Objektovo-orientované programovanie **A**

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

Skúška – 27. máj 2011

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					
17					

Skúška trvá 75 minút. V otázkach 1–17 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 1–17 nehodnotí. Poškodený list nebude uznaný. Odpoveď na otázku 18 píšte na prídavný list.

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

(a) príkazom System.open()
(b) jeho metódou open()
(c) jeho prvým použitím
(d) jeho konštruktorom
(e) príkazom IOStream.open()

2. (1 b) Príkaz

import java.util.∗;

(a) fyzicky pripojí typy balíka java.util k programu bez typov v podbalíkoch
(b) fyzicky pripojí len skutočne použité typy balíka java.util k programu bez typov v podbalíkoch
(c) sprístupní priestor názvov všetkých typov balíka java.util vrátane typov v podbalíkoch
(d) fyzicky pripojí všetky typy balíka java.util k programu vrátane typov v podbalíkoch
(e) sprístupní priestor názvov všetkých typov balíka java.util, ale bez typov v podbalíkoch

3. (1 b) Abstraktná metóda v Jave

(a) nemôže byť statická
(b) nemôže mať návratovú hodnotu
(c) môže mať telo
(d) môže sa vyskytovať v hocijakej triede
(e) nemôže mať argumenty

4. (1 b) Atribút triedy, ktorému predchádza kľúčové slovo **private**

(a) sa nezapíše do súboru pri serializácii objektu
(b) je dostupný len v danej hierarchii tried
(c) je dostupný len v rámci jednej nite
(d) je dostupný len v danej triede
(e) je chránený pred zápisom

5. (2 b) Návrhový vzor Strategy slúži na

(a) pridávanie vzťahov medzi triedami bez ich zmeny
(b) definovanie závislosti stavu viacerých objektov od ďalšieho objektu
(c) zabránenie vytvárania viac než jednej inštancie danej triedy
(d) pridávanie operácií nad objektmi daných tried bez ich zmeny
(e) zastrešenie rôznych spôsobov riešenia toho istého problému

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

(a) prenos parametrov operácie v smere šípky
(b) zľava doprava vyvolanie operácie, a sprava doľava návrat hodnoty
(c) vyvolanie operácie
(d) tok údajov medzi objektmi
(e) návrat hodnoty

7. (2 b) Kód

```
BufferedReader stdin =
    new BufferedReader(
        new InputStreamReader(System.in));
```

zabezpečuje

(a) pohodlnejšiu prácu so štandardným vstupom
(b) otvorenie štandardného vstupu
(c) vytvorenie a otvorenie štandardného vstupu
(d) čítanie zo štandarného vstupu
(e) vytvorenie štandardného vstupu

8. (2 b) Daný je nasledujúci kód:

```
List<R> z = new ArrayList<R>();
z.add(new X());
z.add(new Y());
```

Aby sa tento kód mohol preložiť a vykonať

(a) R musí byť abstraktná trieda, a typy X a Y triedy od nej odvodené
(b) typy X a Y musia byť triedy odvodené od typu R, ale X nesmie byť odvodené od Y alebo naopak
(c) typy X a Y musia byť triedy priamo alebo nepriamo odvodené od typu R
(d) typ X musí byť odvodený od typu Y alebo naopak
(e) R musí byť rozhranie, a typy X a Y triedy, ktoré ho implementujú

9. (2 b) Výraz call(abc∗(..)) v jazyku AspectJ znamená

(a) vyvolanie metódy call() pred všetkými metódami, ktorých názov začína na abc
(b) vyvolanie prvej metódy, ktorej názov začína na abc
(c) vyvolanie všetkých metód, ktorých názov začína na abc
(d) zachytenie prvého volania metódy, ktorej názov začína na abc
(e) zachytenie volania všetkých metód, ktorých názov začína na abc

10. (2 b) Metóda f() triedy A vyhadzuje výnimku 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 deklarovať, že vyhadzuje výnimku typu MyException
(d) musí deklarovať alebo ošetrovať výnimku typu MyException
(e) musí vyhadzovať výnimku typu MyException

11. (2 b) Mechanizmus friend v jazyku C++

(a) umožňuje poskytnúť prístup k neverejným prvkom triedy iným triedam
(b) umožňuje poskytnúť prístup k neverejným prvkom triedy cudzím funkciám
(c) sa používa na označenie user-friendly funkcionality
(d) nejestvuje
(e) slúži na zapínanie a vypínanie polymorfizmu

12. (2 b) Synchronizácia statickej metódy

(a) znamená uzamknutie objektu triedy pre hocijaký iný prístup
(b) nie je možná
(c) znamená uzamknutie objektu triedy pre hocijaký iný synchronizovaný prístup
(d) znamená uzamknutie objektu **this** pre hocijaký iný synchronizovaný prístup
(e) znamená uzamknutie objektu **this** pre hocijaký iný prístup

13. (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) atribút i nadobudne hodnotu -1
(b) vznikne chyba v poslednom riadku
(c) atribút i nadobudne hodnotu 3
(d) atribút i nadobudne hodnotu 1
(e) atribút i nadobudne hodnotu -3

```
abstract class A {
    int i = 0;
    public abstract void m();
}

class X extends A {
    public void m() { i--; }
}

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

Obr. 1: Kód pre otázky 13 a 17.

14. (3 b) Daný je nasledujúci kód v Jave:

```
if (o.class == "Kruh")
    ((Kruh)o).nakresli();
else if (o.class == "Stvorec")
    ((Stvorec)o).nakresli();
else
    ;
```

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

15. (3 b) Pre architektonický vzor Model–View–Controller nie je významný ani jeden z návrhových vzorov
(a) Visitor a Worker Object Creation
(b) Observer a Strategy
(c) Composite a Strategy
(d) Composite a Worker Object Creation
(e) Observer a Composite

16. (3 b) Daný je nasledujúci kód v Jave:

```
class A {
    public void m() { System.out.print("a"); }
}

class B extends A { }

class C extends B {
    public void m() { System.out.print("c"); }
}
```

Čo sa vypíše po vykonaní týchto príkazov:

```
A b = new B();
A c = new C();

b.m();
((A)b).m();
c.m();
((B)c).m();
```

(a) cc
(b) aaca
(c) aacc
(d) aaaa
(e) acc

17. (3 b) Ku kódu v Jave na obr. 1 je daná nasledujúca trieda:

```
class M {
    static void m(Class<? extends A> T, A... o) {
        for (A e : o)
            if (T.isInstance(e))
                System.out.print("a");
    }

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

Jej vykonaním
(a) vypíše sa a
(b) vznikne výnimka
(c) vypíše sa aaa
(d) vypíše sa aa
(e) nevypíše sa nič

Priezvisko:

Meno:

18. (10 b) V informačnom systéme softvérovej firmy sa vedie evidencia o projektoch, oddeleniach a vývojových skupinách. Pre evidované projekty, oddelenia a vývojové skupiny systém umožňuje výpis niektorých spoločných charakteristík ako sú napríklad názov, zoznam pracovníkov, ktorých zahŕňajú, začiatok a koniec realizácie/existencie atď. V ďalších verziách možno vznikne potreba pridania nových spoločných operácií.

Nakreslite diagram tried s najvýznamnejšími vzťahmi, operáciami a atribútmi, ktoré vyplývajú z uvedeného opisu. Napíšte zodpovedajúci kód v Jave. Aplikujte mechanizmy objektovo-orientovaného programovania a vysvetlite ich úlohu. Ak je to vhodné, aplikujte niektorý z návrhových vzorov.

Objektovo-orientované programovanie **A**
Ing. Valentino Vranić, PhD., ÚISI FIIT STU
Skúška – 27. máj 2011

1 d

2 e

3 a

4 d

—

5 e

6 c

7 a

8 c

9 e

10 d

11 b

12 c

—

13 e

14 b

15 a

16 c

17 d

45