Quiz: Loops and Lines

1) ```import javax.swing.JApplet;```  
2) ```import java.awt.Color;```  
3) ```import java.awt.Graphics;```  
4) ```public class LoopsAndLines extends JApplet```  
5) ``` { public void paint(Graphics canvas)```  
6) ``` { this.setSize(201,201);```  
7) ``` canvas.setColor(Color.BLUE);```  
8) ``` for (int i=0; i<=200; i=i+10)```  
9) ``` { canvas.drawLine(?); // (x₁, y₁, x₂, y₂) ```  
10) ``` } ```  
11) ``` } ```

Which arguments in line 9 would draw this?

a) (0, i, i, 200)  
b) (0, i, 200, 200-i)  
c) (0, i, 200, i-200)  
d) (i, i-200, i, 200)  
e) (i, i, i-200, 200)
Loops and Lines

8) for (int i=0; i<=200; i=i+10)
9) { canvas.drawLine(0, i, i, 200); // $x_1, y_1, x_2, y_2$
10) }

$(x_1, y_1)$ to $(x_2, y_2)$
$(0,0)$ to $(0,200)$
$(0,10)$ to $(10,200)$
$(0,20)$ to $(20,200)$
$(0,30)$ to $(30,200)$

java.util.Random

import java.util.Random;
public class ChaosGame
{
    public static void main(String[] args)
    {
        Random rand = new Random();
        System.out.println(rand.nextDouble());
        System.out.println(rand.nextDouble());
        System.out.println(rand.nextDouble());
        System.out.println(rand.nextDouble());
    }
}

$0.0 \leq \text{rand.nextDouble()} < 1.0$
import java.util.Random;
public class ChaosGame {
    public static void main(String[] args) {
        Random rand = new Random();

        // 0.0 ≤ rand.nextDouble() < 1.0
        System.out.println(rand.nextDouble());
        System.out.println(rand.nextDouble());
        System.out.println(rand.nextDouble());
        System.out.println(rand.nextDouble());
    }
}

Uniformly distributed, pseudo-random numbers: [0.0, 1.0).
Probability r < 0.5 is 50%.

Loop: while(true)
import java.util.Random;
public class ChaosGame {
    public static void main(String[] args) {
        Random rand = new Random();

        while(true) {  // No semicolon!!!
            double r = rand.nextDouble();
            System.out.println(r);
        }
    }
}
import java.util.Random;
public class ChaosGame
{
    public static void main(String[] args)
    {
        Random rand = new Random();

        while(true)
        { // 0.0 ≤ r < 1.0
            double r=rand.nextDouble();

            // 50.0 ≤ r < 75.0
            r = r*25.0 + 50.0;
            System.out.println(r);
        }
    }
}

new Random(long seed);

public static void main(String[] args)
{
    Random rand = new Random(3141592654L);
    while(true)
    { System.out.println(rand.nextDouble());
    }
}
java.util.Random: `nextInt(int n)`

1. Random rand = new Random();
2.
3. for (int i=0; i<10; i++)
4. { double r = rand.nextInt(6); // 1 ≤ r < 6
5. System.out.println(r);
6. }

`rand.nextInt(6)`
Uniformly distributed, random int: [0 through 5].
The probability is 1/6 that \( r == 0 \).
The probability is 1/6 that \( r == 1 \).
The probability is 1/6 that \( r == 2 \).
...

Quiz: Which Line is *Incorrectly* Indented?

```java
import java.util.Random;

public class ChaosGame {

  public static void main(String[] args) {
    Random rand = new Random();
    while(true) {
      int r1 = rand.nextInt(6);
      int r2 = rand.nextInt(6);
      System.out.println(r1 + r2);
    }
  }
}
```
import java.awt.Color;
import java.awt.Graphics;
import java.util.Random;

public class DiceHistogram {
    private Picture myPic;
    private Graphics canvas;

    private static final int HIST_HEIGHT = 500;
    private static final int BAR_WIDTH = 90;
    private static final int BAR_GAP = 10;
    private static final int BAR_COUNT = 6;
    private static final int TOTAL_ROLES = 1000000;

    private static final int HIST_WIDTH = (BAR_WIDTH + BAR_GAP) * BAR_COUNT;

    private static final Color DARK_RED = new Color(200,0,0);

    public DiceHistogram() {
        myPic = new Picture(HIST_WIDTH, HIST_HEIGHT);
        canvas = myPic.getOffScreenGraphics();
        myPic.setTitle("Histogram of a 6-sided die");
    }
}
DiceHistogram: main: Part 1 of 3

1) `public static void main(String[] args)`
2) {
3)    Random rand = new Random();
4)    DiceHistogram hist = new DiceHistogram();
5)    
6)    int count1 = 0;
7)    int count2 = 0;
8)    int count3 = 0;
9)    int count4 = 0;
10)   int count5 = 0;
11)   int count6 = 0;
12)   
13)   Create and initialize six Histogram "bins"
14)   Note: this would be better done by using an array:
15)      int[] count = new int[6];

DiceHistogram: main: Part 2 of 3

16)   for (int i=0; i<TOTAL_ROLES; i++)
17)   {
18)       int r = rand.nextInt(6) + 1;
19)       
20)       if (r==1) count1++;
21)       else if (r==2) count2++;
22)       else if (r==3) count3++;
23)       else if (r==4) count4++;
24)       else if (r==5) count5++;
25)       else count6++;
26)   }
This program is a simulation of 1 million rolls of a six-sided die. Data from one experiment.

drawBar(num=1, count=167003)
drawBar(num=2, count=167326)
drawBar(num=3, count=166837)
drawBar(num=4, count=166602)
drawBar(num=5, count=166237)
drawBar(num=6, count=165995)
Quiz: Which is the Inner `for` statement?

1) `for (int a=1; a<=7; a++)`
2) `{ //for ?????
3) { System.out.print(b+" ");
4) }
5) System.out.println();
6) }

a) `for (int b=a; b < a*2; b=b*a)`
b) `for (int b=a; b < a*2; b=a*a)`
c) `for (int b=a; b <= a*a; b=b+a)`
d) `for (int b=a; b < b+a; b++)`
e) `for (int b=a; b < b*a; b++)`
What Does This Draw?

```java
public static void main(String[] args) {
    final int WIDTH = 600;
    final int HEIGHT = 600;
    Random rand = new Random();
    Picture myPic = new Picture(WIDTH, HEIGHT);
    for (int n=0; n<100000; n++) {
        int x = rand.nextInt(WIDTH);
        int y = rand.nextInt(HEIGHT);
        myPic.setRGB(x,y,0,0,255);
        myPic.repaint();
    }
}
```

The Bell

```java
for (int n=0; n<100000; n++) {
    int x = ( rand.nextInt(WIDTH) +
              rand.nextInt(WIDTH) +
              rand.nextInt(WIDTH) )/3;
    int y = ( rand.nextInt(HEIGHT) +
              rand.nextInt(HEIGHT) +
              rand.nextInt(HEIGHT) )/3;
    myPic.setRGB(x,y,0,0,255);
    myPic.repaint();
}
```
Simulation of a "Winning" Strategy for Roulette

When a system is not well enough understood or too complicated to solve analytically, it may be possible to make accurate predictions about the system by creating a model of the system where important, but not easily or accurately measured values are replaced by random variables.

Generally, a simulation needs to be "run" many times with the random variables taking on different values. The results of these "runs" are then statistically analyzed.
The Game of Roulette

- In Roulette, a croupier spins a wheel in one direction, then spins a ball in the opposite direction around a tilted circular surface running around the circumference of the wheel.
- The ball eventually comes to rest in one of 38 colored and numbered pockets on the wheel.
- Players bet on the outcome being a particular number, color (black, red or green), groups of numbers, etc.
- Payout for each type of bet is based on its probability.
- Of the 38 colored and numbered pockets on the wheel, 18 are black, 18 are red and 2 are green.
- The payout for betting on black or on red is 1 to 1.

Why the House Wins

If a player bets on black or on red, then the player will, on average, lose \((18+2)/38\) times.

If 1 million people each make a $10 bet on either black or red, then, on average, the house will win:

\[
10.00 \times \left( \frac{1,000,000 \times 20}{38} - \frac{1,000,000 \times 18}{38} \right)
\]

\[= 10.00 \times (52,631)\]

\[= \text{Half a Million Dollars}\]
Winning Heuristic?

1. User Inputs:
   - cash: Money the player starts with.
   - goal: Money player wants before quitting.
2. Bet $10.00 on black.
3. Whenever black is rolled:
   - Collect the 1:1 payout
   - If you have reached your goal, go party 😊.
   - If not, bet again: ten dollars on black (step 2).
4. Whenever red or green is rolled, either
   - Bet again on black, double the previous bet, or
   - Go home if you do not have enough money to cover the doubled bet (and cry) 😞.

Lose 6, Win 4: Yet Still Makes a Profit

<table>
<thead>
<tr>
<th>Start Balance: $1000.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bet</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
</tbody>
</table>
Algorithm for "Winning" Strategy

1) Get user input: cash, goal, startBet = $10.00.
2) bet = startBet
3) If bet > cash: exit
4) Pick a random number, r, (0 through 37)
   If r == 0, 37 (00) or odd (remainder of r ÷ 2 == 1)
   You Lose:
   Subtract bet from cash, and double bet

   else You Win!!
   Add bet to cash and set bet=startBet
   if cash ≥ goal: exit
5) Go back to step 3

Winning at Roulette:

Extra Credit (up to 20 points)

"Write a Java program that simulates the given Roulette strategy and run it many times."

"Decide, with reason, whether the strategy works. If yes, how much money must you start with to expect to win a particular goal?"

"Make an original modification to the strategy."

"Code your modified strategy in Java."

"Demonstrate, through analysis of experimental simulation results, that your modification is a statistically significant improvement (more money with less risk in less time)."

"Present your results in a 5 minute class demo."