Q>

```
public class Breaker {
 static String o = "";
 public static void main(String[] args) {
       z: o = o + 2;
      for (int x = 3; x < 8; x++) {
            if (x == 4)
                  break;
            if (x == 6)
                  break z;
            o = o + x;
      }
      System.out.println(o);
 }
}
```

Labels are assigned for methods, not for statements
E.g:

```
Task:
for(int i=0; i<10; i++){
  if (i==8){
    continue Task;
    (or)
    break Task;
  }
}
```

What is the result?

A. 23
B. 234
C. 235
D. 2345
E. 2357
F. 23457
G. Compilation fails.

Answer: G

Q>

```
public class Tahiti {
      Tahiti t;
      public static void main(String[] args) {
            Tahiti t = new Tahiti();
            Tahiti t2 = t.go(t);
            t2 = null;
            // more code here 11
      }
      Tahiti go(Tahiti t) {
            Tahiti t1 = new Tahiti();
            Tahiti t2 = new Tahiti();

            t1.t = t2;
            t2.t = t1;
            t.t = t2;
            return t1;
      }
}
```

When line 11 is reached, how many objects are eligible for garbage collection?

A. 0
B. 1
C. 2
D. 3

Answer: 0(island of isolation)

Q>

```
public class ItemTest {
      private final int id;
      public ItemTest(int id) {
            this.id = id;
      }
      public void updateId(int newId) {
            id = newId;
      }
      public static void main(String[] args) {
                  ItemTest fa = new ItemTest(42);
                  fa.updateId(69);
                  System.out.println(fa.id);
      }
}
```

What is the result?
A. Compilation fails.
B. An exception is thrown at runtime.
C. The attribute id in the ItemTest object remains unchanged.
D. The attribute id in the ItemTest object is modified to the new value.
E. A new ItemTest object is created with the preferred value in the id attribute.

Answer: A

Q>

```
class Foo {
      private int x;
      public Foo( int x ){ this.x = x;}
      public void setX( int x ) { this.x = x; }
      public int getX(){ return x;}
}
public class Gamma {
      static Foo fooBar(Foo foo) {
            foo = new Foo(100);
            return foo;
      }
      public static void main(String[] args) {
             Foo foo = new Foo( 300 );
            System.out.println( foo.getX() + "-");
            Foo fooFoo = fooBar(foo);
            System.out.println(foo.getX() + "-");
            System.out.println(fooFoo.getX() + "-");
            foo = fooBar( fooFoo);
            System.out.println( foo.getX() + "-");
            System.out.println(fooFoo.getX());
      }
}
```

What is the output of the program shown in the exhibit?
A. 300-100-100-100-100
B. 300-300-100-100-100
C. 300-300-300-100-100
D. 300-300-300-300-100

Answer: B

Q>

```
interface Fish {}
class Perch implements Fish {}
class Walleye extends Perch {}
class Bluegill {}
public class Fisherman {
 public static void main(String[] args) {
      Fish f = new Walleye();
      Walleye w = new Walleye();
      Bluegill b = new Bluegill();
      if (f instanceof Perch)
            System.out.print("f-p ");
      if (w instanceof Fish)
            System.out.print("w-f ");
      if (b instanceof Fish)
            System.out.print("b-f ");
 }
}
```

What is the result?

A. w-f
B. f-p w-f
C. w-f b-f
D. f-p w-f b-f
E. Compilation fails.
F. An exception is thrown at runtime

Answer: B