Objektovo-orientované programovanie □
Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška — 15. jún 2009

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 90 minút.
V otázkach 1–16 je len jedna možnosť správna. Vyznačte svoju odpoveď krížikom do tabuľky. Hodnotia sa len odpovede v tabuľke.
V prípade opravy jasne vyznačte odpoveď, ktorá platí. 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. Poškodený list nebude uznaný.
Odpovede na otázky 17 a 18 píšte na prídavný list. Na ňom tiež uveďte svoje priezvisko a meno.

1. (1 b) Objekt v objektovo-orientovanom programovaní predstavuje

(a) triedu
(b) inštanciu triedy alebo rozhrania
(c) modul
(d) inštanciu triedy
(e) typ

2. (1 b) Pri dedení je možné v odvodenej triede zadefinovať

(a) len metódy, ktorých názvy nejestvujú v nadtriede
(b) len metódy, ktorých názvy jestvujú v nadtriede
(c) ďalšie atribúty a metódy
(d) len ďalšie atribúty
(e) len ďalšie metódy

3. (1 b) Abstraktná trieda v Jave

(a) môže mať statické metódy
(b) nemôže dediť
(c) nemôže mať prekonávajúce metódy
(d) môže mať len abstraktné metódy
(e) nemôže mať atribúty

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

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

5. (2 b) V porovnaní s objektovo-orientovaným programovaním aspektovo-orientované programovanie umožňuje

(a) rozdelenie kódu do komponentov
(b) oddelenie pretínajúcich záležitostí
(c) tvorbu modulov
(d) zlúčenie pretínajúcich záležitostí
(e) rýchlejšie vykonávanie programu

6. (2 b) Generickým triedam v Jave v jazyku C++ zodpovedajú

(a) abstraktné triedy
(b) štruktúry (struct)
(c) virtuálne triedy
(d) iterátory
(e) šablóny (template)

7. (2 b) Nech o je objekt triedy, ktorá poskytuje verejnú metódu **int** m(). Pole r je definované takto:

Object r[] = **new** Object[o.m()];

Táto definícia je

(a) korektná jedine ak je metóda m() synchronizovaná
(b) korektná
(c) nekorektná
(d) korektná jedine ak je metóda m() finálna
(e) korektná jedine ak je metóda m() statická

8. (2 b) Metóda f() triedy A vyhadzuje výnimku typu MyException. Daná je trieda B:

```
class B {
   void m() { new A().f(); }
}
```

Metóda m() triedy B

(a) je korektná
(b) musí ošetrovať výnimku typu MyException
(c) musí vyhadzovať výnimku typu MyException
(d) musí deklarovať že vyhadzuje výnimku typu MyException
(e) musí deklarovať alebo ošetrovať výnimku typu MyException

9. (2 b) Diagram objektov v jazyku UML predstavuje jeden

(a) z dynamických pohľadov
(b) zo štrukturálnych pohľadov
(c) z pohľadov interakcie
(d) z pohľadov rozloženia
(e) z pohľadov správania

10. (2 b) Zapuzdrenie v objektovo-orientovanom programovaní

(a) umožňuje, aby sa objekt uplatnil namiesto objektu jeho nadtypu
(b) predstavuje spôsob tvorenia hierarchie
(c) predstavuje kritérium pre použitie agregácie
(d) umožňuje znížiť závislosť klientskeho kódu
(e) umožňuje spájanie objektov

11. (2 b) V jazyku AspectJ je pomocou videnia (advice) možné

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

12. (3 b) Návrhový vzor Strategy je vo vzore Model-View-Controller využitý vo vzťahu

(a) View-Controller
(b) Model-Controller
(c) tried vo vnútri časti Controller
(d) Model-View
(e) tried vo vnútri časti View

13. (3 b) Ku kódu v Jave na obr. 1 je daná nasledujúca trieda:

```
class M {
    static int m(Class<? extends A> T, A... o) {
        int i = 0;
        for (A e : o)
            if (T.isInstance(e))
                i++;
        return i;
    }

    public static void main(String... args) {
        System.out.println(
            m(X.class, new A[]{new X(), new Y()}));
    }
}
```

Pri jej vykonaní

(a) vypíše sa 3
(b) vznikne výnimka
(c) vypíše sa 1
(d) vypíše sa 2
(e) vypíše sa 0

14. (3 b) Daný je kód v Jave na obr. 1. Vykonaním týchto príkazov:

```
A o = new X();
o.m();
((X)o).m();
((A)o).m();
```

(a) vznikne chyba v predposlednom riadku
(b) atribút i nadobudne hodnotu 1
(c) atribút i nadobudne hodnotu 3
(d) atribút i nadobudne hodnotu 2
(e) vznikne chyba v poslednom riadku

```
abstract class A {
    public abstract void m();
}

class X extends A {
    int i = 0;
    public void m() { i++; }
}

class Y extends X {
    int i = 0;
    public void m() { i++; }
}
```

Obrázok 1: Kód pre otázky 13 a 14.

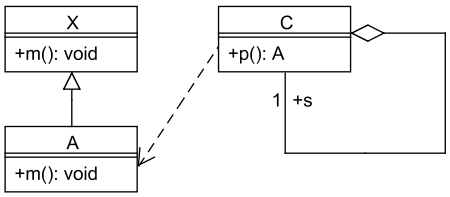
15. (3 b) V jazyku C++ ste implementovali komplexné čísla vlastnou triedou. Na zjednodušenie zápisu matematických operácií s komplexnými číslami je vhodné použiť

(a) virtuálne funkcie
(b) statické funkcie
(c) preťaženie funkcií
(d) virtuálne operátory
(e) preťaženie operátorov

16. (3 b) Vytvorili ste tlačidlo t ako komponent rámca Swing a zviditeľneli ste okno, ktoré ho obsahuje. Potrebujete zmeniť označenie (label) tlačidla na text Abc. Urobíte to volaním

(a) t.invokeLater(t.changeText("Abc");}});
(b) (**new** Runnable(SwingUtilities.invokeLater(
t.changeText("Abc");}}))).run();
(c) SwingUtilities.invokeLater(**new** Runnable() {
public void run() {t.changeText("Abc");}});
(d) SwingUtilities.run(**new** Runnable() {
public void invokeLater() {
t.changeText("Abc");}});
(e) t.changeText("Abc")

17. (5 b) Prepíšte diagram na obr. 2 do kódu.



Obrázok 2: Diagram pre otázku 17.

18. (12 b) Grafický program v rozvoji umožňuje zatiaľ prácu s kruhmi, obdĺžnikmi a štvorcami. Každému grafickému objektu je možné meniť veľkosť a farbu. Uveďte základ kódu v Jave s príkladom klientskeho kódu. Aplikujte pritom mechanizmy objektovo-orientovaného programovania v maximálnej miere. Zdôvodnite svoj návrh.

Objektovo-orientované programovanie **A**
Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška — 26. máj 2009

1 d

2 c

3 a

4 c

—

5 b

6 e

7 b

8 e

9 b

10 d

11 b

—

12 a

13 d

14 c

15 e

16 c

50