Quiz: Nested Loop

Which picture is drawn by the loop below?

1) `canvas.setColor(Color.BLUE);`
2) `for (int x=0; x<200; x=x+10)`
3) `{ for (int y=x; y < 200; y++)`
4) `} canvas.fillRect(x, y, 1, 1);`
5) `}
6) `}
Quiz: Which Picture is Drawn?

1) `this.setSize(210,201); // Applet window`
2) `java.awt.Graphics` object of the applet window.
3) `canvas.setColor(Color.BLUE);`
4) `for (int a=0; a<=200; a=a+10)`
5) `{}
6) `canvas.drawLine(a, a, a, 200-a);`
7) `}`

![Quiz Images]

Constructor

A **constructor** is a special method that is called when an object of the class is created using the `new` operator.

Constructors must have the same name as the class to which they belong.

A constructor is defined like any other method except that it does not declare a return type (not even void) in the method heading.

**Default Constructor:** A constructor without parameters is called a *default constructor*.

If you do not define any constructor for a class, Java will define one for you. This hidden constructor will be a default constructor - that is, one without parameters.
6.1: Omitting the Default Constructor

What is the textbook saying with the "Gotcha" in 6.1:

When defining a class, such as Pet in section 6.1, suppose we were to omit the definition of the default constructor.

The statement `Pet bob = new Pet();`

would then be invalid since Java only provides a default constructor the class does not provide any constructors.

Classes are often reused again and again, and sooner or later you might want to create a new object by using a constructor without arguments. You should carefully consider a decision to omit a default constructor definition from the classes you write.

Constructors for `javax.swing.JOptionPane`

```java
JOptionPane()  // Creates a JOptionPane with a test message.
JOptionPane(Object message)
JOptionPane(Object message, int messageType)
JOptionPane(Object message, int messageType, int optionType)
JOptionPane(Object message, int messageType, int optionType, Icon icon)
JOptionPane(Object message, int messageType, int optionType, Icon icon, Object[] options)
JOptionPane(Object message, int messageType, int optionType, Icon icon, Object[] options, Object initialValue)
```
Constructors for java.awt.image.BufferedImage

**Constructor Summary**

- **BufferedImage** (ColorModel cm, WritableRaster r, boolean isRasterPremultiplied, Hashtable properties)
  - Constructs a new BufferedImage with a specified ColorModel and Raster.
- **BufferedImage** (int width, int height, int imageType)
  - Constructs a BufferedImage of one of the predefined image types.
- **BufferedImage** (int width, int height, int imageType, IndexColorModel cm)
  - Constructs a BufferedImage of one of the predefined image types: TYPE_BYTE_BINARY or TYPE_BYTE_INDEXED.

Does not have a default constructor.

---

**Quiz: Constructors**

When defining a constructor, what do you specify for the type of the value returned?

- a) A primitive type
- b) A complex type
- c) A class type
- d) **void**
- e) No type at all
Variables: Local, Instance and Static

- **Local Variables**: Variables defined within a method (or even within a block inside a method).

- **Class Fields** (also called a class's data members): Variables defined at the class level (within the class and outside all methods)
  - **Instance Variables**: Each instance of the class has its own memory for the value.
  - **Static Variables** (also called **class variables**): A single value shared by all instances of the class.

- **Note**: Do not confuse the term "class variable" with the notion of a variable of a class type.

---

Quiz: What is the output?

```java
public class FooAndBoo {
    private static int foo = 3;
    private int boo = 5;

    public FooAndBoo() {
        foo++;
        boo++;
        System.out.println(foo + " " + boo);
    }

    public static void main(String[] args) {
        new FooAndBoo();
        new FooAndBoo();
    }
}
```

- a) 4, 6
  - 5, 6
- b) 4, 6
  - 4, 6
- c) 5, 6
  - 5, 6
- d) 4, 5
  - 6, 6
Quiz: What is Zoo?

```java
public class FooAndBoo {
    private static int foo = 3;
    private int boo = 5;

    public static void main(String[] args) {
        FooAndBoo zoo = new FooAndBoo();
    }
}
```

a) zoo is a variable of class type.
b) zoo is a static variable.
c) zoo is a class field.
d) zoo is an instance variable.
e) zoo is both a static and an instance variable.

Quiz: What is the Output?

```java
public class Sneed {
    private static int n = 0;
    private int z = 0;

    public Sneed(int a) {
        z++;
        System.out.print(n*z*a + " ");
        n++;
    }

    public static void main(String[] args) {
        new Sneed(4);
        new Sneed(3);
        new Sneed(5);
        new Sneed(1);
    }
}
```

a) 0 0 0 0  

b) 4 3 5 1  

c) 0 3 10 3  

d) 0 6 30 12  

e) 4 12 45 16