Java Quiz Bowl

A fun review of the Java you should know from CMPT 201

If you don't know the answers - this week is for you to study up!

Part 1:

10 seconds / question

Each question is followed by the answer. First try to answer the question, then check your answer.

What will this code print:

```
int x;
System.out.println(x);
```

Nothing. “x” has been declared but not initialized, which causes a compiler error!

What will this code print:

```
int x;
System.out.println(x);
```

What will this code print:

```
String name;
System.out.println(name);
```

Remember: Strings are NOT primitive types.

Another compiler error! “name” has been declared but not initialized.

What will this code print:

```
String name;
System.out.println(name);
```

What will this code print:

```
String name;
System.out.println(name);
```

What is the value of x after this code:

```
int x = 5 / 2;
```

2

What is the value of x after this code:

```
int x = 5 / 2;
```

Will this code compile?

```
int x = 5;
double y = x;
```

If so, what is the value of y?

If not, fix the code.

Yes
5.0

Will this code compile?

```
int x = 5;
double y = x;
```

If so, what is the value of y?

If not, fix the code.

Will this code compile?

```
double d = 3.14;
int i = d;
```

If so, what is the value of i?

No.

Will this code compile?

```
double d = 3.14;
int i = d;
```

If so, what is the value of i?

What is the value of x that is printed out?

```
public class Query
{
    private int x = 5;

    public void output() {
        int x = 10;

        System.out.println(x);
    }
}
```

What is the value of x that is printed out?

X is equal to 10

```
public class Query
{
    int x = 5;

    public void output() {
        int x = 10;

    System.out.println(x);
    }
}
```

What are the values of i, j, and k after this code is run:

```
int i = 5;
int j = i++;
int k = ++i;
```

i = 7
j = 5
k = 7

What are the values of i, j, and k after this code is run:

```
int i = 5;
int j = i++;
int k = ++i;
```

How many objects are created by this code:

```
String a = new String(“hello”);
String b = a;
```

One object.
How many objects are created by this code:

```
String a = new String(“hello”);
String b = a;
```

Part 2:

30 seconds / question

Write a for-loop that prints out:

```
0 2 4 6 8
```

Possible Answers

```
for (int i=0; i<5; i++)
  System.out.print(2*i + " ");
```

```
for (int i=0; i<9; i+=2)
  System.out.print(i + " ");
```

```
for (int i=0; i<=8; i=i+2)
  System.out.print(i + " ");
```

What variables are accessible from inside method “one()”?

```
public class Question {
    public int a = 1;
    private int b = 2;
    public static int c = 3;
    private static int d = 4;

    public void one (int e){
      int f = 5;
      }
    private void two (int g){
      int h = 10;
    }
    public static void three (int i){
      int j = 5;
      }
    private static void four (int m){
      int n = 10;
    }
}
```

What variables are accessible from inside method “one()”?

```
public class Question {
    public int a = 1;
    private int b = 2;
    public static int c = 3;
    private static int d = 4;
    public void one (int e){
      int f = 5;
      }
    private void two (int g){
      int h = 10;
    }
    public static void three (int i){
      int j = 5;
      }
    private static void four (int m){
      int n = 10;
    }
}
```

What variables are accessible from inside method “two()”?

```
public class Question {
    public int a = 1;
    private int b = 2;
    public static int c = 3;
    private static int d = 4;
    public void one (int e){
      int f = 5;
      }
    private void two (int g){
      int h = 10;
    }
    public static void three (int i){
      int j = 5;
      }
    private static void four (int m){
      int n = 10;
    }
}
```

What variables are accessible from inside method “two()”?

```
public class Question {
    public int a = 1;
    private int b = 2;
    public static int c = 3;
    private static int d = 4;
    public void one (int e){
       int f = 5;
      }
    private void two (int g){
       int h = 10;
    }
    public static void three (int i){
       int j = 5;
      }
    private static void four (int m){
       int n = 10;
    }
}
```

What variables are accessible from inside method “three()”?

```
public class Question {
    public int a = 1;
    private int b = 2;
    public static int c = 3;
    private static int d = 4;
    public void one (int e){
      int f = 5;
      }
    private void two (int g){
      int h = 10;
    }
    public static void three (int i){
      int j = 5;
      }
    private static void four (int m){
      int n = 10;
    }
}
```

What variables are accessible from inside method “three()”?

```
public class Question {
    public int a = 1;
    private int b = 2;
    public static int c = 3;
    private static int d = 4;
    public void one (int e){
       int f = 5;
      }
    private void two (int g){
       int h = 10;
    }
    public static void three (int i){
       int j = 5;
      }
    private static void four (int m){
       int n = 10;
    }
}
```

What variables are accessible from inside method “four()”?

```
public class Question {
    public int a = 1;
    private int b = 2;
    public static int c = 3;
    private static int d = 4;
    public void one (int e){
      int f = 5;
      }
    private void two (int g){
      int h = 10;
    }
    public static void three (int i){
      int j = 5;
      }
    private static void four (int m){
      int n = 10;
    }
}
```

What variables are accessible from inside method “four()”?

```
public class Question {
    public int a = 1;
    private int b = 2;
    public static int c = 3;
    private static int d = 4;
    public void one (int e){
       int f = 5;
      }
    private void two (int g){
       int h = 10;
    }
    public static void three (int i){
       int j = 5;
      }
    private static void four (int m){
       int n = 10;
    }
}
```

Fill in the code to print all the elements of an array to the screen:

```
public void print (int []array){



}
```

```
public print (int []array){

   for (int i=0; i<array.length; i++)
     System.out.println(array[i]);

}
```

OR

```
 for (int x : array)
    System.out.println(x);
```

Part 3:

1 minute / question

Write a Circle class with one instance variable (data field):

radius (double)

No methods necessary.

```
public class Circle {
    private double radius;


}
```

Given the current code, what is radius' value?

```
public class Circle {
    private double radius;


}
```

radius = 0.0

Instance variables of primitive types are initialized to 0.

Given the current code, what is radius' value?

```
public class Circle {
   private double radius;


}
```

Given the following code, what is name's value?

```
public class Person{
    private String name;


}
```

name = null

Instance variables of primitive types are initialized to null.

Given the current code, what is name's value?

```
public class Person{
    private String name;


}
```

Add a mutator (*setter* method) to your Circle class that sets the radius to a specified value.

```
public class Circle {
    private double radius;


    public void setRadius(double radius) {
        this.radius = radius;
    }

}
```

Add an accessor (*getter* method) to your Circle class that gets the radius.

```
public class Circle {
    private double radius;


    public void setRadius(double r) {
        radius = r;
    }


    public double getRadius() {
        return radius;
    }
}
```

Now add a constructor that takes a double as a parameter.

Set the radius to the parameter.

```
public class Circle {
    private double radius;

    public Circle(double radius) {
        this.setRadius(radius);
        // or
        this.radius = radius;
    }

    public void setRadius(double radius) {
        this.radius = radius;
    }
    public double getRadius() {
        return radius;
    }
}
```

Now add a default constructor to your Circle class.

Set the radius to 1.

```
public class Circle {
    private double radius;

    public Circle() {
        this(1);
        // or
        radius = 1;
        // or
        setRadius(1);
    }
    public Circle(double radius) {
        this.radius = radius;
    }

    public void setRadius(double radius) {
        this.radius = radius;
    }
    public double getRadius() {
        return radius;
    }
}
```

Write a static method that calculates and returns the area of a circle, taking the radius as a parameter.

```
public class Circle {
    private double radius;

    public void setRadius(double radius) {
        this.radius = radius;
    }
    public double getRadius() {
        return radius;
    }

public static double getArea(double radius)
{
        return Math.PI*radius*radius;
    }

}
```

Write a line of code that creates a Circle object. You can assume this is being written in a main method.

```
public static void main(String[] args) {
        Circle c1 = new Circle();
        Circle c2 = new Circle(10);
    }
}
```

Will this code compile and run?

```
public static void
main(String[] args)
{
      Circle c1;
      c1 = new Circle();
      c1 = new Circle(10);

}
```

Yes!
You can only *declare* a variable once, but you can set it to new objects multiple times.

Is this code valid?

```
public static void
main(String[] args)
{
        Circle c1;
        c1 = new Circle();
        c1 = new Circle(10);

}
```

Write a new class ColoredCircle, that is a child of the Circle class.

It should have one additional instance variable (type String) that represents the color of the ColoredCircle

No methods or constructors yet.

```
public class ColoredCircle extends Circle
{
   private String color;
}
```

Add a default constructor that sets the radius to 1 and the color of the circle to black.

```
public class ColoredCircle extends Circle
{
   private String color;

   public ColoredCircle() {
     super();
     color = “black”;
   }
}
```

Is this code legal?

```
ColoredCircle c = new ColoredCircle();
c.setRadius(100);
```

Yes! ColoredCircle inherits all of the methods in the Circle class!

```
ColoredCircle c = newColoredCircle();
c.setRadius(100);
```

Which of the following code segments are legal?

```
Circle c1 = new
ColoredCircle();
```

```
ColoredCircle c2 =
new Circle();
```

LEGAL:

```
Circle c1 = new ColoredCircle();
```

NOT LEGAL:

```
ColoredCircle c2 = new Circle();
```

If the ColoredCircle class had a setColor() method, is this code legal?

```
Circle c = new
ColoredCircle();

c.setColor("red");
```

No – Circle objects do not have a setColor() method.

```
Circle c = new
ColoredCircle();

c.setColor("red");
```

Modify the 2nd line of code to make this setColor call legal:

```
Circle c = new
ColoredCircle();

c.setColor("red");
```

We would need to typecast c.

```
Circle c = new
ColoredCircle();

((ColoredCircle)c).set
Color("red");
```

What if both classes had their own (different) toString() methods? Which would run here:

```
Circle c = new
ColoredCircle();

c.toString();
```

The ColoredCircle toString() method would execute:

```
Circle c = new
ColoredCircle();

c.toString();
```