1. Consider the following method (with line numbers):

```java
1. public boolean justAMethod(int i, int j, k){
2.     boolean n = false;
3.     int m=0
4.     if ( i > j)
5.         return true;
6.     return n;
7. }
```

There are syntax errors in:

a. line 3
b. lines 1, 2 and 3
c. lines 1, 2 and 5
* [d.] lines 1 and 3
e. lines 1 and 2

2. What sort of variables are used to store the state of an individual object?

a. Local variables.
* [b.] Field variables.
c. Class variables.
d. Argument variables.
e. Method variables.

3. Variables of which of these types can *not* be added using the + operator?

a. int
b. double
* [c.] boolean
d. char
e. String
4. Consider the following method:

```java
public int aMystery(int i, int j){
    int k = 0;
    k = i;
    i = j;
    j = k;
    return j;
}
```

What does `aMystery(10,15)` return?

- a. 15
- [b.] 10
- c. 25
- d. 5
- e. 0

5. What does `booleanMystery(false,true)` return?

```java
public int booleanMystery(boolean b1, boolean b2)
{
    if (b1 && b2) {
        return 100;
    }
    else if (b1 || b2) {
        return 200;
    } else {
        return 300;
    }
}
```

What does `booleanMystery(false,true)` return?

- a. 100
- [b.] 200
- c. 300
- d. true
- e. false
6. Consider the following block of code, where variables a, b and c each store integer values:

```java
if (a < b) {
    if (a < c) {
        System.out.println(a);
    } else {
        System.out.println(c);
    }
} else if (b < c) {
    System.out.println(b);
} else {
    System.out.println(c);
}
```

Which one of the following values for the variables will cause the value in variable b to be printed?

- a. a = 1; b = 2; c = 3;
- b. a = 3; b = 2; c = 1;
- c. a = 1; b = 3; c = 2;
- d. a = 2; b = 1; c = 3;
- e. a = 2; b = 3; c = 2;

7. The following method, which is intended to find the shortest name String in the collection SongNames, is incorrect.

```java
public int shortestName( ArrayList<String> songNames ) {
    int min = 0;
    for (String name : songNames) {
        if ( name.length() < min) {
            min = name.length();
        }
    }
    return min;
}
```
Which of the following statements best describes when `shortestName` fails? An empty String is one such as String noName = ";

a. It fails whenever the collection `songNames` contains an empty String.
b. It fails whenever the collection `songNames` contains only empty Strings.
   *[c.] It fails whenever the collection `songNames` contains no empty Strings.
d. It fails whenever more than one element of the collection `songNames` has the same length.
e. It fails whenever the first element of the collection `songNames` is the shortest.

8. What will the method call `sillyMethod(645)` return, where the method is defined as follows:

```java
public int sillyMethod(int n) {
    int m = 0;
    while (n != 0) {
        m = n % 10;
        n = n / 10;
    }
    return m;
}
```

a. 15
   *[b.] 6
c. 5
d. 546
e. 0
9. What is the value of array element a[99] after executing these statements?

```java
int[] a = new int[100];
a[0] = 1;
for (int i = 1; i < a.length; i++) {
    a[i] = 1 - a[i-1];
}
```

a. -2  
b. -1  
*c.* 0  
d. 1  
e. 2

10. Consider the following class definition

```java
public class BankAccount {

    private int balance;
    private int total;

    public BankAccount(int balance) {
        this.balance = balance;
        total = balance;
    }

    public void deposit(int amount) {
        balance = balance + amount;
        total = total + amount;
    }

    public void withdraw(int amount) {
        balance = balance - amount;
        total = total + amount;
    }

    public int getBalance() {
        return balance;
    }
}
```
What would b1.getTotal() and b2.getTotal() return after executing these statements?

```java
BankAccount b1 = new BankAccount(1000);
BankAccount b2 = new BankAccount(500);
b1.deposit(1500);
b2.withdraw(200);
b1.deposit(b2.getBalance());
```

a. 1700 and 700  
b. 2500 and 200  
c. 2500 and 700  
d. 1000 and 500  
*e.* 2800 and 700

11. Consider a method in the class `TicketMachine` (as used as in lectures and labs) with the signature

```java
public boolean higherPrice(TicketMachine other) {
    return price > other.getPrice();
}
```

does that should test whether the target object (this) has a strictly higher price than the argument object (other). Strictly means that the method should return false if the prices are equal.

Three methods are proposed for this purpose:

```java
public boolean higherPrice(TicketMachine other) {
    return price > other.getPrice();
}
```

```java
public boolean higherPrice(TicketMachine other) {
}
```
if (this.getPrice() > other.getPrice()) {
    return this;
} else {
    return other;
}
}

public boolean higherPrice(TicketMachine other) {
    if (price <= other.getPrice()) {
        return false;
    } else {
        return true;
    }
}

Which of them will work correctly?

a. All three of them
b. The 1st and 2nd only
   *[c.] The 1st and 3rd only
d. The 2nd and 3rd only
e. The 3rd only
12. How many of these statements about constructors are true?

- All constructors in a class must have different return types.
- All constructors in a class must have different argument types.
- Every class must have at least one constructor.
- Constructors cannot initialise field variables.

a. 0  
   *[b.] 1  
c. 2  
d. 3  
e. 4

13. If I want to create and populate an array list of n StudentMarks, how many calls to new do I need?

a. 1  
b. n  
   *[c.] n+1  
d. 2n  
e. 2n+1

14. What is the value of red == xxx after these two statements, and why?

java.awt.Color red = new java.awt.Color(255,0,0);  
java.awt.Color xxx = new java.awt.Color(255,0,0);  

*[a.] false, because they are references to different objects, even though the objects contain identical data.  
b. true, because they both represent the colour red.  
c. A reference with the same value as xxx, because an assignment expression has the same value as the expression on the right hand side of the assignment.  
d. We cannot tell, because it depends on details of the implementation that are hidden from the users.  
e. A runtime error occurs, because == can only be used for primitive types.
15. How many of these statements apply to a large program written in good object-oriented style?

- The problem is decomposed into several classes.
- Each class provides a narrow range of well-defined services.
- Each class hides its implementation details as far as possible.
- Objects communicate as little as possible at runtime.

a. 0
b. 1
c. 2
*d.* 3
e. 4