while Loops

Introduction

In programming, a loop is a control structure that allows a set of instructions to be repeated multiple times. Like conditional statements, a Boolean condition is needed. A loop uses a Boolean condition to control its iterations. This enables efficient execution of repetitive tasks without the need to write the same code multiple times - write once, repeat many times. Loops are a fundamental concept in most programming languages and are commonly used to iterate over data collections, perform calculations, or repeat code blocks. The Boolean condition is not checked in the loop's body; instead, it is tested before each iteration of the body (in a **pre-test** loop) or after each iteration of the loop body (in a **post-test** loop).

In Java, there are three primary types of loops: `for`, `while`, and `do-while`. The `for` and `while` loops are pre-test loops, and the `do-while` loop is a post-test loop. We will study the `for` and `while` loops.

`while` Loop: A `while` loop can be used when the number of iterations is uncertain, and it will continue to execute a block of code as long as its Boolean condition remains `true`. This condition is checked before each iteration, making it a **pre-test** loop. If the condition is `true`, the body of the loop will execute. If the condition is `false`, the loop will terminate, and the next line of code after the loop will execute.

Format of a `while` loop:

```
/* while loop with a single statement body - no braces required */
while (condition)
     statement;

/* while loop with a multiple statements body - braces required */
while (condition)
{
    statement1;
    . . .
    statementN;
}
```

Like the `if` and `else`, the `while` loop only executes one statement. You must use braces `{}` to group multiple statements in the body of a `while` loop.

HW: Complete the even numbered problems (#2, 4, 6, 8, 10, 12, 14)

Explain what happens if the condition changes as follows.

```
count = 0;
powOf2 = 1;
while (powOf2 != 100)
{
   powOf2 = powOf2 * 2;
   count++;
   System.out.print(powOf2  + " " );
}
System.out.println("\nThere are " + count +
                     " powers of 2 less than 100.");
```

What output is produced by the following code segments? Assume all variables have been properly declared.

	Code Segment	Output

1.	`k = 1;`
	`while(k <= 10)`
	`{`
	`   System.out.println(k + " ");`
	`   k++;`
	`}`	

2.	`a = 1;`
	`while(17 % a != 5)`
	`{`
	`   System.out.println(a + " " + 17 % a);`
	`   a++;`
	`}`	

3.	`a = 2; b = 50;`
	`while(a < b)`
	`{`
	`   a *= 3;`
	`}`
	`System.out.println(a + " " + b);`	

4.	`count = sum = 0;`
	`while(count < 5)`
	`{`
	`   count++;`
	`   sum += count;`
	`   System.out.println("Partial sum = " + sum);`
	`}`
	`System.out.println("The count is " + count);`
	`System.out.println("The sum is " + sum);`	

Complete the even numbered problems (#2, 4, 6, 8, 10, 12, 14)

	Code Segment	Output

5.	`x = 3.0;  y = 2.0;`
	`while(x * y < 100)`
	`{`
	   `x *= y;`
	`}`
	`System.out.println(x + " " + y);`	

6.	`count = 1; y = 100;`
	`while(count < 100)`
	`{`
	   `y--;`
	   `count++;`
	`}`
	`System.out.println("y = " + y + " and count = "`
	                    `+ count);`	

7.	`num = 1;`
	`while(num < 10)`
	`{`
	   `System.out.print(num + " ");`
	   `num += 2;`
	`}`	

8.	`num = 1;`
	`while(num < 10)`
	`{`
	   `num += 2;`
	   `System.out.print(num + " ");`
	`}`	

9.	`num = 5;`
	`while(num < 5)`
	`{`
	   `num += 2;`
	`}`
	`System.out.println(num);`	

Complete the even numbered problems (#2, 4, 6, 8, 10, 12, 14)

For the following exercises, indicate which loops are infinite. If infinite, explain why.

	Code Segment	**Infinite?**

10.	`j = 1;`
	`while(j < 10)`
	`   System.out.print(j + " ");`
	`   j++;`	

11.	`a = 2;`
	`while(a < 100)`
	`{`
	`   System.out.println(a);`
	`   a *= 2;`
	`}`	

12.	`a = 2;`
	`while(a != 100)`
	`{`
	`   System.out.println(a);`
	`   a *= 2;`
	`}`	

13.	`num = 1;`
	`while(num < 10)`
	`{`
	`   num += 2;`
	`   System.out.print(num + " ");`
	`}`	

14.	`b = 15;`
	`while(b / 3  >  3)`
	`{`
	`   System.out.println(b + " " + b/3);`
	`   b--;`
	`}`	