





Quiz Solutions

Note: these answers are based on the ‘A’ version of the quiz. If you wrote the ‘B’ version, the same questions appear, but in a different order.

In some cases, the wording may be ‘inverted’ so that a true answer in the ‘A’ exam becomes a false answer in the ‘B’ exam. Check the wording carefully.





2. Which one of the following is used in UML diagrams to signal that a member is constant?
(a) underline (b) *italics* (c) **boldface** (d) CAPITALIZE (e) none of the above
3. When a subclass constructor calls a superclass constructor (using `super()`), it is said to be:
(a) derived (b) overridden (c) chained (d) static (e) none of the above
4. In debug mode, to *step return* from a method, you would select:
(a)  (or F5) (b)  (or F6) (c)  (or F7) (d)  (or Ctl-F11) (e) none of the above
5. Which one of the following statements is **TRUE** of the *enhanced for* loop
(a) Like the regular *for* loop, the *enhanced for* loop uses the semicolon (;) in its format
(b) You can only count up through the elements of an *enhanced for*; you cannot count down.
(c) You can both get data *out* of an array, and set data *into* an array using the *enhanced for*.
(d) You cannot break out of an *enhanced for* using the `break` command
(e) none of the above



Sample Quiz Solutions

6. True (a) or False (b): a project can only have one `main()` method
7. True (a) or False (b): Subclasses do not inherit constructors.
8. True (a) or False (b): you cannot inherit from a `final` class
9. Which one of the following is *NOT* a method of the `Object` class
(a) `clone()` (b) `equals()` (c) `toString()` (d) `finalize` (e) none of the above
10. True (a) or False (b): The `File` class contains methods that allow the user to add or delete items stored a file on, for example, the hard drive or USB drive.
11. If a method is declared *without* an access modifier, then, assuming it is in a public class, its visibility is limited to
(a) the class (b) the package (c) the project (d) subclasses
(e) it can't be done; methods must always have an access modifier



12. Which one of the following symbols is used to indicate an *annotation* in Java?
(a) - (b) + (c) # (d) @ (e) none of the above
13. Which one of the following is *NOT* a typical use for a static member?
(a) It can serve as a common location that all objects in the same class can reference
(b) It can be used when an operation is not instance-dependent
(c) It can be used to reduce space requirements, for example, in systems with restricted RAM space
(d) It can be used to build factory methods
(e) none of the above
14. **True** (a) or **False** (b): When a property has been declared `final`, its value must be assigned at the point of declaration, since a value cannot be assigned afterward
15. Which one of the following symbols would be used to indicate a casual, one-to-one relationship between two objects, for example a factory method and the objects it produced
(a)  (b)  (c)  (d)  (e) none of the above

16. The main purpose of Unit Testing is to:

- (a) Test for errors that may arise due to the unexpected interaction between different units of code by looping through all possible combinations of a program's inputs
- (b) Test for errors that result from different units of measurement being used in different parts of a program
- (c) Test for errors that arise at the interface between different components, or 'units', of code
- ☒ (d) Test for errors in code that was previously working, but which may now be flawed due to further code development or existing code modification
- (e) None of the above

17. **True** (a) or **False** (b): In general, an object instantiated from a superclass will have at least as many or more members than an object instantiated from one of its subclasses

18. **True** (a) or **False** (b): In order to instantiate a new Scanner in your code, you must first use `import java.util.Scanner;` at the top of the class, otherwise the JVM will not be able to locate the appropriate code at runtime



The following code is used in the next two questions. Assume the classes MySuperClass and MySubClass exist in their respective Java files.

```
public class MySuperClass {  
    private int x=0;  
    public void setX(int x){this.x = x;}  
    public int getTwiceAsMuch{return 2*x;}  
}  
  
public class MySubClass extends MySuperClass {  
    // ...assume constructors and additional subclass members go here  
}
```

19. **True (a) or False (b):** An object instantiated from MySubClass will *not* have access to the toString() method, only to the methods extended from MySuperClass, i.e. setX() and getTwiceAsMuch() – but not toString().
20. **True (a) or False (b):** MySubClass inherits all of MySuperClass's non-private members, which it may then override



Part B: Terminology – worth 1 mark each

FILL IN THE SINGLE BEST ANSWER—ONE WORD OR SYMBOL IN THE EACH SPACE PROVIDED BELOW.

USE ONE WORD ONLY IN EACH SPACE; DO NOT USE ACRONYMS

21. . In Java, single inheritance ensures that each subclass only has one parent
22. Rather than attempting to change the name of a file and all its dependencies (like its class name and constructors) directly, you should refactor it instead using the Eclipse menu.
23. The `import` and `package` statements at the top of a class are necessary if the JVM is going to find the Fully Qualified Name of each library class required by your code, otherwise your Java program will generate run-time errors on a host PC.
24. The binding together of properties and related methods in an object is known as encapsulation
25. Two methods of a class which have the same name but a different parameter list are said to be overloaded
26. In an array declaration prototype like: `datatype[] identifier = new datatype[arraysize];`
`arraysize` indicates the total number of elements in the array



Part C: Written Answers – worth 9 marks total

```
import java.util.Scanner;
public class TestCar {

    1 public void main(String[] args) {
        Scanner in = new Scanner(); 2

        Car c1 = new Car(20000L, 1800);
        outputMsg(c1);

        System.out.println("Enter the distance " +
            "travelled by Car2 (in metres) in one hour");
        Car c2 = new Car(in.nextLong()); 3
        outputMsg(c2);

        System.out.println("Car1 is " +
            ((getKPH(c1) < getKPH(c2))?"faster":"slower") +
            " than Car2"); 4
    }

    private static double getKPH(Car c) {
        long km = c.getDistance()/1000; 5
        int hrs = c.getTime()/(3600);
        return(km/hrs); 6
    }

    private static void outputMsg(Car c) { 7
        System.out.println("Car" + getThisCarNum() +
            " has velocity " + getKPH(c) + " kph");
    }
}
```

2. Missing System.in as parameter to Scanner() constructor
3. Should be nextLong(), not nextLong, which would be a property rather than method
4. "faster"/"slower" logic is backwards
5. Calculation is always rounded down to nearest integral value; hence if time=3599 seconds, this is stored as 0 hours
6. Possible divide by zero errors if time not set correctly, or roundoff occurs and hrs=0
7. Must use c.getThisCarNum() to access instance method in a Car object

```

public class Car {
    private long distance;
    private int time, thisCarNum;
    private static int carNum = 1;

    public Car() {this(0)} 8
    public Car(long distance) {this(distance);}
    public Car(long distance, int time) {
        setDistance(distance); setTime(time);
        setThisCarNum(); 9
    }

    public long getDistance(){return distance;}
    public void setDistance(long distance)
        {this.distance = distance;}

    public int getTime() {return time;}
    public void setTime(int time)
        {this.time = time;}

    public int getThisCarNum()
        {return thisCarNum;}
    private void setThisCarNum() 10
        {carNum = thisCarNum;}

    public int incrCarNum() {carNum++;}
} 11 12

```

- 8) no-arg constructor missing semi-colon at end
- 9) Need to call incrCarNum whenever a new Car is instantiated, otherwise each car has the same number (1)
- 10) Need to set thisCarNum = carNum if you're going to store the car number with the instance variable thisCarNum
- 11) incrCarNum() should be static, since we want one method only to control the static value carNum. (Otherwise, the method could be overridden.)
- 12) Should return void, not int