

2020/21

Čo sa stane ak sa pokúsime skompilovať a spustiť nasledovný kód? (Poznámka: Number je predok Integer.)

Čo sa stane ak sa pokúsime skompilovať a spustiť nasledovný kód? (Poznámka: Number je predok Integer.)

```
1. public class Testing {
2.     public static void main(String[] args) {
3.         List<? super Integer> sList = new ArrayList<Number>();
4.         int i = 2007;
5.         sList.add(i);
6.         sList.add(++i);
7.         Number num = sList.get(0);
8.     }
9. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Kód sa neskompiluje kvôli chybe na riadku 3.
- ☐ b. Kód sa skompiluje, ale po spustení program vyhodí výnimku ClassCastException na riadku 7.
- ☐ c. Kód sa skompiluje a spustí normálne.
- ☐ d. Kód sa neskompiluje kvôli chybe na riadku 6.
- ☒ e. Kód sa neskompiluje kvôli chybe na riadku 7.



Trieda GenericFruit deklaruje nasledovnú metódu:

Public void setCalorieContent(float f)

Pri vytváraní triedy Apple odvodenej z triedy GenericFruit potrebujeme pridať metódy, ktoré preťažujú (overload) túto metódu. Ktoré z nasledovných deklarácií metód sú pre tento prípad použiteľné?

Trieda GenericFruit deklaruje nasledovnú metódu:

public void setCalorieContent(float f)

Pri vytváraní triedy Apple odvodenej z triedy GenericFruit potrebujeme pridať metódy, ktoré preťažujú (overload) túto metódu. Ktoré z nasledovných deklarácií metód sú pre tento prípad použiteľné?

Označte jednu alebo viac odpovedí:

- ☐ a. protected void setCalorieContent(float x)
- ☒ b. protected float setCalorieContent(String s)
- ☒ c. public void setCalorieContent(String s) throws NumberFormatException
- ☒ d. public void setCalorieContent(double d)



Kde je chyba v nasledovnom kóde?

(Public class MyClass)

Kde je chyba v nasledovnom kóde?

```
1. public class MyClass {
2.     public static void main(String[] args) throws A {
3.         try {
4.             f();
5.         } finally {
6.             System.out.println("Done.");
7.         } catch (A e) {
8.             throw e;
9.         }
10.    }
11.    public static void f() throws B {
12.        throw new B();
13.    }
14. }
15. class A extends Throwable {}
16. class B extends A {}
```

Označte jednu alebo viac odpovedí:

- ☐ a. V metóde main() musí catch blok deklarovať, že odchyťava výnimky typu B a nie typu A.
- ☐ b. Deklarácia triedy A nie je správna.
- ☐ c. Metóda main() musí deklarovať, že vyhadzuje výnimku typu B.
- ☐ d. Za blok try nesmie nasledovať zároveň obidva bloky finally aj catch.
- ☒ e. V metóde main() musí blok finally nasledovať za blokom catch.



Aký bude výsledok po skompilovaní a spustení nasledovného programu? (Class Vehicle)

Aký bude výsledok po skompilovaní a spustení nasledovného programu?

```
1. class Vehicle { }
2. class Car extends Vehicle { }
3. class Sedan extends Car { }
4. class Garage<V> {
5.     private V v;
6.     public V get() { return this.v; }
7.     public void put(V v) { this.v = v; }
8. }
9. public class GarageAdmin {
10.    private Object object = new Object();
11.    private Vehicle vehicle = new Vehicle();
12.    private Car car = new Car();
13.    private Sedan sedan = new Sedan();
14.    public void doD(Garage<? extends Car> g) {
15.        g.put(object);
16.        g.put(vehicle);
17.        g.put(car);
18.        g.put(sedan);
19.        object = g.get();
20.        vehicle = g.get();
21.        car = g.get();
22.        sedan = g.get();
23.    }
24. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Volanie metódyput()na riadkoch 15 a 16 sa skompilujú.
- ☒ b. Priradenia na riadkoch 19, 20 a 21 sa skompilujú.
- ☐ c. Volanie metódyput()na riadkoch 17 a 18 sa skompilujú.
- ☐ d. Priradenie na riadku 22 sa skompiluje.



Aký bude výsledok po skompilovaní a spustení nasledovného kódu? (public class ExIn)

Aký bude výsledok po skompilovaní a spustení nasledovného kódu?

```
1. import java.io.*;
2. public class ExIn {
3.     public static void main(String argv[]) {
4.         ExIn ei = new ExIn();
5.         System.out.println(ei.getCharacter());
6.     }
7.     public char getCharacter() {
8.         FileInputStream fis = null;
9.         try {
10.            fis = new FileInputStream("file.txt");
11.            DataInputStream dis = new DataInputStream(fis);
12.            char c = dis.readChar();
13.            return 1;
14.        } catch (IOException ioe) {
15.            System.out.println(ioe.getMessage());
16.            return 2;
17.        } catch (Exception e){
18.            System.out.println(e.getMessage());
19.        } finally {
20.            try {
21.                fis.close();
22.            } catch (Exception ec) {
23.                System.out.println("test fis.close()" + ec.getMessage());
24.            }
25.        }
26.        return c;
27.    }
28. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Kompilácia prebehne bez chyby.
- ☐ b. Kompilátor vyhlási chybu, pretože blok finally nemôže obsahovať vnorený try/catch blok (riadok 20).
- ☐ c. Kompilátor vyhlási chybu, pretože príkazy return vracajú čísla namiesto typu char (riadky 13 a 16).
- ☒ d. Kompilátor vyhlási chybu kvôli príkazu return c; (riadok 26).



Majme nasledovné triedy a deklarácie objektov. Ktoré z uvedených tvrdení sú správne? (class Foo)

Majme nasledovné triedy a deklarácie objektov. Ktoré z uvedených tvrdení sú správne?

```
1. class Foo {
2.     private int i;
3.     public void f() { }
4.     public void g() { }
5. }
6. class Bar extends Foo {
7.     public int j;
8.     public void g() { }
9. }
10. Foo a = new Foo();
11. Bar b = new Bar();
```

Označte jednu alebo viac odpovedí:

- ☒ a. Príkaz a.g(); je správny.
- ☒ b. Príkaz b.f(); je správny.
- ☐ c. Príkaz a.j = 5; je správny.
- ☐ d. Príkaz b.i = 3; je správny.
- ☒ e. Trieda Bar je podtriedou triedy Foo.




Aký bude výsledok po skompilovaní nasledovného programu? (public class MyClass)

Aký bude výsledok po skompilovaní nasledovného programu?

```
1. public class MyClass {
2.     long var;
3.     public void MyClass(long param) { var = param; }
4.     public static void main(String[] args) {
5.         MyClass a, b;
6.         a = new MyClass();
7.         b = new MyClass(5);
8.     }
9. }
```

Označte jednu alebo viac odpovedí:


- ☐ a. Kompilátor zahlásí chybu na riadku 6, pretože trieda nemá definovaný konštruktor bez parametrov.
- ☒ b. Kompilátor zahlásí chybu na riadku 7, pretože trieda nemá konštruktor s parametrom typu int. 
- ☐ c. Program sa skompiluje bez chyby.
- ☐ d. Kompilátor zahlásí chybu na riadku 3, pretože konštruktor nemôže špecifikovať návratovú hodnotu.

Aký bude výsledok po skompilovaní nasledovného kódu? (public class Testing)

Aký bude výsledok po skompilovaní nasledovného kódu?

```
1. public class Testing {
2.     public static void main(String[] args) {
3.         List<Integer> glst1 = new ArrayList();
4.         List nglst1 = glst1;
5.         List nglst2 = nglst1;
6.         List<Integer> glst2 = glst1;
7.     }
8. }
```


Označte jednu alebo viac odpovedí:

- ☐ a. Kód sa skompiluje s varovnou správou na riadku 5.
- ☐ b. Kód sa skompiluje s varovnou správou na riadku 6.
- ☒ c. Kód sa skompiluje s varovnou správou na riadku 3. 
- ☐ d. Kód sa skompiluje s varovnou správou na riadku 4.
- ☐ e. Kód sa skompiluje bez varovných správ.

Ako navzájom komunikujú objekty v objektovo orientovanom jazyku akým je napr. Java?

Ako navzájom komunikujú objekty v objektovo orientovanom jazyku akým je napr. Java?

Označte jednu alebo viac odpovedí:

- ☐ a. Komunikujú vzájomným volaním svojich statických metód.
- ☒ b. Komunikujú vzájomným volaním svojich členských metód. 
- ☐ c. Komunikujú vzájomnou modifikáciou svojich statických premenných.
- ☐ d. Komunikujú vzájomnou modifikáciou svojich členských premenných.

Majme nasledovné deklarácie rozhraní, tried a referencií: (interface I1)

Majme nasledovné deklarácie rozhraní, tried a referencií:

```
interface I1 {}  
interface I2 {}  
class C1 implements I1 {}  
class C2 implements I2 {}  
class C3 extends C1 implements I2 {}  
C1 obj1;  
C2 obj2;  
C3 obj3;
```

Vyberte, ktoré z nasledovných priradení sú správne:

Označte jednu alebo viac odpovedí:

- ☐ a. I1 a = obj2;
- ☒ b. I1 b = obj3;
- ☐ c. obj3 = obj2;
- ☐ d. obj3 = obj1;
- ☐ e. obj2 = obj1;
- ☐ f. I2 c = obj1;



Ktoré z tvrdení sú pravdivé pre nasledovný kód? (class Fruit)

Otázka 7

Správna
odpoveď

Známka 3,00 z
3,00

🚩 Označiť
otázku

Ktoré z tvrdení sú pravdivé pre nasledovný kód?

```
1. class Fruit {}  
2. class Apple extends Fruit {}  
3. public class TestApp {  
4.     public static void main(String[] args) {  
5.         List<? extends Apple> lst1 = new ArrayList<Fruit>();  
6.         List<? extends Fruit> lst2 = new ArrayList<Apple>();  
7.         List<? super Apple> lst3 = new ArrayList<Fruit>();  
8.         List<? super Fruit> lst4 = new ArrayList<Apple>();  
9.         List<?> lst5 = lst1;  
10.        List<?> lst6 = lst3;  
11.        List lst7 = lst6;  
12.        List<?> lst8 = lst7;  
13.    }  
14. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Riadok 11 sa skompiluje, riadok 12 nie.
- ☐ b. Riadok 5 sa skompiluje, riadok 6 nie.
- ☐ c. Ani jedna odpoveď nie je správna.
- ☐ d. Riadok 9 sa skompiluje, riadok 10 nie.
- ☒ e. Riadok 7 sa skompiluje, riadok 8 nie.

Keď sa pokúsite skompilovať nasledovný kód, objaví sa správa, že premenná tmp nie je inicializovaná: (class Demo)

Otázka 8

Čiastočne
správna
odpoveď

Známka 2,25 z
3,00

Označiť
otázku

Keď sa pokúsite skompilovať nasledovný kód, objaví sa správa, že premenná tmp nie je inicializovaná:

```
1. class Demo {  
2.     String msg = "Type is ";  
3.     public void showType(int n) {  
4.         String tmp;  
5.         if (n > 0) tmp = "positive";  
6.         System.out.println(msg + tmp);  
7.     }  
8. }
```

Ktoré z nasledujúcich úprav umožnia eliminovať túto správu kompilátora?

Označte jednu alebo viac odpovedí:

- ☒ a. Zmeniť riadok 4 na: String tmp = "";
- ☒ b. Presunúť riadok 4 a vložiť ho medzi riadky 2 a 3 (čím sa tmp stane členskou premennou).
- ☒ c. Vložiť nový riadok medzi riadky 5 a 6: else tmp = "not positive";
- ☒ d. Zmeniť riadok 4 na: String tmp = null;



Majme vlastnú hierarchiu výnimiek odvodenú od java.lang.Exception nasledovne (BitterException a SourException sú potomkami BadTasteException): (TexMexCook)

Otázka 9

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Majme vlastnú hierarchiu výnimiek odvodenú od java.lang.Exception nasledovne (BitterException a SourException sú potomkami BadTasteException):

```
Exception  
+-- BadTasteException  
    +-- BitterException  
    +-- SourException
```

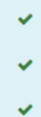
Vaša trieda, napr. BaseCook, má metódu deklarovanú nasledovne:

```
int rateFlavor(Ingredient[] list) throws BadTasteException
```

Trieda TexMexCook, zdedená z BaseCook ma metódu prekryvajúcu rateFlavor(). Ktoré z nasledovných deklarácií prekrytej metódy sú správne?

Označte jednu alebo viac odpovedí:

- ☒ a. int rateFlavor(Ingredient[] list) throws BadTasteException
- ☒ b. int rateFlavor(Ingredient[] list) throws BitterException
- ☒ c. int rateFlavor(Ingredient[] list)
- ☐ d. int rateFlavor(Ingredient[] list) throws Exception



Aký bude výsledok po pokuse skompilovať a spustiť nasledovný kód? (Poznámka: Number je predok Integer.) (public class TestList)

Otázka 3

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Aký bude výsledok po pokuse skompilovať a spustiť nasledovný kód? (Poznámka: Number je predok Integer.)

```
1. public class TestList {  
2.     public static void main(String[] args) {  
3.         List<Integer> lst = new ArrayList<Integer>();  
4.         lst.add(2007);  
5.         lst.add(2008);  
6.         List<Number> numList = lst;  
7.         for (Number n : numList)  
8.             System.out.println(n + " ");  
9.     }  
10. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Nastane chyba pri kompilácii na riadku 7.
- ☐ b. Program sa skompiluje, ale počas vykonávania sa vyhodí výnimka ClassCastException na riadku 6.
- ☐ c. Program sa skompiluje a po spustení vypíše "2007 2008 ".
- ☒ d. Nastane chyba pri kompilácii na riadku 6.
- ☐ e. Nastane chyba pri kompilácii na riadku 3.



Ktoré z deklarácií členskej premennej accounts je možné použiť na riadku 2 uvedeného kódu?

Otázka 3
Ešte nezodpovedané
Max. hodnotenie 3,00
Označiť otázku

Ktoré z deklarácií členskej premennej accounts je možné použiť na riadku 2 uvedeného kódu?

```
1. public class Testing {  
2.     // VLOŽTE SPRÁVNÚ DEKLARÁCIU ČLENSKEJ PREMENNEJ  
3.     public long getNum(String name) {  
4.         long number = accounts.get(name);  
5.         return number == null ? 0 : number;  
6.     }  
7.     public void setNum(String name, long number) {  
8.         accounts.put(name, number);  
9.     }  
10. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. private Map<String, Long> accounts = new HashMap<String, Long>();
- ☐ b. private Map<String<Long>> accounts = new HashMap<String<Long>>();
- ☐ c. private Map<String, long> accounts = new HashMap<String, long>();
- ☐ d. private Map accounts = new HashMap();
- ☐ e. private Map<String, Long> accounts = new Map<String, Long>();

A je správne

Ktoré z nasledujúcich deklarácií členskej premennej accounts je možné použiť na riadku 2 uvedeného kódu?

Správna odpoveď je: private Map<String, Long> accounts = new HashMap<String, Long>();

Označiť

Aký bude výsledok po skompilovaní a spustení nasledovného programu?
(public class MyClass)

Aký bude výsledok po skompilovaní a spustení nasledovného programu?

```
1. public class MyClass {  
2.     public static void main(String[] args) {  
3.         C c = new C();  
4.         System.out.println(c.max(13, 29));  
5.     }  
6. }  
7. class A {  
8.     int max(int x, int y) { if (x > y) return x; else return y; }  
9. }  
10. class B extends A {  
11.     int max(int x, int y) { return super.max(y, x) - 10; }  
12. }  
13. class C extends B {  
14.     int max(int x, int y) { return super.max(x + 10, y + 10); }  
15. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Kód sa neskompiluje, pretože volanie metódy max() nie je jednoznačné.
- ☐ b. Kód sa skompiluje a po spustení vypíše 23.
- ☐ c. Kód sa neskompiluje, pretože metóda max() v triede B posiela argumenty vo volaní super.max(y, x) v nesprávnom poradí.
- ☐ d. Kód sa skompiluje a po spustení vypíše 13.
- ☒ e. Kód sa skompiluje a po spustení vypíše 29. ✓
- ☐ f. Kód sa skompiluje a po spustení vypíše 39.

Majme ľubovoľnú triedu nazvanú Book. Ktoré z nasledujúcich deklarácií konštruktorov sú správne?

Majme ľubovoľnú triedu nazvanú Book. Ktoré z nasledujúcich deklarácií konštruktorov sú správne?

Označte jednu alebo viac odpovedí:

- ☐ a. ~~private final Book() {}~~
- ☒ b. ~~abstract Book() {}~~
- ☐ c. ~~Book Book() {}~~
- ☒ d. ~~void Book() {}~~
- ☒ e. ~~Book(Book b) {}~~ ✓
- ☐ f. ~~public static void Book(String[] args) {}~~

Pracuješ na triede `Aquarius`, ktorá bude simulovať akvárium. Už máš metódu, ktorá pridá rybu do akvária a vráti zostávajúcu voľnú kapacitu akvária. Táto metóda má nasledovnú deklaráciu:

```
Public int addFish(Fish f)
```

Teraz chceš vytvoriť metódu na vloženie celej skupiny rýb naraz. Navrhovaná metóda má nasledovnú deklaráciu:

```
Protected boolean addFish(Fish[] f)
```

Otázka 2

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Pracuješ na triede `Aquarius`, ktorá bude simulovať akvárium. Už máš metódu, ktorá pridá rybu do akvária a vráti zostávajúcu voľnú kapacitu akvária. Táto metóda má nasledovnú deklaráciu:

```
public int addFish(Fish f)
```

Teraz chceš vytvoriť metódu na vloženie celej skupiny rýb naraz. Navrhovaná metóda má nasledovnú deklaráciu:

```
protected boolean addFish(Fish[] f)
```

Myšlienka je, že metóda vráti `true`, ak je ešte miesto v akváriu, alebo `false`, ak už tam viac miesta nie je. Ktoré z nasledujúcich tvrdení sú pravdivé?

Označte jednu alebo viac odpovedí:

- ☐ a. Táto technika sa nazýva overriding (prekrývanie).
- ☐ b. Kompilátor odmietne novú funkciu pretože má iný návratový typ.
- ☐ c. Kompilátor odmietne novú funkciu, pretože má iný modifikátor prístupu.
- ☒ d. Táto technika sa nazýva overloading (preťaženie). ✓

Ktoré z uvedených príkazov sa nedajú použiť pre nasledovný kód(class AClass<V>)

```
1. class AClass<V> {  
2.     AClass() { System.out.println(this); }  
3.     <T> AClass(T t) { System.out.println(t); }  
4.     <T> AClass(T t, V v) { System.out.println(t + ", " + v); }  
5. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. AClass<String> ref4 = new <Integer>AClas<String>(2007);
- ☐ b. AClass<String> ref5 = new <String>AClass<String>("one");
- ☒ c. AClass<String> ref8 = new <Integer>AClass<String>("one", 2007);
- ☐ d. AClass<String> ref1 = new AClass<String>();
- ☐ e. AClass<String> ref7 = new <Integer>AClass<String>(2007, "one");
- ☐ f. AClass<String> ref3 = new AClass<String>(2007);
- ☐ g. AClass<String> ref6 = new AClass<String>(2007, "one");
- ☐ h. AClass<String> ref2 = new AClass<String>("one");

Majme vlastnú hierarchiu výnimiek odvodenú od java.lang.Exception nasledovne (BitterException a SourException sú potomkami BadTasteException): (int rateFlavor)

Majme vlastnú hierarchiu výnimiek odvodenú od java.lang.Exception nasledovne (BitterException a SourException sú potomkami BadTasteException):

```
Exception  
+-- BadTasteException  
    +-- BitterException  
    +-- SourException
```

Tieto výnimky majú konštruktor s jediným parametrom typu reťazec (String). Majme metódu deklarovanú nasledovne:

```
int rateFlavor(Ingredient[] list) throws BadTasteException
```

Ktoré z nasledujúcich konštrukcií vyjadrujú správny spôsob vyhodnenia jednej z týchto výnimiek v uvedenej metóde?

Označte jednu alebo viac odpovedí:

- ☐ a. throws new SourException("Ewww!");
- ☒ b. throw new SourException("Ewww!");
- ☐ c. new SourException("Ewww!");
- ☐ d. throw SourException("Ewww!");

Majme vlastnú hierarchiu výnimiek odvodenú od java.lang.Exception nasledovne (BitterException a SourException sú potomkami BadTasteException):

```
Exception  
+-- BadTasteException  
    +-- BitterException  
    +-- SourException
```

Vaša trieda, napr. BaseCook, má metódu deklarovanú nasledovne:

```
int rateFlavor(Ingredient[] list) throws BadTasteException
```

Trieda TexMexCook, zdedená z BaseCook ma metódu pokrývajúcu rateFlavor(). Ktoré z nasledovných deklarácií prekrytej metódy sú správne?

Označte jednu alebo viac odpovedí:

- ☒ a. int rateFlavor(Ingredient[] list) throws BadTasteException
- ☒ b. int rateFlavor(Ingredient[] list) throws BitterException
- ☒ c. int rateFlavor(Ingredient[] list)
- ☐ d. int rateFlavor(Ingredient[] list) throws Exception

Ktoré z príkazov, keď ich vložíme na riadok 6, nespôsobia chybu kompilácie?

Otázka 10

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Ktoré z príkazov, keď ich vložíme na riadok 6, nespôsobia chybu kompilácie?

```
1. public class ThisUsage {  
2.     int planets;  
3.     static int suns;  
4.     public void gaze() {  
5.         int i;  
6.         // SEM VLOŽTE SPRÁVNY PRÍKAZ  
7.     }  
8. }
```

Označte jednu alebo viac odpovedí:

- ☒ a. i = this.planets;
- ☒ b. suns = planets;
- ☐ c. this = new ThisUsage();
- ☐ d. this.i = 4;
- ☒ e. i = suns;

Ktoré konštruktory môžeme vložiť na riadok 11, aby nedošlo ku chybe pri kompilácii?

Ktoré konštruktory môžeme vložiť na riadok 11, aby nedošlo ku chybe pri kompilácii?

```
1. class MySuper {  
2.     int number;  
3.     MySuper(int i) { number = i; }  
4. }  
5. class MySub extends MySuper {  
6.     int count;  
7.     MySub(int count, int num) {  
8.         super(num);  
9.         this.count = count;  
10.    }  
11.    // SEM VLOŽTE SPRÁVNY KONŠTRUKTOR  
12. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. MySub(int count) { this(count, count); } ✓
- ☐ b. MySub(int count) { super(); this.count = count; } ✓
- ☐ c. MySub(int count) { this.count = count; super(count); } ✓
- ☐ d. MySub(int count) { super(count); this(count, 0); } ✓
- ☐ e. MySub() {} ✓
- ☐ f. MySub(int count) { this.count = count; } ✓

Aký je najmenší zoznam tried výnimiek, ktoré musí deklarovať klauzulu throws prekrytá metóda f() v nasledovnom kóde (riadok 10)?

Otázka 1

Ešte nezodpovedané

Max. hodnotenie 3,00

Označiť otázku

Aký je najmenší zoznam tried výnimiek, ktoré musí deklarovať klauzulou throws prekrytá metóda f() v nasledovnom kóde (riadok 10)?

```
1. class A {
2.     // InterruptedException je priamy potomok Exception.
3.     void f() throws ArithmeticException, InterruptedException {
4.         div(5, 5);
5.     }
6.     int div(int i, int j) throws ArithmeticException {
7.         return i / j;
8.     }
9. }
10. public class MyClass extends A {
11.     void f() /* throws DOPLŇTE ZOZNAM TRIED VÝNIMIEK */ {
12.         try {
13.             div(5, 0);
14.         } catch (ArithmeticException e) {
15.             return;
16.         }
17.         throw new RuntimeException("ArithmeticException was expected.");
18.     }
19. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Musí deklarovať, že vyhadzuje ArithmeticException.
- ☐ b. Musí deklarovať, že vyhadzuje RuntimeException.
- ☒ c. Nemusí deklarovať žiadne výnimky. ✓
- ☐ d. Musí deklarovať, že vyhadzuje ArithmeticException a InterruptedException.
- ☐ e. Musí deklarovať, že vyhadzuje InterruptedException.

Ktoré z tvrdení sú pravdivé pre nasledovný kód? (class Fruit)

Ktoré z tvrdení sú pravdivé pre nasledovný kód?

```
1. class Fruit {}
2. class Apple extends Fruit {}
3. public class TestApp {
4.     public static void main(String[] args) {
5.         List<? extends Apple> lst1 = new ArrayList<Fruit>();
6.         List<? extends Fruit> lst2 = new ArrayList<Apple>();
7.         List<? super Apple> lst3 = new ArrayList<Fruit>();
8.         List<? super Fruit> lst4 = new ArrayList<Apple>();
9.         List<?> lst5 = lst1;
10.        List<?> lst6 = lst3;
11.        List lst7 = lst6;
12.        List<?> lst8 = lst7;
13.    }
14. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Riadok 11 sa skompiluje, riadok 12 nie.
- ☐ b. Riadok 9 sa skompiluje, riadok 10 nie.
- ☐ c. Riadok 5 sa skompiluje, riadok 6 nie.
- ☒ d. Riadok 7 sa skompiluje, riadok 8 nie. ✓
- ☐ e. Ani jedna odpoveď nie je správna.

Ktoré z uvedených výrazov je možné použiť na riadku 13, aby uvedený program vypísal hodnotu členej premennej text z triedyMessage? (class Message)

Ktoré z uvedených výrazov je možné použiť na riadku 13, aby uvedený program vypísal hodnotu členej premennej text z triedyMessage?

```
1. class Message {
2.     String text = "Hello, world!";
3. }
4. class MySuperclass {
5.     Message msg = new Message();
6. }
7. public class MyClass extends MySuperclass {
8.     public static void main(String[] args) {
9.         MyClass object = new MyClass();
10.        object.print();
11.    }
12.    public void print() {
13.        System.out.println( /* SEM VLOŽTE SPRÁVNÝ VÝRAZ */ );
14.    }
15. }
```

Označte jednu alebo viac odpovedí:

☒ a. msg.text ✓

☐ b. object.super.msg.text

☐ c. super.text

☐ d. Message.text

☐ e. object.msg.text

☐ f. text

Aký je najmenší zoznam tried výnimiek, ktoré musí deklarovať klauzulou throws prekrytá metóda f() v nasledovnom kóde (riadok 10)?

Aký je najmenší zoznam tried výnimiek, ktoré musí deklarovať klauzulou throws prekrytá metóda f() v nasledovnom kóde (riadok 10)?

```
1. class A {
2.     // InterruptedException je priamy potomok Exception.
3.     void f() throws ArithmeticException, InterruptedException {
4.         div(5, 5);
5.     }
6.     int div(int i, int j) throws ArithmeticException {
7.         return i / j;
8.     }
9. }
10. public class MyClass extends A {
11.     void f() /* throws DOPLŇTE ZOZNAM TRIED VÝNIMIEK */ {
12.         try {
13.             div(5, 0);
14.         } catch (ArithmeticException e) {
15.             return;
16.         }
17.         throw new RuntimeException("ArithmeticException was expected.");
18.     }
19. }
```

Označte jednu alebo viac odpovedí:

☒ a. Musí deklarovať, že vyhadzuje ArithmeticException.

☐ b. Musí deklarovať, že vyhadzuje RuntimeException.

☒ c. Nemusí deklarovať žiadne výnimky. ✓

☐ d. Musí deklarovať, že vyhadzuje ArithmeticException a InterruptedException.

☐ e. Musí deklarovať, že vyhadzuje InterruptedException.

Otázka 1

Nesprávna
odpoveď

Známka 0,00 z
3,00

✓ Označiť
otázku

Aký bude výsledok po skompilovaní a spustení nasledovného programu? (public class MyClass)

Aký bude výsledok po skompilovaní a spustení nasledovného programu?

```
1. public class MyClass {
2.     public static void main(String[] args) {
3.         C c = new C();
4.         System.out.println(c.max(13, 29));
5.     }
6. }
7. class A {
8.     int max(int x, int y) { if (x > y) return x; else return y; }
9. }
10. class B extends A {
11.     int max(int x, int y) { return super.max(y, x) - 10; }
12. }
13. class C extends B {
14.     int max(int x, int y) { return super.max(x + 10, y + 10); }
15. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Kód sa neskompiluje, pretože metóda `max()` v triede B posielala argumenty vo volaní `super.max(y, x)` v nesprávnom poradí.
- ☐ b. Kód sa skompiluje a po spustení vypíše 39.
- ☐ c. Kód sa skompiluje a po spustení vypíše 23.
- ☐ d. Kód sa skompiluje a po spustení vypíše 13.
- ☐ e. Kód sa neskompiluje, pretože volanie metódy `max()` nie je jednoznačné.
- ☐ f. Kód sa skompiluje a po spustení vypíše 29.

Čo spôsobí, že nasledovný kód sa nebude dať skompilovať?

Čo spôsobí, že nasledovný kód sa nebude dať skompilovať?

```
1. class A {
2.     void f() throws ArithmeticException { }
3. }
4. public class MyClass extends A {
5.     public static void main(String[] args) {
6.         A obj = new MyClass();
7.         try {
8.             obj.f();
9.         } catch (ArithmeticException e) {
10.             return;
11.         } catch (Exception e) {
12.             System.out.println(e);
13.             throw new RuntimeException("Something wrong here");
14.         }
15.     }
16.     // InterruptedException je priamy potomok Exception.
17.     void f() throws InterruptedException { }
18. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Kompilátor bude namietat, že blok `catch (ArithmeticException)` zatieňuje blok `catch (Exception)`.
- ☐ b. Prekrytá metóda `f()` v `MyClass` musí deklarovať, že vyhadzuje `ArithmeticException`, keďže je to tak v triede `A`.
- ☐ c. Prekrytá metóda `f()` v `MyClass` nemôže vyhadzovať `InterruptedException`, keďže v `A` takúto výnimku nevyhadzuje.
- ☐ d. Nemôžeme vyhadzovať výnimku v bloku `catch`.
- ☐ e. Všetko je v poriadku, kompilácia prebehne bez chýb.
- ☐ f. Metóda `main()` musí deklarovať, že vyhadzuje `RuntimeException`.

Majme nasledujúce páry deklarácií metód. Ktoré z uvedených tvrdení sú správne? (fly)

Majme nasledujúce páry deklarácií metód. Ktoré z uvedených tvrdení sú správne?

```
void fly(int distance) {}  
int fly(int time, int speed) { return time * speed; }
```

```
void fall(int time) {}  
int fall(int distance) { return distance; }
```

```
void glide(int time) {}  
void Glide(int time) {}
```

Označte jednu alebo viac odpovedí:

- ☒ a. Prvý pár metód sa skompiluje, pretože ide o preťaženie (overloading) metód fly(). ✓
- ☐ b. Druhý pár metód sa neskompiluje. ✓
- ☒ c. Druhý pár metód sa skompiluje, pretože ide o preťaženie (overloading) metód fall(). ✓
- ☐ d. Tretí pár metód sa skompiluje, pretože ide o preťaženie (overloading) metód glide(). ✓
- ☐ e. Tretí pár metód sa neskompiluje. ✓



Ktoré z deklarácií členskej premennej accounts je možné použiť na riadku 2 uvedeného kódu? (Testing)

Otázka 3

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Ktoré z deklarácií členskej premennej accounts je možné použiť na riadku 2 uvedeného kódu?

```
1. public class Testing {  
2.     // VLOŽTE SPRÁVNU DEKLARÁCIU ČLENSKEJ PREMENNEJ  
3.     public long getNum(String name) {  
4.         Long number = accounts.get(name);  
5.         return number == null ? 0 : number;  
6.     }  
7.     public void setNum(String name, long number) {  
8.         accounts.put(name, number);  
9.     }  
10. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. private Map<String, long> accounts = new HashMap<String, long>();
- ☒ b. private Map<String, Long> accounts = new HashMap<String, Long>(); ✓
- ☐ c. private Map<String<Long>> accounts = new HashMap<String<Long>>();
- ☐ d. private Map accounts = new HashMap();
- ☐ e. private Map<String, Long> accounts = new Map<String, Long>();

Aký bude výsledok po skompilovaní a ... (class Vehicle, volvo toyota)

```
6. static public String getModelName() { return "Toyota"; }
7. public long getRegNo() { return 54321; }
8. }
9. public class TakeARide {
10. public static void main(String args[]) {
11.     Car c = new Car();
12.     Vehicle v = c;
13.     System.out.println("|" + v.getModelName() + "|" + c.getModelName() +
        "|" + v.getRegNo() + "|" + c.getRegNo() + "|");
14. }
15. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Kód sa skompiluje a po spustení vypíše "|Volvo|Volvo|12345|54321|"
- ☐ b. Kód sa skompiluje a po spustení vypíše "|Toyota|Toyota|12345|12345|".
- ☐ c. Kód sa skompiluje a po spustení vypíše "|Volvo|Toyota|12345|54321|".
- ☐ d. Kód sa skompiluje a po spustení vypíše "|Toyota|Volvo|12345|12345|".
- ☐ e. Kód sa skompiluje a po spustení vypíše "|Toyota|Volvo|12345|54321|".
- ☒ f. Kód sa skompiluje a po spustení vypíše "|Volvo|Toyota|54321|54321|".
- ☐ g. Program sa nebude dať skompilovať.



Majme definovanú nasledovnú triedu: (widget extends thingee)

Majme definovanú nasledovnú triedu:

```
1. class Widget extends Thingee {
2.     static final int maxWidgetSize = 40;
3.     static String title;
4.     public Widget(int mx, String t) {
5.         maxWidgetSize = mx;
6.         title = t;
7.     }
8.     // ďalší kód
9. }
```

Čo sa stane, keď sa pokúsime skompilovať tento kód a spustiť program, ktorý vytvára objekt widget nasledovným spôsobom?

```
10. Widget myWidget = new Widget(50, "Bigger");
```

Označte jednu alebo viac odpovedí:

- ☐ a. Program sa skompiluje a spustí bez chyby.
- ☐ b. Nastane chyba pri kompilácii na riadku 5.
- ☐ c. Nastane chyba pri kompilácii na riadku 6.
- ☐ d. Program sa skompiluje, ale počas behu programu nastane chyba na riadku 5.






Majme nasledovné deklarácie: (interface A)

Majme nasledovné deklarácie:

```
interface A {}  
class B {}  
class C extends B implements A {}  
class D implements A {}  
B b = new B();  
C c = new C();  
D d = new D();
```

Ktoré z nasledovných priradení sú správne?

Označte jednu alebo viac odpovedí:

- ☐ a. c = d; 
- ☒ b. d = (D) c; 
- ☐ c. d = c; 
- ☐ d. c = b; 
- ☒ e. A a = d; 

Ktoré z príkazov, keď ich vložíme na riadok 4, nespôsobia chybu kompilácie?

Otázka 10

Správna
odpoveď



Známka 3,00 z
3,00

🚩 Označiť
otázku

Ktoré z príkazov, keď ich vložíme na riadok 4, nespôsobia chybu kompilácie?

```
1. public class Testing {  
2.     public static void main(String[] args) {  
3.         List<?> lst = new ArrayList<String>();  
4.         // SEM VLOŽTE SPRÁVNY PRÍKAZ  
5.     }  
6. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. String v1 = lst.get(0);
- ☒ b. lst.add(null); 
- ☐ c. lst.add("OK");
- ☐ d. lst.add(2007);
- ☒ e. Object v2 = lst.get(0); 

Ktoré z deklarácií môžu byť použité na riadku 3, aby sa nasledovný program dal skompilovať a spustiť bez chyby? (Testing)

Ktoré z deklarácií môžu byť použité na riadku 3, aby sa nasledovný program dal skompilovať a spustiť bez chyby?

```
1. public class Testing {  
2.     public static void main(String[] args) {  
3.         // SEM VLOŽTE SPRÁVNU DEKLARÁCIU  
4.         for (int i = 0; i <= 5; i++) {  
5.             List<Integer> row = new ArrayList<Integer>();  
6.             for (int j = 0; j <= i; j++)  
7.                 row.add(i * j);  
8.             ds.add(row);  
9.         }  
10.        for (List<Integer> row : ds)  
11.            System.out.println(row);  
12.    }  
13. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. `ArrayList<ArrayList<Integer>> ds = new ArrayList<ArrayList<Integer>>();`
- ☒ b. `List<List<Integer>> ds = new ArrayList<List<Integer>>();`
- ☐ c. `List<List, Integer> ds = new ArrayList<List, Integer>();`
- ☐ d. `List<List, Integer> ds = new ArrayList<ArrayList, Integer>();`
- ☐ e. `List<List<Integer>> ds = new ArrayList<ArrayList<Integer>>();`
- ☐ f. `List<ArrayList<Integer>> ds = new ArrayList<ArrayList<Integer>>();`
- ☐ g. `List<List, Integer> ds = new List<List, Integer>();`
- ☐ h. `List<List<Integer>> ds = new List<List<Integer>>();`

Majme nasledovný kód: (exceptions)

Majme nasledovný kód:

```
1. public class Exceptions {  
2.     public static void main(String[] args) {  
3.         try {  
4.             if (args.length == 0) return;  
5.             System.out.println(args[0]);  
6.         } finally {  
7.             System.out.println("The end");  
8.         }  
9.     }  
10. }
```

Ktoré z tvrdení sú pravdivé?

Označte jednu alebo viac odpovedí:

- ☐ a. Ak spustíme program s jedným argumentom, program vypíše tento argument.
- ☐ b. Program vyhodí výnimku `ArrayIndexOutOfBoundsException`.
- ☐ c. Ak spustíme program bez argumentov, program nebude mať žiaden výstup.
- ☐ d. Ak spustíme program bez argumentov, program vypíše "The end".
- ☐ e. Ak spustíme program s jedným argumentom, program vypíše tento argument a za ním vypíše "The end".

Aký bude výsledok po skompilovaní a spustení nasledovného programu?
(exceptions)

Majme nasledovný kód:

```
1. public class Exceptions {
2.     public static void main(String[] args) {
3.         try {
4.             if (args.length == 0) return;
5.             System.out.println(args[0]);
6.         } finally {
7.             System.out.println("The end");
8.         }
9.     }
10. }
```

Ktoré z tvrdení sú pravdivé?

Označte jednu alebo viac odpovedí:

- ☐ a. Ak spustíme program s jedným argumentom, program vypíše tento argument.
- ☐ b. Program vyhodí výnimku `ArrayIndexOutOfBoundsException`.
- ☐ c. Ak spustíme program bez argumentov, program nebude mať žiaden výstup.
- ☒ d. Ak spustíme program bez argumentov, program vypíše "The end". ✓
- ☒ e. Ak spustíme program s jedným argumentom, program vypíše tento argument a za ním vypíše "The end". ✓

Aký bude výsledok po skompilovaní a sustení nasledovného programu?
(MyClass)

Aký bude výsledok po skompilovaní a spustení nasledovného programu?

```
1. class MyClass {
2.     static MyClass ref;
3.     String[] arguments;
4.     public static void main(String[] args) {
5.         ref = new MyClass();
6.         ref.func(args);
7.     }
8.     public void func(String[] args) {
9.         ref.arguments = args;
10.    }
11. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Program sa neskompiluje, pretože parameter `args` prijatý statickou metódou `main()` nemôže byť odoslaný do nestatickej metódy `func()`.
- ☐ b. Program sa neskompiluje, pretože nestatická metóda `func()` nemôže pristupovať k statickej premennej `ref`.
- ☐ c. Program sa neskompiluje, pretože statická metóda `main()` nemôže zavolať nestatickú metódu `func()`.
- ☐ d. Program sa skompiluje a vykoná bez chýb.
- ☐ e. Program sa skompiluje, ale po spustení vyhodí výnimku.

Ktoré z nasledovných deklarácií sa skompilujú bez varovných správ?

Otázka 9

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Ktoré z nasledovných deklarácií sa skompilujú bez varovných správ?

Označte jednu alebo viac odpovedí:

- ☒ a. `Map<? super Integer, ? super Integer> map5 = new HashMap<Number, Number>();` ✓
- ☒ b. `Map<Integer, HashMap<Integer, String>> map2 = new HashMap<Integer, HashMap<Integer, String>>();` ✓
- ☒ c. `Map<Integer, Integer> map3 = new HashMap<Integer, Integer>();` ✓
- ☐ d. `Map<?, ?> map7 = new HashMap<?, ?>();`
- ☐ e. `Map<Integer, Map<Integer, String>> map1 = new HashMap<Integer, HashMap<Integer, String>>();`
- ☐ f. `Map<? super Integer, ? super Integer> map4 = new HashMap<? super Integer, ? super Integer>();`
- ☒ g. `Map<? extends Number, ? extends Number> map6 = new HashMap<Number, Number>();` ✓

Toto je hierarchia tried výnimiek súvisiacich s chybami indexovania poľa a reťazca:

Toto je hierarchia tried výnimiek súvisiacich s chybami indexovania poľa a reťazca:

```
Exception
+-- RuntimeException
    +-- IndexOutOfBoundsException
        +-- ArrayIndexOutOfBoundsException
        +-- StringIndexOutOfBoundsException
```

Majme metódu `f()`, v ktorej môžu byť hodené obidve výnimky indexovania poľa aj indexovania reťazca. V samotnej metóde `f()` nie je blok `try/catch`, ktorý by tieto výnimky odchytil. Označte správne tvrdenia pre túto situáciu:

Označte jednu alebo viac odpovedí:

- ☐ a. Ak deklarácia `f()` obsahuje `"throws IndexOutOfBoundsException"`, akákoľvek volajúca metóda musí použiť `try/catch` blok.
- ☐ b. Deklarácia `f()` musí obsahovať `"throws ArrayIndexOutOfBoundsException, StringIndexOutOfBoundsException"`.
- ☐ c. Deklarácia metódy `f()` nemusí vôbec riešiť výnimky.
- ☐ d. Ak metóda volajúca `f()` odchyta výnimku `IndexOutOfBoundsException`, obidve výnimky indexovania poľa aj reťazca sú odchytené.

Čo vypíše nasledovný program po jeho spustení? (myclass test)

Otázka 3

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Čo vypíše nasledovný program po jeho spustení?

```
1. public class MyClass {
2.     public static void main(String[] args) {
3.         B b = new B("Test");
4.     }
5. }
6. class A {
7.     A() { this("1", "2"); }
8.     A(String s, String t) { this(s + t); }
9.     A(String s) { System.out.println(s); }
10. }
11. class B extends A {
12.     B(String s) { System.out.println(s); }
13.     B(String s, String t) { this(t + s + "3"); }
14.     B() { super("4"); }
15. }
```

Označte jednu alebo viac odpovedí:

- ☒ a. Vypíše sa reťazec "12" nasledovaný reťazcom "Test".
- ☐ b. Vypíše sa reťazec "Test".
- ☐ c. Vypíše sa reťazec "123" nasledovaný reťazcom "Test".
- ☐ d. Vypíše sa reťazec "4" nasledovaný reťazcom "Test".
- ☐ e. Vypíše sa reťazec "Test" nasledovaný reťazcom "Test".



Čo vypíše nasledovný program po jeho spustení? (public class Myclass)

Čo vypíše nasledovný program po jeho spustení?

```
1. public class MyClass {
2.     public static void main(String[] args) throws InterruptedException {
3.         try {
4.             f();
5.             System.out.println("1");
6.         } finally {
7.             System.out.println("2");
8.         }
9.         System.out.println("3");
10.    }
11.    static void f() throws InterruptedException {
12.        throw new InterruptedException("Time to go home.");
13.    }
14. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Program vypíše "1" a "3".
- ☐ b. Program vypíše "3" a "2".
- ☐ c. Program vypíše "1", "2" a "3".
- ☐ d. Program vypíše "2" a vyhodí výnimku InterruptedException.
- ☐ e. Program vypíše "1" a "2".
- ☐ f. Program vypíše "2" a "3".

(class Fruit)

Aký bude výsledok po skompilovaní a spustení nasledovného kódu:

```
1. class Fruit {}
2. class Apple extends Fruit {}
3. class Orange extends Fruit {}
4. public class Testing {
5.     public static void main(String[] args) {
6.         ArrayList<Apple> aList = new ArrayList<Apple>();
7.         aList.add(new Apple());
8.         ArrayList bList = aList;
9.         ArrayList<Orange> oList = bList;
10.        oList.add(new Orange());
11.        System.out.println(aList);
12.    }
13. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Kód sa skompiluje a po spustení hodí výnimku `ClassCastException` na riadku 9.
- ☐ b. Kód sa neskompiluje kvôli chybe na riadku 8.
- ☐ c. Kód sa skompiluje bez chýb a varovaní a po spustení vypíše text "[Apple@hhhhh, Orange@HHHHHHH]", pričom hhhhhh a HHHHHHH reprezentujú nejaký hash kód.
- ☒ d. Kód sa skompiluje, ale ohlásí neošetrenú konverziu na riadku 9. ✓
- ☒ e. Kód sa skompiluje, ale ohlásí neošetrenú konverziu na riadku 8. ✗

Aký bude výsledok po pokuse skompilovať a spustiť nasledovný kód? (public class polymorphism)

Aký bude výsledok po pokuse skompilovať a spustiť nasledovný kód?

```
1. public class Polymorphism {
2.     public static void main(String[] args) {
3.         A ref1 = new C();
4.         B ref2 = (B) ref1;
5.         System.out.println(ref2.g());
6.     }
7. }
8. class A {
9.     private int f() { return 0; }
10.    public int g() { return 3; }
11. }
12. class B extends A {
13.     private int f() { return 1; }
14.     public int g() { return f(); }
15. }
16. class C extends B { public int f() { return 2; } }
```

Označte jednu alebo viac odpovedí:

- ☒ a. Program sa skompiluje a po spustení vypíše 1. ✓
- ☐ b. Program sa skompiluje a po spustení vypíše 3.
- ☐ c. Program sa skompiluje a po spustení vypíše 0.
- ☐ d. Program sa skompiluje a po spustení vypíše 2.
- ☐ e. Program sa nepodarí skompilovať.

Čo vypíše nasledovný program po jeho spustení? (public class MyClass)

Otázka **7**

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Čo vypíše nasledovný program po jeho spustení?

```
1. public class MyClass {
2.     public static void main(String[] args) {
3.         int k = 0;
4.         try {
5.             int i = 5 / k;
6.         } catch (ArithmeticException e) {
7.             System.out.println("1");
8.         } catch (RuntimeException e) {
9.             System.out.println("2");
10.        return;
11.        } catch (Exception e) {
12.            System.out.println("3");
13.        } finally {
14.            System.out.println("4");
15.        }
16.        System.out.println("5");
17.    }
18. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Program vypíše "3" a "5".
- ☐ b. Program vypíše "1", "2", "4" a "5".
- ☐ c. Program vypíše "1", "2" a "4".
- ☐ d. Program vypíše "5".
- ☒ e. Program vypíše "1", "4" a "5".
- ☐ f. Program vypíše "1" a "4".



Ktorú z deklarácií metódy justDoIt() môžeme použiť na riadku 9, aby sme nedostali varovné správy pri kompilácii? (public class testing)

Otázka **2**

Správna
odpoveď

Známka 3,00 z
3,00

Označiť
otázku

Ktorú z deklarácií metódy justDoIt() môžeme použiť na riadku 9, aby sme nedostali varovné správy pri kompilácii?

```
1. public class Testing {
2.     public static void main(String[] args) {
3.         List raw = new ArrayList();
4.         raw.add("2007");
5.         raw.add(2008);
6.         raw.add("2009");
7.         justDoIt(raw);
8.     }
9.     // SEM VLOŽTE SPRÁVNU DEKLARÁCIU FUNKCIE
10. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. static void justDoIt(List<Integer> lst) { }
- ☐ b. Žiadna z uvedených deklarácií.
- ☒ c. static void justDoIt(List<?> lst) { }
- ☐ d. static void justDoIt(List<T> lst) { }



Nech Thing je trieda. Koľko objektov a koľko referencií je vytvorených nasledovným kódom?

Nech Thing je trieda. Koľko objektov a koľko referencií je vytvorených nasledovným kódom?

```
1. Thing item, stuff;  
2. item = new Thing();  
3. Thing entity = new Thing();
```

Označte jednu alebo viac odpovedí:

- ☒ a. Sú vytvorené tri referencie. ✓
- ☐ b. Je vytvorená jedna referencia.
- ☐ c. Sú vytvorené dve referencie.
- ☒ d. Sú vytvorené dva objekty. ✓
- ☐ e. Je vytvorený jeden objekt.
- ☐ f. Sú vytvorené tri objekty.

Majme triedy A, B a C, pričom B je odvodená od A a C je odvodená od B. Každá z týchto tried implementuje metódu doIt(). Ako môžem zavolať metódu doIt() z triedy A v objekte vytvorenom z triedy C?

Majme triedy A, B a C, pričom B je odvodená od A a C je odvodená od B. Každá z týchto tried implementuje metódu doIt(). Ako môžem zavolať metódu doIt() z triedy A v objekte vytvorenom z triedy C?

Označte jednu alebo viac odpovedí:

- ☐ a. ((A) this).doIt();
- ☐ b. A.this.doIt();
- ☐ c. this.super.doIt();
- ☒ d. Nie je to možné. ✓
- ☐ e. super.doIt();
- ☐ f. super.super.doIt();
- ☐ g. doIt();

Otázka 4

Správna
odpoveď

Známka 3,00 z
3,00

🚩 Označiť
otázku

Čo spôsobí, že nasledovný kód sa nebude dať skompilovať? (class A)

Čo spôsobí, že nasledovný kód sa nebude dať skompilovať?

```
1. class A {
2.     void f() throws ArithmeticException { }
3. }
4. public class MyClass extends A {
5.     public static void main(String[] args) {
6.         A obj = new MyClass();
7.         try {
8.             obj.f();
9.         } catch (ArithmeticException e) {
10.            return;
11.        } catch (Exception e) {
12.            System.out.println(e);
13.            throw new RuntimeException("Something wrong here");
14.        }
15.    }
16.    // InterruptedException je priamy potomok Exception.
17.    void f() throws InterruptedException { }
18. }
```

Označte jednu alebo viac odpovedí:

- ☐ a. Kompilátor bude namietať, že blok catch (ArithmeticException) zatieňuje blok catch (Exception).
- ☐ b. Prekrytá metóda f() v MyClass musí deklarovať, že vyhadzuje ArithmeticException, keďže je to tak v triede A.
- ☐ c. Prekrytá metóda f() v MyClass nemôže vyhadzovať InterruptedException, keďže v A takúto výnimku nevyhadzuje.
- ☐ d. Nemôžeme vyhadzovať výnimku v bloku catch.
- ☐ e. Všetko je v poriadku, kompilácia prebehne bez chýb.
- ☒ f. Metóda main() musí deklarovať, že vyhadzuje RuntimeException. ✓

(public boolean strCvt)

Otázka 6

Nesprávny

Známka 0,00 z 3,00

Flag question

Nasledovná metóda je navrhnutá na konverziu vstupného reťazca na reálne číslo, pričom deteguje nesprávny formát (factor je premenná definovaná inde)

```
1. public boolean strCvt(String s) {
2.     try {
3.         factor = Float.valueOf(s).floatValue();
4.         return true;
5.     } catch (NumberFormatException e) {
6.         System.out.println("Bad number " + s);
7.         factor = Float.NaN;
8.     } finally {
9.         System.out.println("Finally");
10.    }
11.    return false;
12. }
```

Ktoré výsledky metódy pre zadané vstupy sú správne?

Vyberte jednu alebo viac:

- ☐ a. Vstup s = null. Výsledok factor = NaN, text "Finally" je vypísaný, vrátená je hodnota false.
- ☐ b. Vstup s = null. Výsledok factor je nezmenený, text "Finally" je vypísaný, výnimka NullPointerException je hodaná. ✓
- ☒ c. Vstup s = "0.234". Výsledok factor = 0.234, text "Finally" je vypísaný, vrátená je hodnota false. ✗
- ☒ d. Vstup s = "0.234". Výsledok factor = 0.234, text "Finally" je vypísaný, vrátená je hodnota true. ✓

Správna odpoveď je Vstup s = "0.234". Výsledok factor = 0.234, text "Finally" je vypísaný, vrátená je hodnota true. Vstup s = null. Výsledok factor je nezmenený, text "Finally" je vypísaný, výnimka NullPointerException je hodaná.

Majme nasledovnú definíciu triedy: (Demo)

Majme nasledovnú definíciu triedy:

```
1. public class DerivedDemo extends Demo {  
2.     int M, N, L;  
3.     public DerivedDemo(int x, int y) {  
4.         M = x; N = y;  
5.     }  
6.     public DerivedDemo(int x) {  
7.         super(x);  
8.     }  
9. }
```

Ktoré z nasledovných signatúr konštruktorov musia existovať v triede Demo, aby sa trieda DerivedDemo skompilovala správne?

Vyberte jednu alebo viac:

- ☐ a. public Demo(int c) ✓
- ☒ b. public Demo(int a, int b) ✗
- ☐ c. Nie je potrebné mať konštruktor v triede Demo.
- ☒ d. public Demo() ✓

Správna odpoveď je: public Demo(int c), public Demo().

Majme nasledovnú definíciu triedy Widget:

Majme nasledovnú definíciu triedy Widget:

```
1. class Widget extends Thingee {  
2.     static private int widgetCount = 0;  
3.     public String wName;  
4.     int wNumber;  
5.     static int addWidget() {  
6.         widgetCount++;  
7.         wName = "I am Widget # " + widgetCount;  
8.         return widgetCount;  
9.     }  
10.    public Widget() {  
11.        wNumber = addWidget();  
12.    }  
13. }
```

Čo sa stane po skompilovaní tejto triedy a použití viacerých inštancií Widget objektov v programe?

Vyberte jednu alebo viac:

- ☒ a. Trieda sa skompiluje, každý objekt Widget bude mať jedinečnú hodnotu wNumber a wName v takom poradí v akom sa objekty vytvárajú. ✗
- ☒ b. Program sa skompiluje, ale počas behu programu vznikne chyba súvisiaca s prístupom k premennej wName v metóde addWidget(). ✗
- ☐ c. Nastane chyba pri kompilácii na riadku 7. ✓
- ☐ d. Nastane chyba pri kompilácii na riadku 11.