Welcome,

PROGRAMMERS

Loops (Repetition Structure)

Loops allows the **execution of a block of code repeatedly** as long as a specified **condition is true**.

1. Entry Control Loop
   - while loop
   - for loop

2. Exit Control Loop
   - do while loop

Let’s see **do while loop** in detail…

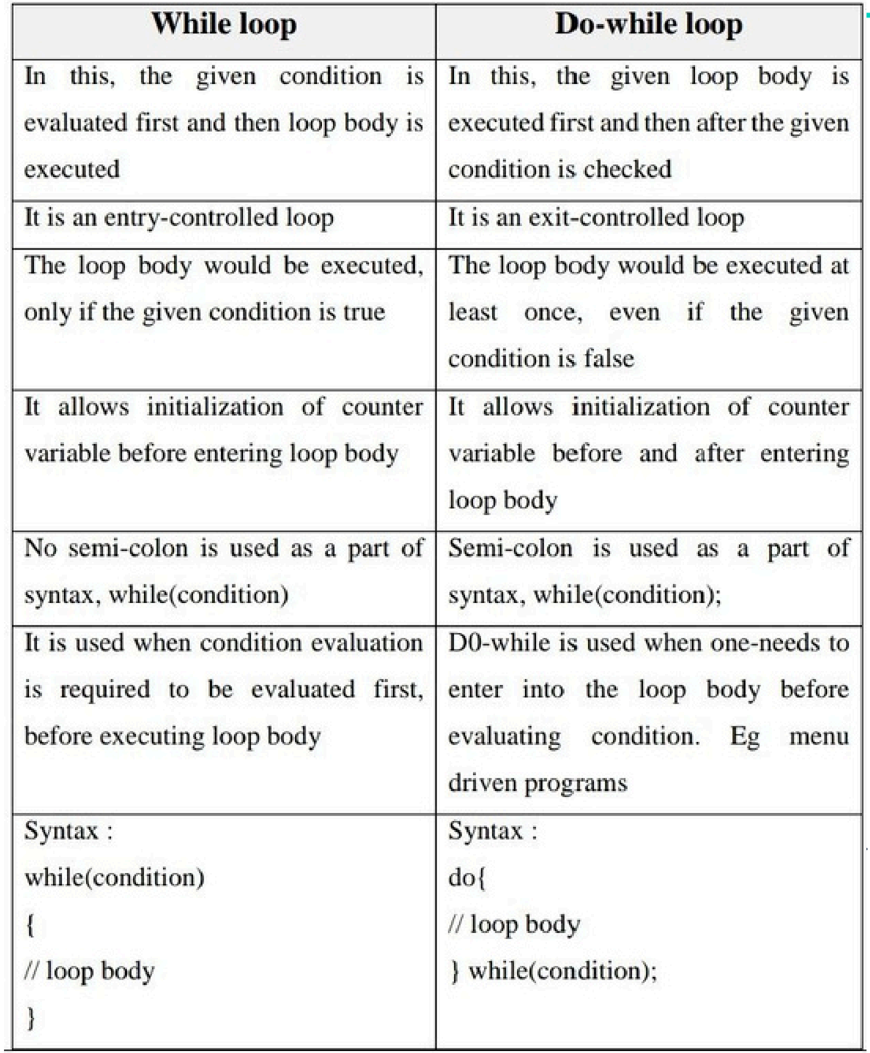
Syntax of do while loop

The do-while loop is a control flow structure that repeats a block of code while a given condition is true.

The unique feature of the do-while loop is that **the condition is checked after the execution of the loop block**, meaning that **the block of code is guaranteed to run at least once**.

```
Initialization
do
{
    // Code to be executed
 // Increment / Decrement
} while (Condition);
```

Difference between while & do while loop…



While loop	Do-while loop
In this, the given condition is evaluated first and then loop body is executed	In this, the given loop body is executed first and then after the given condition is checked
It is an entry-controlled loop	It is an exit-controlled loop
The loop body would be executed, only if the given condition is true	The loop body would be executed at least once, even if the given condition is false
It allows initialization of counter variable before entering loop body	It allows initialization of counter variable before and after entering loop body
No semi-colon is used as a part of syntax, while(condition)	Semi-colon is used as a part of syntax, while(condition);
It is used when condition evaluation is required to be evaluated first, before executing loop body	D0-while is used when one-needs to enter into the loop body before evaluating condition. Eg menu driven programs

Syntax :
while(condition)
{
// loop body
}
	Syntax :
	do{
	// loop body
	} while(condition);

Let's see **for loop** in detail…

Syntax of for loop

The for loop is a control flow statement that **allows a certain block of code to be executed repeatedly as long as a specified condition is true**.

The for loop is often used when the number of iterations is known

before entering the loop.

```
for (initialization; condition; increment/decrement) {
}
    // code to be executed is the given condition is true
```

Let’s see **for loop** written **approaches** in detail…

01

```
for (initialization; condition; increment/decrement)
{
    // code to be executed is the given condition is true
}
```

02

```
initialization
for ( ; condition; increment/decrement)
{
    // code to be executed is the given condition is true
}
```

03

```
initialization
for ( ; condition; )
{
    // code to be executed is the given condition is true
    increment/decrement
}
```

04

```
initialization
for ( ; ; )
{
    // code to be executed is the given condition is true
    increment/decrement
}
```

Note: This is also known as **infinite loop.**

Language

Let’s start now…