Μεθοδολογία Προγραμματισμού

Εισαγωγή στη Java

1. (1 Μονάδες) Δείτε τον ακόλουθο κώδικα.

```
public class SuperClass {
    private int x;
    private int y;
    public SuperClass(int x, int y) {
        this.x = x;
        this.y = y;
    }
    public int getX() {
        return x;
    }
    public int getY() {
        return y;
    }
}
public class SubClass extends SuperClass {
    private int x;
    public SubClass() {
    }
    public int getX() {
        return x;
    }
    public void setX(int x) {
        this.x = x;
    }
}
```

Ο κώδικας αυτός:

Α΄. Είναι λάθος γιατί δε μπορεί η SubClass να έχει πεδία με το ίδιο όνομα με τη SuperClass.

Β΄. Είναι λάθος γιατί δε μπορεί η SubClass να έχει constructor χωρίς παραμέτρους.

Γ΄. Είναι λάθος γιατί δεν καλούμε πουθενά τον constructor της SuperClass. Μια σωστότερη προσέγγιση θα μπορούσε να ήταν για παράδειγμα η ακόλουθη:

```
public SubClass() {
      super(1,2);
   }
```

Δ΄. Ο κώδικας είναι σωστός.

Ε΄. Καμία απάντηση δεν είναι σωστή

2. (1 Μονάδες) Δείτε τον ακόλουθο κώδικα:

```
public class SuperClass {
    private int x;
    public int getX() {
        return x;
    }
    public void setX(int x) {
        this.x = x;
    }
}
public class SubClass extends SuperClass {
    private int x;
    public SubClass() {
    }
    public int getX() {
        return x;
    }
    public void setX(int x) {
        super.setX(2*x);
        this.x = super.getX()+x;
    }
}
```

Αν εκτελεστεί έτσι:

```
SubClass subClass = new SubClass();
subClass.setX(10);
System.out.println(subClass.getX());
```

τι θα τυπώσει;

Α΄. Θα βγάλει λάθος

Β΄. 10

Γ΄. 20

Δ΄. 30

Ε΄. Τίποτα από τα παραπάνω

3. (1 Μονάδες) Δείτε τον ακόλουθο κώδικα:

```
public class SuperClass {
    protected int x;
    public int getX() {
        return x;
    }
    public void setX(int x) {
        this.x = x;
    }
}
public class SubClass extends SuperClass {
    private int x;
    public SubClass() {
    }
    public int getX() {
        return x;
    }
    public void setX(int x) {
        super.setX(2*x);
        this.x = super.x+x;
    }
}
```

Αν εκτελεστεί έτσι:

```
        SubClass subClass = new SubClass();
        subClass.setX(10);
        System.out.println(subClass.getX());
        SuperClass superClass = new SubClass();
        superClass.setX(10);
        System.out.println(superClass.getX());
```

τι θα τυπώσει;

Α΄. Θα βγάλει λάθος

Β΄. 10, 10

Γ΄. 30, 10

Δ΄. 30, 30

Ε΄. Τίποτα από τα παραπάνω

4. (1 Μονάδες) Δείτε τον ακόλουθο κώδικα:

```
public class SuperClass {
    protected int x;
    public SuperClass(int x) {
        this.x = x;
    }
    public int getX() {
        return x;
    }
}
public class SubClass extends SuperClass {
    private int y;
    public SubClass() {
        y = 1;
        super(2);
    }
    public int getY() {
        return x;
    }
}
```

Ο κώδικας αυτός

Α΄. Είναι σωστός

Β΄. Στην getY() επιστρέφουμε ένα x αλλά αυτό δεν είναι δηλωμένο στην κλάση SubClass και έτσι υπάρχει λάθος

Γ΄. Το **super**(2)θα έπρεπε να είναι η πρώτη εντολή στον constructor και όχι η 2η

Δ΄. Τα δύο προηγούμενα (β) και (γ)

Ε΄. Καμία απάντηση δεν είναι σωστή.

5. (1 Μονάδες) Δείτε τον ακόλουθο κώδικα:

```
public class SuperClass {
    public void thisIsAnInstanceMethod() {
        System.out.println("SuperClass␣instance␣method");
    }
    public static void thisIsAClassMethod() {
        System.out.println("SuperClass␣class␣method");
    }
}
public class SubClass extends SuperClass {
    @Override
    public void thisIsAnInstanceMethod() {
        System.out.println("SubClass␣instance␣method");
    }
    public static void thisIsAClassMethod() {
        System.out.println("SubClass␣class␣method");
    }
    public static void main(String[] args) {
        SubClass subClass = new SubClass();
        SuperClass superClass = subClass;
        SuperClass.thisIsAClassMethod();
        SubClass.thisIsAClassMethod();
        superClass.thisIsAnInstanceMethod();
    }
}
```

Ο κώδικας αυτός θα τυπώσει

Α΄. `SuperClass class method`
`SubClass class method`
`SubClass instance method`

Β΄. SuperClass class method
SubClass class method
SuperClass instance method

Γ΄. SubClass class method
SubClass class method
SuperClass instance method

Δ΄. Θα βγάλει λάθος

Ε΄. Καμία απάντηση δεν είναι σωστή.

6. (1 Μονάδες) Ορίστε σε ένα enum τις μέρες τις εβδομάδας με το αγγλικό και το ελληνικό τους όνομα.

7. (1 Μονάδες) Δείτε το documentation της enum και δείτε τι κάνει η συνάρτηση ordinal(). Αν ορίζατε τις μέρες τις εβδομάδας στο enum DAY με τη σειρά SUNDAY, MONDAY, ... τι θα τύπωνε το DAY.ordinal() και τι το SUNDAY.ordinal()

Α′. 7, 0

Β′. 7, 1

Γ′. Το πρώτο είναι λάθος το άλλο θα έβγαζε 1

Δ′. Το πρώτο είναι λάθος το άλλο θα έβγαζε 0

8. (1 Μονάδες) Δείτε τον παρακάτω κώδικα:

```
public class Test
{
    public static void aMethod() throws Exception
    {
        try {
            throw new Exception();
        }
        finally  {
            System.out.print("finally␣");
        }
    }
    public static void main(String args[])
    {
        try       {
            aMethod();
        }    catch (Exception e)    {
            System.out.print("exception␣");
        }
        System.out.print("finished");
    }
}
```

Ο κώδικας αυτός θα τυπώσει

Α′. finally

Β′. exception finally

Γ′. finally exception finished

Δ′. exception finished

Ε′. Καμία από τις απαντήσεις δεν είναι σωστή

9. (1 Μονάδες) Ο παρακάτω κώδικας

```
static String readFirstLineFromFile(String path) throws IOException {
    try (BufferedReader br =
                   new BufferedReader(new FileReader(path))) {
        return br.readLine();
    }
}
```

Α΄. Είναι ισοδύναμος με αυτόν:

```
static String readFirstLineFromFile(String path)
                                                        throws IOException {
    BufferedReader br = new BufferedReader(new FileReader(path));
    try {
        return br.readLine();
    } catch {
        if (br != null) throw new IOException();
    }
}
```

Β΄. Είναι ισοδύναμος με αυτόν:

```
static String readFirstLineFromFile(String path)
                                                        throws IOException {
    BufferedReader br = new BufferedReader(new FileReader(path));
    try {
        return br.readLine();
    } finally {
        if (br != null) br.close();
    }
}
```

Γ΄. Βγάζει μήνυμα λάθους κατά τη μεταγλώττιση (Compile time error)

Δ΄. Βγάζει μήνυμα λάθους κατά την εκτέλεση (Runtime error)

Ε΄. Τίποτα από τα παραπάνω

10. (1 Μονάδες) Όλες οι κλάσεις της Java κληρονομούν από την κλάση Object η οποία παρέχει κάποιες μεθόδους. Κάποιες από αυτές φαίνονται στο ακόλουθο παράδειγμα. Μπορείτε να μαντέψετε τι θα δείξει στο πρόγραμμα;

```
package pmtest;
public class MyNumber {
    int x;
    public MyNumber(int x) {
        this.x = x;
    }
    public int getX() {
        return x;
    }
     public static void main(String[] args) {
        MyNumber n1 = new MyNumber(3);
        MyNumber n2 = new MyNumber(3);
        // Δοκιμάστεκαιχωρίστιςπαρενθέσεις     (n1==n2). Δουλεύει; Γιατί;
        System.out.println("n1 == n2 is " + (n1==n2));
        //equals() is a method of Object
        System.out.println("n1 equals n2 is " + n1.equals(n2));
        //hashCode() is a method of Object
        System.out.println("n1.hash() is " + n1.hashCode());
        System.out.println("n2.hash() is " + n2.hashCode());
        //toString() is a method of Object
        System.out.println("n1 is:" + n1.toString());
    }
}
```

Α′. n1 == n2 is false
n1 equals n2 is false
n1.hash() is <a number>
n2.hash() is <a different>
n1 is:pmtest.MyNumber@<a number>

Β′. n1 == n2 is true
n1 equals n2 is true
n1.hash() is <a number>
n2.hash() is <a different number>
n1 is:pmtest.MyNumber@<a number>

Γ′. n1 == n2 is true
n1 equals n2 is true
n1.hash() is <a number>
n2.hash() is <the same number>
n1 is:pmtest.MyNumber@<a number>

Δ′. n1 == n2 is false
n1 equals n2 is true
n1.hash() is <a number>
n2.hash() is <the same number>
n1 is:pmtest.MyNumber@<a number>

Ε′. Τίποτα από τα παραπάνω