

Priezvisko:

Meno:

1 b	
2 b	
3 b	

Skúška trvá 90 minút.

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

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 píšete na prídavný list. Na ňom tiež uveďte svoje priezvisko a meno.

1. (1 b) Daný je nasledujúci kód:

```
for (Object o : l) {  
    if (o.class == "StrasnyObor")  
        ((StrasnyObor)o).utoc();  
    else if (o.class == "PlachyObor")  
        ((PlachyObor)o).utec();  
    else  
        ;  
}
```

Tento kód porušuje

- (a) princíp polymorfizmu
- (b) Liskovej princíp substitúcie
- (c) princíp generalizácie a špecializácie
- (d) princíp abstrakcie
- (e) princíp otvorenosti a uzavretosti kódu

2. (1 b) Objekt v objektovo-orientovanom programovaní predstavuje

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

3. (1 b) Ak trieda v jazyku C++ obsahuje čisto virtuálne funkcie, potom

- (a) nemôže mať inštalácie
- (b) nemôže dediť od triedy, ktorá ich neobsahuje
- (c) môže dediť od inej triedy len virtuálne
- (d) sa od nej dá dediť len virtuálne
- (e) nepodporuje polymorfizmus

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

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

5. (2 b) Dedeniu typu extend medzi triedami v Jave v jazyku C++ zodpovedá dedenie typu:

- (a) **private**
- (b) **protected** s virtuálnymi triedami
- (c) **protected**
- (d) **public** s virtuálnymi triedami
- (e) **public**

6. (2 b) V rámci Swing sa kliknutie na tlačidlo v okne sleduje

- (a) volaním metódy actionPerformed() príslušného tlačidla v slučke
- (b) metódou onClick() tlačidla implementovanou pri odvodení od všeobecného tlačidla JButton
- (c) prijímačom udalosti kliknutia registrovaným pre dané tlačidlo
- (d) zavolaním statickej metódy EventQueue.onClick() a následným zistením, či sa kliknutie vzťahuje na dané tlačidlo
- (e) automaticky po pridaní tlačidla do okna jeho metódou add()

7. (2 b) Na rozdiel od objektovo-orientovaného programovania aspektovo-orientované programovanie umožňuje

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

8. (2 b) Diagram sekvencií v jazyku UML znázorňuje

- (a) vzťahy medzi inštanciami tried v určitom okamihu vykonávania programu
- (b) triedy a vzťahy medzi nimi
- (c) funkcionality z pohľadu používateľa
- (d) postupnosť volaní medzi objektmi
- (e) štruktúru systému

9. (2 b) Princíp otvorenosti a uzavretosti hovorí, že

- (a) kód má byť otvorený pre zmeny, ale uzavretý pre rozšírenie
- (b) každý prúd údajov okrem štandardného vstupu a výstupu je potrebné po otvorení aj zatvoriť
- (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 je potrebné po otvorení aj zatvoriť

10. (2 b) Synchronizácia statickej metódy

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

11. (2 b) Pri rozhodovaní o použití dedenia je kľúčové sledovať

- (a) možnosť uplatnenia objektov podtypu namiesto objektov nadtypu
- (b) možnosti zovšeobecnovania podtypu nadtypom
- (c) možnosti abstrakcie podtypu nadtypom
- (d) počet spoločných atribútov a metód podtypu a nadtypu
- (e) mieru príbuznosti objektov modelovaných podtypom a nadtypom v realite

12. (3 b) Implementácie rozhraní prijímačov vo Swingu predstavujú

- (a) pozorovateľov vo vzore Observer
- (b) návštevníkov vo vzore Visitor
- (c) prvky vo vzore Visitor
- (d) konkrétne stratégie vo vzore Strategy
- (e) kontexty vo vzore Strategy

13. (3 b) Po spustení kódu z obr. 1

- (a) vypíše sa xy a program vyhodí výnimku
- (b) vypíše sa yyy
- (c) vznikne výnimka
- (d) vypíše sa yyx
- (e) vypíše sa yy a vznikne výnimka

```
class X {
    public void m() { System.out.print("x"); }
}
class Y extends X {
    public void m() { System.out.print("y"); }
}
class A {
    public Runnable r(final X x) {
        return new Runnable(){
            public void run(){
                x.m();
            }
        };
    }
    public static void main(String[] args) {
        X ox = new Y();
        Y oy = new Y();

        Runnable a = (new A()).r(ox);
        Runnable b = (new A()).r(oy);
        Runnable c = (new A()).r((X)oy);
        a.run();
        b.run();
        c.run();
    }
}
```

Obrázok 1: Kód pre otázky 13 a 15.

14. (3 b) V jednej z tried programu sa realizuje výpočet na základe dodaných koeficientov. V prípade ich nesúladu, ktorý sa dá zistiť ich porovnaním, výpočet nemá zmysel. Toto treba ošetriť

- (a) priamo v metóde, ktorá realizuje výpočet, zastavením programu pre chybu
- (b) vyhodnením všeobecnej výnimky typu Exception a jej ošetrovaním v metóde, ktorá výpočet potrebovala
- (c) vyhodnením výnimky typu vytvoreného špeciálne pre za týmto účelom a jej ošetrovaním v metóde, ktorá výpočet potrebovala
- (d) priamo v metóde, ktorá realizuje výpočet, pokusom o nápravu výsledku
- (e) vyhodnením výnimky typu RuntimeException

15. (3 b) Ku kódu v Jave na obr. 1 je daná nasledujúca trieda:

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

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

Jej vykonaním

- (a) vypíše sa ii
- (b) vznikne výnimka
- (c) vypíše sa i
- (d) nevypíše sa nič
- (e) vypíše sa iii

16. (3 b) V aspektovo-orientovanej implementácii vzoru Observer

- (a) vyčleňuje sa do aspektu logika vzoru, ktorá je predovšetkým v časti Observer
- (b) vzniká tretia časť vzoru označovaná ako Mediator medzi časťami Subject a Observer
- (c) vyčleňuje sa do aspektu logika vzoru, ktorá je predovšetkým v časti Subject
- (d) zaniká potreba implementovať Subject
- (e) oddeľuje sa aplikačná logika od rozhrania aplikácie

17. (5 b) Nakreslite diagram tried v UML pre kód na obr. 1 rozšírený o nasledujúcu triedu:

```
class C extends X {
    private X x;
    public void setX(X x) { this.x = x; }
    public X getX() { return x; }
}
```

18. (12 b) Aplikácia umožňuje šifrovanie rôznych druhov súborov. Predpokladajme binárne a textové súbory. Pri textových ešte treba brať do úvahy kódovanie – predpokladajme ASCII a Unicode. Aplikácia poskytuje dva spôsoby šifrovania pre všetky podporené typy súborov. Aplikácia v budúcnosti bude podporovať ďalšie spôsoby šifrovania a možno aj ďalšie druhy súborov.

Napíšte kód v Jave, ktorý implementuje súbory a šifrovanie. Aplikujte pritom mechanizmy objektovo-orientovaného programovania v maximálnej miere a vysvetlite ich úlohu. Samotné spôsoby šifrovania a operácie so súbormi neuvádzajte.

1 e

2 b

3 a

4 d
—

5 d

6 c

7 e

8 d

9 c

10 b

11 a
—

12 d

13 b

14 c

15 a

16 c

50