

 <b>OAK 3</b> ACADEMY	<b>Datum:</b> 20/12/2017 <b>Opleiding:</b> Java Developer <b>Lesmodule:</b> OCA <b>Test:</b> OCA deel 1
<b>Resultaat:</b> ..... / 15	<b>Naam:</b> .....

### Instructies

Deze test is een gesloten boek test. Deze test bevat geen giscorrectie. Elk juist antwoord is 1 punt waard. Als er meerdere antwoordmogelijkheden zijn, krijg je 1 punt als je alles correcte antwoorden hebt aangeduid.

Veel succes.

```
1. public class EnumTest{
    enum Languages{ JAVA, DOTNET, PHP };
    public static void main(String[] args){
        Languages myLanguage = Languages.JAVA;
        System.out.println(myLanguage.ordinal());
    }
}
```

What will be the output?

- 0
- 1
- 2
- Compilation fails
- An exception is thrown at runtime

```
2. public class Superman extends SuperHero {
    public static void main (String [] args) {
        String name = "Superman";
        String superpower = "fly";
        System.out.println(whichSuperHero(name, superpower));
    }

    class SuperHero{
        String whichSuperHero(String name, String superpower){return name + " " +
        "+ superpower;}
    }
}
```

What will be the output?

- Superman fly
- Compilation fails
- An exception is thrown at runtime

```
3. class Fruit {  
    Haha haha;  
}  
class Apple extends Fruit { }  
class Haha { Fruit f; }
```

Which are true? (Choose all that apply)

- Apple IS-A Haha and IS-A Fruit
- Fruit IS-A Apple and HAS-A Haha
- Haha HAS-A Apple and Appel HAS-A Fruit
- Apple HAS-A Fruit and Haha HAS-A Fruit
- Fruit HAS-A Haha and Apple IS-A Fruit
- Haha HAS-a Fruit and Apple HAS-A Haha

```
4. // INSERT CODE HERE  
public class DoubleTest {  
    public static void main(String[] args){  
        System.out.println(Double.MIN_VALUE);  
    }  
}
```

Which inserted independently at line 1, compiles? (Choose all that apply)

- import static java.lang;
- import static java.lang.Double;
- import static java.lang.Double.\*;
- import static java.lang.Double.MIN\_VALUE;
- import static java.lang.Double.MIN\_VALUE.\*;
- None of the above!

```
5. class Mother {  
    protected Mother() { System.out.println("Created a Mother"); }  
}  
public class Child extends Mother {  
    private Child(){ System.out.println("Created a Child"); }  
    public static void main (String [] args){ new Child(); }  
}
```

What is the result?

- Created a Child  
Created a Mother
- Created a Child
- Created a Mother  
Created a Child
- Compilation fails
- An exception is thrown at runtime

```

6. class Programmer {
    Programmer debug() { return this; }
}
class OCA extends Programmer { // INSERT CODE HERE }
```

Which, inserted after the comment 'INSERT CODE HERE' will compile (Choose all that apply)

- Programmer debug() { return this; }
- OCA debug() { return this; }
- Object debug() { return this; }
- int debug() { return 1; }
- int debug (int x ) { return 1;}
- Object debug( int x) { return this; }

```

7. public class Animal {
    public void makeNoise(){
        System.out.println("WHAHAHA");
    }

}
class Bear extends Animal{
    public void makeNoise(){
        System.out.println("GROOOAAAW");
    }

    public static void main (String [] args){
        Animal a = new Bear();
        a.makeNoise();
    }
}
```

What will be the output?

- WHAHADA
- GROOOAAAW
- WHAHADA
- GROOOAAAW
- Compilation fails
- An exception is thrown at runtime

```

8. public class Animal {
    public Animal(){
        System.out.println("I'm an Animal ");
    }

}
class Bear extends Animal{
    public Bear(){
        System.out.println("I'm an awesome Bear");
    }

    public static void main (String [] args){
        Animal a = new Bear();
    }
}
```

What will be the output?

- I'm an Animal
- I'm an awesome Bear
- I'm an awesome Bear
- I'm an Animal
- Compilation fails
- An exception is thrown at runtime

```
9. public class Animal {  
}  
class Bear extends Animal{  
    public static void main (String [] args){  
        // INSERT CODE HERE  
    }  
}
```

Which, inserted after the comment 'INSERT CODE HERE' will compile (Choose all that apply)

- Bear bear = new Bear();
- Bear bear = (Bear) new Animal();
- Animal animal = (Animal) new Bear();
- Animal animal = (Bear) new Bear();
- Animal animal = new Animal();  
 Bear bear = (Bear) animal;
- All of the above

```
10. public class Animal {  
    public void makeNoise(){  
        System.out.println("WHAHAHA");  
    }  
}  
class Bear extends Animal{  
    public void makeNoise(String sound){  
        System.out.println(sound);  
    }  
  
    public static void main (String [] args){  
        Animal a = new Bear();  
        a.makeNoise("GROOOAAAW");  
    }  
}
```

What will be the output?

- WHAHADA
- GROOOAAAW
- No output
- Compilation fails
- An exception is thrown at runtime

```

11. public class Animal {
    public Animal getAnimal(){
        return this;
    }

}
class Bear extends Animal{
    public Bear getAnimal{
        return new Bear();
    }

    public static void main (String [] args){
        Animal a = new Bear();
        Bear b = a.getAnimal();

    }
}

```

Which statement is true?

- This code works perfect
- Compilation fails
- An exception is thrown at runtime

```

12. public class Animal {
    Animal() { main("hi"); }

    public static void main (String [] args){
        System.out.print("2 ");
    }
    public static void main (String args){
        System.out.print("3 " + args);
    }

}

```

What is the result?

- 2 will be included in the output
- 3 will be included in the output
- hi will be included in the output
- Compilation fails
- An exception is thrown at runtime

```

13. public abstract class Animal implements EenInterface {}
public class Bear extends Animal{}
public interface EenInterface { void doStuff(); }

```

In which class you get a compilation fails at the moment?

- Animal
- Bear
- Animal and Bear
- None of them

```
14. public class Sequence {  
    Sequence() { System.out.print("c "); }  
    { System.out.print("y "); }  
    public static void main(String[] args) {  
        new Sequence().go();  
    }  
    void go() { System.out.print("g "); }  
    static { System.out.print("x "); }  
}
```

What is the result?

- cxyg
- cgyx
- xcyg
- xycg
- yxcg
- ycgx

```
15. public class Animal {}  
public class Bear extends Animal{}  
  
public class App {  
  
    static void doStuff(Animal a){  
        System.out.println("Animal is doing stuff");  
    }  
    static void doStuff(Bear b){  
        System.out.println("Bear is doing stuff");  
    }  
  
    public static void main (String [] args){  
        Animal a = new Bear();  
        doStuff(a);  
    }  
}
```

What is the result?

- Animal is doing stuff
- Bear is doing stuff
- Animal is doing stuff  
Bear is doing stuff
- Compilation fails
- An exception is thrown at runtime