Java Control Statements

Liv

2021-02-23

1 Questions and Examples

1.1 The most basic control flow statement supported by the Java programming language is the _____ statement.

The if-then statement.

1.2 The _____ statement allows for any number of possible execution paths.

The switch statement does.

1.3 The _____ statement is similar to the while statement, but evaluates it's expression at the _____ of the loop.

The do-while evaluates it's expression at the bottom of the loop.

1.4 How do you write an infinite loop using the for statement?

```
for (int i = 0; i > 0; i++) {
    System.out.println("This is an infinite loop");
}
for ( ; ; ) {
    System.out.println("This is also an infinite loop");
}
```

1.5 How do you write an infinite loop using the while statement?

```
while (true) {
    System.out.println("This is an infinite loop");
}
```

1.6 Consider the following code snippet.

1.6.1 What output do you think the code will produce if aNumber is 3?

```
int aNumber = 3;

if (aNumber >= 0) { // This block is entered because 3 is >= 0
    if (aNumber == 0) {
        System.out.println("first string"); // does not print
    } else {
        System.out.println("second string"); // prints
    }
}
System.out.println("third string"); // prints
```

1.6.2 Write a test program containing the previous code snippet, make aNumber 3. What is the output of the program? Is it what you predicted? Explain why the output is what it is; in other words, what is the control flow for the code snippet?

```
public class NumberTest {
    public static void main(String[] args) {
        int aNumber = 3;

        if (aNumber >= 0) { // This block is entered because 3 is >= 0
            if (aNumber == 0) {
                System.out.println("first string"); // does not print
            } else {
                System.out.println("second string"); // prints
            }
        }
        System.out.println("third string"); // prints
    }
}
```

The output of the program is as follows, as predicted by the code comments. I explained the control flow in the comments, I feel it is clear enough.

```
second string
third string
```

1.6.3 Using only spaces and line breaks, reformat the code snippet to make the control flow easier to understand.

I feel this is as clear as you will get with only spaces and linebreaks.

```
if (aNumber >= 0)
    if (aNumber == 0) System.out.println("first string");
    else System.out.println("second string");

System.out.println("third string");
```

1.6.4 Use braces, and , to further clarify the code.

I have already done this in examples 1.6.1 and 1.6.2.

1.7 Many centuries ago in India a very smart man is said to have invented the game of chess.

I feel like I'm too tired to do this but oh well. I'm also doing this without an IDE or testing so I will be surprised if it works.

```
public class RiceFarmer {
    public static void main(String[] args) {
        int rice = 1;

        for (i = 0; i < 64; i++) {
            rice *= 2;
        }

        System.out.println("Total rice grains are: " + rice);
    }
}
```

1.8 Write a program that simulates rolling a pair of dice.

This is dumb.

```
public class DiceRoller {
    public static void main(String[] args) {
        int roll1 = (int) (Math.random()*6) + 1;
        int roll2 = (int) (Math.random()*6) + 1;

        System.out.println("The first die comes up " + roll1);
        System.out.println("The second die comes up " + roll2);
        System.out.println("Your total roll is " + (roll1 + roll2));
    }
}
```

1.9 How many times do you have to roll a pair of dice before they come up snake eyes?

```
public class DiceRoller {
    public static void main(String[] args) {
        int attempts = 0;
        Boolean snakeEyes = false;

        do {
            int roll1 = (int) (Math.random()*6) + 1;
            int roll2 = (int) (Math.random()*6) + 1;

            attempts++;
            if (roll1 == 0 && roll2 == 0) {
                snakeEyes = true;
            }
        } while (!snakeEyes)

        System.out.println("Found snake eyes after " + attempts + "attempts");
    }
}
```

1.10 Write a program to extract each digit from an int, in the reverse order.

This could probably be more efficient if I used an IDE honestly, but I think it works.

```
public class DigitExtractor {
    public static void main(String[] args) {
        int in = 1542;
        String reversed = "";

        for (int i = 0; i < String.valueOf(in).length(); i++) {
            reversed += String.valueOf((int) (in / (Math.pow(10, i))) % 10);
        }

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

1.11 Write a program called TimeTable to produce the multiplication table of 1 to 9 as shown using nested for-loops.

```java
public class TimeTable {
    public static void main(String[] args) {
        int[][] arr = new int[10][10];
        String[] tmp = new String[10];

        for (int i = 1; i < 10; i++) {
            for (int j = 1; j < 10; j++) {
                tmp[j] = (i * j) + "";
            }
            // I am too lazy to fix the fact that tmp[0] is always null
            System.out.println(String.join(" ", tmp).substring(5));
        }
    }
}
```

1.12 Modify the program to print the multiplication table of 1 to 12.

Just change the 10 to a 13.