While Loops with Natural Language

A while loop is a structure within ROBOTC which allows a section of code to be repeated as long as a certain condition remains true.

There are three main parts to every while loop.

Part 1. The keyword “while”.

```c
while (condition) {
    // repeated-commands
}
```

Every while loop begins with the keyword “while”.

Part 2. The condition.

```c
while (condition) {
    // repeated-commands
}
```

The condition is true as long as 1 is equal to 1, which is always.

Part 3. The commands to be repeated, or “looped”.

```c
while (condition) {
    // repeated-commands
}
```

Commands placed between the curly braces will repeat while the (condition) is true when the program checks at the beginning of each pass through the loop.

Below is an example of a program using an infinite While Loop.

```c
task main() {
    while (1 == 1) {
        startMotor(port2, 63);
        wait(5.0);
        startMotor(port2, -63);
        wait(5.0);
    }
}
```

The condition is true as long as 1 is equal to 1, which is always.

While the condition is true, the port2 motor will turn forward for 5 seconds, then in reverse for 5 seconds.

Result: The port2 motor will turn back and forth, forever.
### While Loops with Natural Language

Below is an example of a program using a **counter-controlled** While Loop.

```c
task main()
{
    int count = 0;
    while (count < 4)
    {
        startMotor(port2, 63);
        wait(5.0);
        startMotor(port2, -63);
        wait(5.0);
        count = count + 1;
    }
}
```

- Creates an integer variable named “count” and gives it an initial value of 0.
- Checks if count is “less than” 4.
- Adds 1 to count every time the loop runs.
- Result: The loop repeats 4 times, causing the port2 motor to turn back and forth, four times.

Below is an example of a program using a **sensor-controlled** While Loop.

```c
#pragma config(Sensor, dgtl1, Estop, sensorTouch)
#pragma config(Sensor, dgtl2, controlBtn, sensorTouch)
#pragma config(Sensor, dgtl3, LED, sensorDigitalOut)

task main()
{
    while (SensorValue[Estop] == 0)
    {
        if (SensorValue[controlBtn] == 1)
        {
            turnLEDOn(LED);
        }
        else
        {
            turnLEDOff(LED);
        }
    }
}
```

- Checks if the “Estop” touch sensor is equal to 0 (unpressed).
- If the “controlBtn” is pressed, turn the LED on; if it’s not, turn the LED off.
- Result: The loop repeats continuously, allowing the LED to be turned on while the “controlBtn” is pressed, and off while “controlBtn” is released. The loop will stop as soon as the “Estop” touch sensor is pressed.