1. The this reference refers to
   1. The current executing object
   2. The first parameter in a method
   3. the current class
   4. a reference that doesn’t point to any object
   5. the object that called the current method
2. Which of the following should be used for an object reference variable that does not refer any object ?
   1. this
   2. static
   3. null
   4. 0
   5. -1
3. pWhich of the following expressions can be used to test whether two objects o1 and o2 are aliases to each other?
   1. o1==o2
   2. o1.equals(o2)
   3. o2.equals(o1)
   4. o1.compareTo(o2) ==0;
4. Given the declarations

String s1 =”Scooby”;

String s2 =”Scooby”;

Multiple answers are possible.

Which of the following statements are true ?

* 1. S1 and s2 are aliases to each other.
  2. s1==s2 are true
  3. s1.equals(s2) is true
  4. s1 and s2 are the same because they have same number of characters.

1. Which of the following is often static ?
   1. Local variables
   2. Instance variables
   3. Parameters to methods
   4. Object reference variables
   5. Constants
2. A class that implements the interface must implement which methods of interface ?
   1. Only those that return a value
   2. Only those that are not abstract.
   3. Only those that are marked with the word required
   4. The class is not required to implement any methods.
   5. All methods must be implemented.
3. A method that does not have an implementation :
   1. Is not allowed by the Java compiler
   2. Takes no parameters and returns the integer 0
   3. Is a static method
   4. Is a null method
   5. Is an abstract method

True/False

1. A reference variable that does not currently point to an object is called a null referene.
2. The this reference lets an object to refer to itself.
3. When an object is passed to a method , what is actually passed is a reference to that object.
4. Static variables and methods are accessed through a class rather than through an instance of a class.
5. A static method may use the instance variable in the same class.
6. Constructors may not be static.
7. A class may implement more than one interface.

**public class** Test {  
  
 **private** String **testName**;  
  
 **public** Test(String testName) {  
 **this**.**testName** = testName;  
 }  
  
 **public void** printTestName(){  
 System.***out***.println(**this**.**testName**);  
 }  
  
 **public static void** main(String[] args) {  
  
 Test test1 = **new** Test(**"Java Test"**);  
 test1.printTestName();  
 Test test2 = **new** Test(**"Java Test1"**);  
 test2.printTestName();  
  
   
 }  
}

1. What will get printed when this code **test1.printTestName();**

executes and why ?

1. What will get printed when this code test2**.printTestName();**

executes and why ?

Integer integer1 = **new** Integer(3);  
Integer integer2 = **new** Integer(3);  
  
a) System.***out***.println(integer1==integer2);

What is the output of the above code ?

b) System.***out***.println(integer1.equals(integer2));

What is the output of above code ?

**public class** AliasTest {  
  
 **private int number**;  
  
 **public** AliasTest(**int** number) {  
 **this**.**number** = number;  
 }  
  
 **public static void** main(String[] args) {  
  
 AliasTest aliasTest1 = **new** AliasTest(1);  
 AliasTest aliasTest2 = **new** AliasTest(1);

**}**

1. What is the output of the below code ?

System.***out***.println(aliasTest1 == aliasTest2);

**public class** Params {  
 **public void** modParams(**int** x , String s1, String s2){  
  
 x=2;  
  
 }  
  
 **public static void** main(String[] args) {  
 Params params = **new** Params();  
 **int** k=1;

String s1=Hi”;

String s2=Hello”;  
 params.modParams(k,s1,s2);  
  
 System.***out***.println(k + **" "** + s1 + **" "** + s2 );  
 }  
}

1. 1
2. 11
3. 2
4. 3
5. Create an interface called Visible that includes two methods **makeVisible** and **makeInvisible** .
   1. Create a class named VisibleImpl and implement the interface.
   2. Implement the method that **makeVisible** and return true.
   3. Implement the method that **makeInvisible** and return false.
   4. Create the main method and create an object of VisibleImpl
   5. Call the **makeVisible** and **makeInvisible** using the object that got created in step d and print the values returned from the makeVisible and makeInvisible.