodnoty
(c) tok údajov medzi objektmi
(d) vyvolanie operácie
(e) prenos parametrov operácie v smere šípky

7. (2 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 otvorený pre rozšírenie, ale uzavretý pre zmeny
(d) kód má byť zároveň aj otvorený, aj uzavretý pre úpravy
(e) každý prúd údajov okrem štandardného vstupu a výstupu je potrebné po otvorení aj zatvoriť

8. (2 b) V jazyku AspectJ je pomocou videnia (advice) možné

(a) pridať novú metódu do triedy
(b) pridať novú metódu do aspektu
(c) zmeniť vykonávanie metódy
(d) definovať bod spájania
(e) definovať nové závislosti medzi triedami

9. (2 b) Implementácii rozhrania v Jave v jazyku C++ zodpovedá:

(a) **public** dedenie od triedy, ktorá má len virtuálne metódy bez tela
(b) **private** dedenie od triedy, ktorá má len virtuálne metódy bez tela
(c) **private** dedenie od **abstract** triedy
(d) **protected** dedenie od **abstract** triedy
(e) **public** dedenie od **abstract** triedy

10. (2 b) Prístup **protected** je vhodné použiť pri takých prvkoch triedy, ku ktorým chceme pristupovať len

(a) v odvodených triedach
(b) v odvodených triedach a v triedach toho istého balíka
(c) v triedach toho istého balíka
(d) v odvodených triedach toho istého balíka
(e) v danej triede

11. (2 b) Agregácia v objektovo-orientovanom programovaní

(a) znamená spájanie objektov do väčších celkov
(b) znamená skrytie implementácie objektu
(c) stanovuje kritéria pre použitie abstraktných tried
(d) predstavuje kritérium pre použitie dedenia
(e) umožňuje, aby sa objekt uplatnil namiesto objektu jeho nadtypu

12. (3 b) Kód na obr. 1 predstavuje implementáciu (riešte najprv otázku 13)

(a) vzoru Observer, pričom Num je pozorovateľ, a NOperation predmet
(b) vzoru Observer, pričom NOperation je pozorovateľ, a Num predmet
(c) idiómu double dispatch, pričom Num je double dispatch, a NOperation predmet
(d) vzoru Visitor, pričom NOperation je návštevník, a Num predmet
(e) vzoru Visitor, pričom Num je návštevník, a NOperation predmet

13. (3 b) Daný je kód na obr. 1. Ktoré fragmenty treba do kódu doplniť, aby bol správny?

(a) `**1**: PlainInt **2**: NOperation`
`**3**: p.op(this); **4**: n.calc(new NSquare())`
(b) `**1**: NOperation **2**: PlainInt`
`**3**: p.get() **4**: n.calc(new PlainInt());`
(c) `**1**: PlainInt **2**: int`
`**3**: p.op(this); **4**: n.calc(n.get())`
(d) `**1**: PlainInt **2**: NOperation`
`**3**: p; **4**: n.calc(this)`
(e) `**1**: NOperation **2**: PlainInt`
`**3**: this; **4**: n.calc(new NSquare())`

```
public interface NOperation {
    int op(**1** n);
}
public interface Num {
    int calc(**2** p);
    void set(int i);
    int get();
}
public class PlainInt implements Num {
    private int i;
    public void set(int i) { this.i = i; }
    public int get() { return i; }
    public int calc(**2** p) { return **3** }
}
public class NSquare implements NOperation {
    public int op(**1** n) { return n.get() * n.get(); }
}
public class M {
    public static void main(String[] args) {
        Num n = new PlainInt();
        n.set(3);

        System.out.println(**4**);
    }
}
```

Obrázok 1: Kód pre otázky 12 a 13.

14. (3 b) K typom

```
abstract class A { }
interface I { }
```

je daný nasledujúci kód (umiestnený v metóde v tom istom balíku):

```
A[] a = new A[5];
I[] i = new I[5];
```

Tento kód sa

(a) preloží, ale vznikne chyba pri vykonávaní prvého riadku
(b) preloží, ale vznikne chyba pri vykonávaní druhého riadku
(c) nepreloží, lebo prekladač hlási chybu v prvom riadku
(d) nepreloží, lebo prekladač hlási chybu v druhom riadku
(e) preloží a vykoná korektne

15. (3 b) V programe je každý druh geometrického útvaru reprezentovaný triedou. Kód mimo týchto tried využíva rozmery útvarov na výpočet ich povrchu. Je vhodné odvodiť triedu, ktorá reprezentuje štvorec, od triedy, ktorá reprezentuje obdĺžnik?

(a) nie, lebo to porušuje princíp zapuzdrenia
(b) áno
(c) nie, lebo to porušuje Liskovej princíp substitúcie
(d) nie, lebo to porušuje princíp otvorenosti a uzavretosti
(e) áno, ale musíme zabezpečiť, aby sa pri štvorci šírka a výška rovnali

16. (3 b) Daný je kód v Jave na obr. 2. Vykonaním týchto príkazov pre objekt m triedy M:

```
A o1 = (A)m.f(0);
A o2 = (A)m.f(1);
A o3 = m.f(0);
```

(a) kompilátor hlási chybu na druhom riadku; po jej odstránení počas vykonávania vznikne výnimka na treťom riadku
(b) kompilátor hlási chybu na treťom riadku; po jej odstránení počas vykonávania vznikne výnimka na druhom riadku
(c) kompilátor hlási chybu na druhom a treťom riadku
(d) počas vykonávania vznikne výnimka na druhom a treťom riadku
(e) nevzniknú žiadne chyby

```
interface I { }
class A implements I { }
class B implements I { }
class M {
    I f(int a) {
        if (a == 0)
            return new A();
        else
            return new B();
    }
}
```

Obrázok 2: Kód pre otázky 16 a 17.

17. (5 b) Nakreslite diagram tried v UML pre kód na obr. 2.

18. (12 b) Stopky umožňujú spustenie merania času, zastavenie merania a jeho vynulovanie. Merajú s presnosťou na sekundu. Implementujte ich ako triedu, v ktorej čas bude uchovaný v sekundách. Predpokladajte, že sa posun času uskutočňuje volaním metódy, ktorá aktualizuje uchovanú hodnotu. Táto metóda bude zaradená na vykonávanie každú sekundu do plánovača, čo nie je predmetom tejto úlohy.

Čas je potrebné zobraziť súčasne vo viacerých verziách, z ktorých každá má definovaný časový posun vzhľadom na samotné stopky (napr. + 10 s alebo − 255 s). Každé zobrazenie je v tvare hh:mm:ss alebo len v sekundách. V budúcnosti bude potrebné pridať nové druhy zobrazenia (napr. klasickými hodinkami).

Napíšte relevantný kód v Jave, ktorý zodpovedá týmto požiadavkám. Aplikujte pritom mechanizmy objektovo-orientovaného programovania v maximálnej miere a vysvetlite ich úlohu. Samotný výpis času stačí realizovať metódou println().

Objektovo-orientované programovanie **A**
Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška — 4. jún 2008

1 c

2 b

3 e

4 a

—

5 a

6 d

7 c

8 c

9 a

10 b

11 a

—

12 d

13 a

14 e

15 c

16 b

50