

# Java backend - 3. részvizsga - 02.15.

## Respondent Complete Analysis

Assessment Unique ID: A003C3MW7KXD

Email Address: djtesla@gmail.com  
Full Name: Zámbo, Ernő  
Oktatóközpont: TRAINING360  
Date Started: 02/15/2021 4:21:59 PM  
Date Completed: 02/15/2021 4:42:41 PM  
Attempt: 1


Question Topic		
Num	Question	
	Respondent's Answer	Correct Answer
RESZV		

1 Mit ír ki az alábbi kódrészlet?  
public class TestException {

```
    public void throwException() {  
        System.out.print("throwex ");  
        throw new RuntimeException();  
    }  
  
    public static void main(String[] args) {  
        try {  
            System.out.print("try ");  
            new TestException().throwException();  
        } catch (Exception re) {  
            System.out.print("catch ");  
        } finally {  
            System.out.print("finally ");  
        }  
        System.out.println("done ");  
    }  
}
```

- ☐ ( ) try throwex catch
- ☒ (X) try throwex catch finally done
- ☐ ( ) try throwex RuntimeException catch finally done
- ☐ ( ) try throwex RuntimeException

- ☐ ( ) try throwex catch
- ☐ (X) try throwex catch finally done
- ☐ ( ) try throwex RuntimeException catch finally done
- ☐ ( ) try throwex RuntimeException

 Earned Points: 1 of 1

2

Hogyan dobható Exception kézzel?

- ☒ (X) throw new MyException();
- ☐ ( ) throw MyException;
- ☐ ( ) throws new MyException();
- ☐ ( ) throws myException;

- ☒ (X) throw new MyException();
- ☐ ( ) throw MyException;
- ☐ ( ) throws new MyException();
- ☐ ( ) throws myException;



Earned Points: 1 of 1

3

Az alábbiak közül melyik állítás igaz?

- ☒ (X) Több elemet egyszerre törölni a List names-ből az names.removeAll(namesToRemove) metódussal lehet, ahol az namesToRemove a törlendő elemek listája.
- ☐ ( ) Az ArrayList belsejében egy tömb van, ezért a tömbökhöz hasonlóan előre meg kell adni a méretét létrehozáskor.
- ☐ ( ) Az List names legelső elemét az names.get(1) metódussal lehet elkérni.
- ☐ ( ) Bármilyen típusú érték kerülhet egy ArrayList-be, például: ArrayList integers = new ArrayList

- ☒ (X) Több elemet egyszerre törölni a List names-ből az names.removeAll(namesToRemove) metódussal lehet, ahol az namesToRemove a törlendő elemek listája.
- ☐ ( ) Az ArrayList belsejében egy tömb van, ezért a tömbökhöz hasonlóan előre meg kell adni a méretét létrehozáskor.
- ☐ ( ) Az List names legelső elemét az names.get(1) metódussal lehet elkérni.
- ☐ ( ) Bármilyen típusú érték kerülhet egy ArrayList-be, például: ArrayList integers = new ArrayList



Earned Points: 1 of 1

4

Létre akarok hozni egy Object típusú példányt, és elmenteni őt egy Object típusú változóba, hogyan tudom ezt megtenni?

- ☒ (X) Object object = new Object();
- ☐ ( ) Object object = Object();
- ☐ ( ) Object object = new Object;
- ☐ ( ) Object = new Object();

- ☒ (X) Object object = new Object();
- ☐ ( ) Object object = Object();
- ☐ ( ) Object object = new Object;
- ☐ ( ) Object = new Object();



Earned Points: 1 of 1

5

Melyik állítás hamis a következők közül?

- ☐ ( ) A következő utasítás egy üres listát hoz létre: List names = new ArrayList
- ☒ (X) A lista elemeit a következőképp adhatjuk meg: List names = {"Joe", "Jane"};
- ☐ ( ) A lista mérete a size() metódussal kérdezhető le
- ☐ ( ) A listába új elemeket lehet hozzáadni, és elemeket lehet törölni

- ☐ ( ) A következő utasítás egy üres listát hoz létre: List names = new ArrayList
- ☒ (X) A lista elemeit a következőképp adhatjuk meg: List names = {"Joe", "Jane"};
- ☐ ( ) A lista mérete a size() metódussal kérdezhető le
- ☐ ( ) A listába új elemeket lehet hozzáadni, és elemeket lehet törölni



Earned Points: 1 of 1

6

Melyik igaz az absztrakt függvényekre?

- ☒ (X) Nincs függvénytörzsük (method body)
- ☐ ( ) Mindenképpen void a visszatérési típusuk
- ☐ ( ) Nem kaphatnak bemeneti paramétert
- ☐ ( ) Nem szükséges implementálni őket a nem-absztrakt alosztályokban

- ☒ (X) Nincs függvénytörzsük (method body)
- ☐ ( ) Mindenképpen void a visszatérési típusuk
- ☐ ( ) Nem kaphatnak bemeneti paramétert
- ☐ ( ) Nem szükséges implementálni őket a nem-absztrakt alosztályokban



Earned Points: 1 of 1

7

Milyen operációs rendszeren lehet Java alkalmazást futtatni?

- ☒ (X) Mindegyiken, a Java alkalmazásokat az egyes platformokra elkészített Java Virtuális Gépek (JVM) futtatják, emiatt platformfüggetlen
- ☐ ( ) A Java kifejezetten a Microsoft Windows operációs rendszerhez készült programozási nyelv
- ☐ ( ) A Java egy open-source programozási nyelv, ezért Linuxra készíthetünk vele programokat
- ☐ ( ) A Java az Apple terméke, MacOS és IOS alkalmazásokat készíthetünk vele

- ☒ (X) Mindegyiken, a Java alkalmazásokat az egyes platformokra elkészített Java Virtuális Gépek (JVM) futtatják, emiatt platformfüggetlen
- ☐ ( ) A Java kifejezetten a Microsoft Windows operációs rendszerhez készült programozási nyelv
- ☐ ( ) A Java egy open-source programozási nyelv, ezért Linuxra készíthetünk vele programokat
- ☐ ( ) A Java az Apple terméke, MacOS és IOS alkalmazásokat készíthetünk vele



Earned Points: 1 of 1

8

Az alábbiak közül melyik igaz az absztrakt osztályokra?

- ☐ ( ) Származhatnak egy másik absztrakt osztályból.
- ☒ (X) Nem lehetnek példányváltozók.
- ☐ ( ) Absztrakt osztályból való leszármazáshoz az implements kulcsszót használjuk.
- ☐ ( ) Csak absztrakt metódusokat tartalmazhatnak.

- ☒ (X) Származhatnak egy másik absztrakt osztályból.
- ☐ ( ) Nem lehetnek példányváltozók.
- ☐ ( ) Absztrakt osztályból való leszármazáshoz az implements kulcsszót használjuk.
- ☐ ( ) Csak absztrakt metódusokat tartalmazhatnak.



Earned Points: 0 of 1

9

Melyik változónév helyes a Clean Code alapelvek szerint?

- ☒ (X) firstName
- ☐ ( ) cc
- ☐ ( ) last\_name
- ☐ ( ) FirstName

- ☒ (X) firstName
- ☐ ( ) cc
- ☐ ( ) last\_name
- ☐ ( ) FirstName



Earned Points: 1 of 1

10

Adott a következő kódrészlet:

```
public class Trainer {

    private String name;

    public Trainer(String name) {
        this.name = name;
    }
}
```

Mely állítás hamis az alábbiak közül?

- ☒ (X) Az osztály példányosítható (instantiate) a következő módon: `Trainer t = new Trainer();`
- ☐ ( ) Az osztály példányosítható (instantiate) a következő módon: `Trainer t = new Trainer("Anonymous");`
- ☐ ( ) Az osztály példányosítható (instantiate) a következő módon: `Trainer t = new Trainer(null);`
- ☐ ( ) Nem kerül legenerálásra paraméter nélküli implicit (default) konstruktor, hiszen van explicit konstruktor

- ☒ (X) Az osztály példányosítható (instantiate) a következő módon: `Trainer t = new Trainer();`
- ☐ ( ) Az osztály példányosítható (instantiate) a következő módon: `Trainer t = new Trainer("Anonymous");`
- ☐ ( ) Az osztály példányosítható (instantiate) a következő módon: `Trainer t = new Trainer(null);`
- ☐ ( ) Nem kerül legenerálásra paraméter nélküli implicit (default) konstruktor, hiszen van explicit konstruktor



Earned Points: 1 of 1

11

Adott az alábbi kódrészlet.

```
public class Concatenator {

    public static String convert(int a, int b) {
        return Integer.toString(a) + Integer.toString(b);
    }

    public static void main(String[] args) {
        // ???
    }
}
```

- ☐ ( ) Csak a `Concatenator.convert(5, 6);` utasítással
- ☐ ( ) Csak a `convert(5, 6);` utasítással
- ☐ ( ) Csak a `new Concatenator().convert(5, 6);` utasítással
- ☒ (X) Mindhárom utasítással, de nem mindegyik javasolt

- ☐ ( ) Csak a `Concatenator.convert(5, 6);` utasítással
- ☐ ( ) Csak a `convert(5, 6);` utasítással
- ☐ ( ) Csak a `new Concatenator().convert(5, 6);` utasítással
- ☒ (X) Mindhárom utasítással, de nem mindegyik javasolt



Earned Points: 1 of 1

12

Mit ír ki az alábbi kódrészlet?

```
public class State {

    private static int instance = 0;

    public State() {
        instance++;
    }

    public static void main(String[] args) {
        new State();
        new State();
        System.out.println(instance);
    }
}
```

- ☐ ( ) Nem fordul le
- ☐ ( ) 0
- ☐ ( ) 1
- ☒ (X) 2

- ☐ ( ) Nem fordul le
- ☐ ( ) 0
- ☐ ( ) 1
- ☒ (X) 2



Earned Points: 1 of 1

13

Mit ír ki az alábbi Java kód?

```
public class Test {

    public void doSomething() {
        throw new RuntimeException();
    }

    public static void main(String[] args) {
        try {
            new Test().doSomething();
        }
        catch (Throwable t) {
            System.out.println("Yupp");
        }
    }
}
```

- ☐ ( ) Nem fordul le, mert a doSomething() metódus nem definiálja a fejben (method header) a RuntimeException kivételt (exception)
- ☒ (X) Yupp
- ☐ ( ) Az alkalmazás leáll, mert nem kezelt kivétel (unchecked exception) keletkezik
- ☐ ( ) Nem ír ki semmit, az alkalmazás hiba nélkül lefut

- ☐ ( ) Nem fordul le, mert a doSomething() metódus nem definiálja a fejben (method header) a RuntimeException kivételt (exception)
- ☒ (X) Yupp
- ☐ ( ) Az alkalmazás leáll, mert nem kezelt kivétel (unchecked exception) keletkezik
- ☐ ( ) Nem ír ki semmit, az alkalmazás hiba nélkül lefut




Earned Points: 1 of 1

14 Melyik primitív típus?

- ☒ (X) char
- ☐ ( ) Integer
- ☐ ( ) String
- ☐ ( ) String[]


- ☒ (X) char
- ☐ ( ) Integer
- ☐ ( ) String
- ☐ ( ) String[]

 Earned Points: 1 of 1

15 Melyik nem primitív típus?

- ☒ (X) String
- ☐ ( ) int
- ☐ ( ) char
- ☐ ( ) double


- ☒ (X) String
- ☐ ( ) int
- ☐ ( ) char
- ☐ ( ) double

 Earned Points: 1 of 1

16 Egy korábbi sorban már létrehozott Object object változóm értékét szeretném felülírni egy új Object példánnyal, hogyan tudom ezt megtenni?

- ☒ (X) object = new Object();
- ☐ ( ) Object object = new Object();
- ☐ ( ) object = Object();
- ☐ ( ) object = new object();

- ☒ (X) object = new Object();
- ☐ ( ) Object object = new Object();
- ☐ ( ) object = Object();
- ☐ ( ) object = new object();


 Earned Points: 1 of 1

17 Mit ír ki az alábbi kódrészlet?

```
int i = 5;
if (i % 2 != 0) {
    String value = "Odd";
}
System.out.println(value);
```

- ☒ (X) Nem fordul le.
- ☐ ( ) Odd
- ☐ ( ) null
- ☐ ( ) Futás közben hibával leáll.

- ☒ (X) Nem fordul le.
- ☐ ( ) Odd
- ☐ ( ) null
- ☐ ( ) Futás közben hibával leáll.

 Earned Points: 1 of 1

18

Melyik hamis a kódolási konvenciókkal (coding conventions) kapcsolatban?

- |   |   |
|---|---|
| <input type="radio"/> Változónevek első karaktere, ha betű, kisbetű legyen<br><input type="radio"/> Metódusnevek első karaktere, ha betű, kisbetű legyen<br><input checked="" type="radio"/> Osztálynevek első karaktere, ha betű, kisbetű legyen<br><input type="radio"/> A konstans értékek nevei csupa nagybetűvel és a szavak között aláhúzás karakterrel szerepeljenek | <input type="radio"/> Változónevek első karaktere, ha betű, kisbetű legyen<br><input type="radio"/> Metódusnevek első karaktere, ha betű, kisbetű legyen<br><input checked="" type="radio"/> Osztálynevek első karaktere, ha betű, kisbetű legyen<br><input type="radio"/> A konstans értékek nevei csupa nagybetűvel és a szavak között aláhúzás karakterrel szerepeljenek |
|---|---|



Earned Points: 1 of 1

19

Adott két változó:

`int a = 2;`

`int b = 4;`

Milyen adattípusban tárolható a következő kifejezés értéke: `b % a == 0`?

- |   |   |
|---|---|
| <input checked="" type="radio"/> boolean<br><input type="radio"/> int<br><input type="radio"/> Integer<br><input type="radio"/> false | <input checked="" type="radio"/> boolean<br><input type="radio"/> int<br><input type="radio"/> Integer<br><input type="radio"/> false |
|---|---|



Earned Points: 1 of 1

20

Melyik érvényes access modifier?

- |  |  |
|--|--|
| <input checked="" type="radio"/> protected<br><input type="radio"/> static<br><input type="radio"/> void<br><input type="radio"/> closed | <input checked="" type="radio"/> protected<br><input type="radio"/> static<br><input type="radio"/> void<br><input type="radio"/> closed |
|--|--|



Earned Points: 1 of 1

21

Válaszd ki az igaz állítást!

- |  |  |
|--|--|
| <input type="radio"/> Az osztályok tagjai kizárólag az attribútumok (fields), metódusok és getter/setterek.<br><input type="radio"/> Az attribútumok (fields) speciális metódusok.<br><input checked="" type="radio"/> A konstruktorok felelősek az objektum állapotának inicializálásáért.<br><input type="radio"/> A metódusok tárolják az attribútumok (fields) értékeit. | <input type="radio"/> Az osztályok tagjai kizárólag az attribútumok (fields), metódusok és getter/setterek.<br><input type="radio"/> Az attribútumok (fields) speciális metódusok.<br><input checked="" type="radio"/> A konstruktorok felelősek az objektum állapotának inicializálásáért.<br><input type="radio"/> A metódusok tárolják az attribútumok (fields) értékeit. |
|--|--|




Earned Points: 1 of 1

22 Melyik annotáció használható (opcionálisan) annak jelzésére, hogy az adott függvény implementációja eltér egy superclass-ban szereplő implementációtól?

- ☒ (X) @Override
- ☐ ( ) @Different
- ☐ ( ) @Polymorph
- ☐ ( ) @Implement


- ☒ (X) @Override
- ☐ ( ) @Different
- ☐ ( ) @Polymorph
- ☐ ( ) @Implement

 Earned Points: 1 of 1

23 Melyik írja ki a konzolra a String msg változó értékét?

- ☒ (X) System.out.println(msg)
- ☐ ( ) System.console.write(msg)
- ☐ ( ) msg.toString()
- ☐ ( ) System.print("msg")


- ☒ (X) System.out.println(msg)
- ☐ ( ) System.console.write(msg)
- ☐ ( ) msg.toString()
- ☐ ( ) System.print("msg")

 Earned Points: 1 of 1

24 Melyik helyes main() deklaráció?

- ☐ ( ) public static int main(String[] args)
- ☐ ( ) public final int main(String args)
- ☐ ( ) public final void main(String args[])
- ☒ (X) public static void main(String[] args)


- ☐ ( ) public static int main(String[] args)
- ☐ ( ) public final int main(String args)
- ☐ ( ) public final void main(String args[])
- ☒ (X) public static void main(String[] args)

 Earned Points: 1 of 1

25 Mi a public static int countVowels(String word) függvény visszatérési típusa?

- ☒ (X) int
- ☐ ( ) String
- ☐ ( ) static
- ☐ ( ) public

- ☒ (X) int
- ☐ ( ) String
- ☐ ( ) static
- ☐ ( ) public

 Earned Points: 1 of 1

## Respondent Summary

Overall Time Used: 00:20:42  
Score Percentile: 92nd  
Overall Result: Pass

**Final Score: 96%**