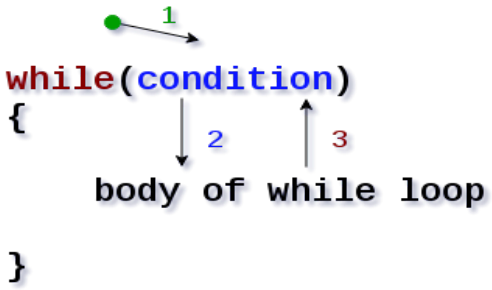
While Loop

The **while** loop is one of the three main forms of looping you will come across. Take note that the condition to check whether or not to execute the body of the loop happens at the very beginning. Remember, if a variable is used in the condition, it has to already be declared and defined. For a general runthrough of a **while** loop:



Step 1: Check condition. True = proceed / False = exit
Step 2: Execute body of the while loop
Step 3: Go to step 1

An example of a **while** loop that terminates due to a condition:

```
int count = 5;
while(count >= 0) {
    cout << count << ' '; //Displays 5 4 3 2 1 0
    --count; //Decrement count so when it is -1 the loop quits
}
```

The condition does not always have to contain a variable. If you would like to run a loop until you **break** from it in the body, you can write the condition as **while(1)** or **while(true)**. The **break** keyword means to exit a loop early due to some other condition in the body of the loop. Here is an example of a loop that runs until a **break** statement:

```
cout << "Enter grades 0-100. Enter negative number to stop\n";
int input, count = 0, grades = 0;
while(1) {
    cin >> input;
    if(input < 0)
        break;
    grades += input;
    count++;
}
double average = static_cast<double>(grades)/count;
cout << "The average of the grades is : " << average << endl;
```

This program's loop runs until the user enters a number less than zero. Once the negative number has been entered, the **break** statement will be executed. This is useful, because you do not want the negative number to be added to the sum of the grades and you do not want the count incremented by one as there is no such thing as a negative grade. After the loop has been terminated, the average grade is calculated and the average is then displayed.