Suppose the xMethod() is invoked from a main method in a class as follows, xMethod() is _________ in the class.

```java
public static void main(String[] args) {
    xMethod();
}
```

- A. a static method
- B. an instance method
- C. a static method or an instance method
A static method in a class can access the instance variables in the same class.

- A. false
- B. true
Which of the following statements are true?

☐ A. Every class has a default constructor.

☐ B. At least one constructor must always be defined explicitly.

☐ C. The default constructor is a no-arg constructor.

☐ D. A default constructor is provided automatically if no constructors are explicitly declared in the class.
What is the output of the following code?

```java
public class Test {
    public static void main(String[] args) {
        String s1 = new String("Welcome to Java!");
        String s2 = new String("Welcome to Java!");
        if (s1 == s2)
            System.out.println("s1 and s2 reference to the same String object");
        else
            System.out.println("s1 and s2 reference to different String objects");
    }
}
```
What is the output of running class C?

class A {
    public A() {
        System.out.println("The default constructor of A is invoked");
    }
}

class B extends A {
    public B() {
        System.out.println("The default constructor of B is invoked");
    }
}

public class C {
    public static void main(String[] args) {
        B b = new B();
    }
}
class Super {
    public String m() {
        return "Super";
    }
}

class Sub extends Super {
    public String m(String s) {
        return s + "Sub";
    }
}

public class Base {
    public static void main(String[] args) {
        Sub sub = new Sub();
        System.out.println(sub.m());
    }
}
Assume int[][] x = {{1, 2}, {3, 4, 5}, {5, 6, 5, 9}}, what are x[0].length, x[1].length, and x[2].length?

A. 2, 3, and 3
B. 2, 3, and 4
C. 2, 2, and 2
D. 3, 3, and 4
E. 3, 3, and 3
The printout from the following code is _________.

```java
import java.util.ArrayList;
import java.util.List;

public class Main {
    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<String>();
        list.add("New York");
        ArrayList<String> list1 = new ArrayList<String>();
        list.add("Atlanta");
        list1.add("Dallas");
        System.out.println(list1);
    }
}
```

A. [New York, Dallas]
B. [New York, Atlanta]
C. [New York]
D. [New York, Atlanta, Dallas]
What is the output of running class C?

class A {
  public A() {
    System.out.println( "The default constructor of A is invoked");
  }
}

class B extends A {
  public B() {
    System.out.println( "The default constructor of B is invoked");
  }
}

class C {
  public static void main(String[] args) {
    B b = new B();
  }
}

A. "The default constructor of A is invoked" followed by "The default constructor of B is invoked"
B. Nothing displayed
C. "The default constructor of B is invoked" followed by "The default constructor of A is invoked"
D. "The default constructor of B is invoked"
E. "The default constructor of A is invoked"
What is the output of the following code:

```java
public class Test {
    public static void main(String[] args) {
        String s1 = new String("Java");
        String s2 = new String("Java");
        System.out.print((s1 == s2) + " " + (s1.equals(s2)));
    }
}
```

A. false true
B. true false
C. false false
D. true true
Which statements are most accurate regarding the following classes?

class A {
    private int i;
    protected int j;
}

class B extends A {
    private int k;
    protected int m;
    // some methods omitted
}

- A. In the class B, an instance method can only access i, j, k, m.
- B. In the class B, an instance method can only access j, m.
- C. In the class B, an instance method can only access k, m.
- D. In the class B, an instance method can only access j, k, m.
A instance of a subclass is also an instance of its superclass.

- A. false
- B. true
In OOP, a reference variable can reference a subtype object. This is called _____.

A. encapsulation
B. inheritance
C. polymorphism
D. abstraction
The following code causes Java to throw ________.

```java
int number = Integer.MAX_VALUE + 1;
```

A. RuntimeException  
B. Throwable  
C. Error  
D. Exception  
E. no exceptions
What exception type does the following program throw?

```java
public class Test {
    public static void main(String[] args)
    {
        int[] list = new int[5];
        System.out.println(list[5]);
    }
}
```

A. No exception
B. StringIndexOutOfBoundsException
C. ArithmeticException
D. ClassCastException
E. ArrayIndexOutOfBoundsException
What is displayed on the console when running the following program?

class Test {
    public static void main(String[] args) {
        try {
            method();
            System.out.println("After the method call");
        } catch (NumberFormatException ex) {
            System.out.println("NumberFormatException");
        } catch (RuntimeException ex) {
            System.out.println("RuntimeException");
        }
    }
    static void method() {
        String s = "5.6";
        Integer.parseInt(s);
        // Cause a NumberFormatException
        int i = 0;
        int y = 2 / i;
        System.out.println("Welcome to Java");
    }
}

A. The program displays NumberFormatException followed by RuntimeException.
B. The program displays NumberFormatException followed by After the method call.
C. The program displays RuntimeException.
D. The program displays NumberFormatException.
E. The program has a compilation error.
Analyze the following code.

```java
public class Test {
    public static void main(String[] args) {
        Fruit[] fruits = {new Fruit(2), new Fruit(3), new Fruit(1)};
        java.util.Arrays.sort(fruits);
    }
}

class Fruit {
    private double weight;
    public Fruit(double weight) {
        this.weight = weight;
    }
}
```

A. The program has a runtime error on Line 4 because the Fruit class does not implement the java.lang.Comparable interface and the Fruit objects are not comparable.

B. The program has a compile error on Line 4 because the Fruit class does not implement the java.lang.Comparable interface and the Fruit objects are not comparable.

C. The program has a compile error because the Fruit class does not have a default constructor.

D. The program has a runtime error on Line 3 because the Fruit class does not have a default constructor.
A subclass cannot extend more than one class, but may implement any number of interfaces.

- A. false
- B. true
All methods in an interface are abstract.

- A. true
- B. false
You cannot create an instance of an abstract class using the new operator.

A. true
B. false
________ is a reference type.

☐ A. An interface type
☐ B. An array type
☐ C. A primitive type
☐ D. A class type
```java
class Count {
    int count;
    public Count() {
        count = 0;
    }
}

public class Foo {

    public static void main(String[] args) {
        Count myCount = new Count();
        for (int i = 0; i < 100; i++) {
            increment(myCount);
        }
        System.out.println("myCount.count = " + myCount.count);
    }

    public static void increment(Count c) {
        c.count++;
    }
}
```
Analyze the following recursive method.

```java
public static long factorial(int n) {
    return n * factorial(n - 1);
}
```
public static int m(int value) {
    if (value >= 0)
        return 5 * m(value - 2);
    else
        return 1;
}

What value is returned when invoking m(5)?

- A. 75
- B. 25
- C. 100
- D. 225
- E. 125
What are the base cases in the following recursive method?

```java
public static void xMethod(int n) {
    if (n > 0) {
        System.out.print(n % 10);
        xMethod(n / 10);
    }
}
```

- A. n <= 0
- B. no base cases
- C. n > 0
- D. n < 0
What is the printout of invoking `xfunction(1234)`?

```java
public static void xfunction(int n) {
    if (n > 0) {
        System.out.print(n % 10 + " ");
        xfunction(n / 10);
    }
}
```
A recursive method can always be converted into a nonrecursive method using iterations.

- A. false
- B. true
public class ThisConstructor {
    private int x;

    public ThisConstructor(String s){
        this();
        System.out.print(s);
        System.out.print(" this ");
    }

    public ThisConstructor(int x){
        this(" is ");
    }

    public ThisConstructor(){
        System.out.print(" a test ");
    }

    public static void main(String[] args) {
        int val = 2;
        ThisConstructor t = new ThisConstructor(val);
    }
}
Which of the following statements are true?

☐ A. Infinite recursion can occur if recursive call does not reduce the problem in a manner that allows it to eventually converge into the base case.

☐ B. A recursive method is invoked differently from a non-recursive method.

☐ C. Every recursive method must have a return value.

☐ D. Every recursive call reduces the original problem, bringing it increasingly closer to a base case until it becomes that case.

☐ E. Every recursive method must have a base case or a stopping condition.
Which of the following statements are true?

☐ A. In some cases using recursion enables you to give a natural, straightforward, simple solution to a problem that would otherwise be difficult to solve.

☐ B. A recursive method can always be replaced by a non-recursive method.

☐ C. Recursive methods usually take more memory space than non-recursive methods.

☐ D. Recursive methods run faster than non-recursive methods.