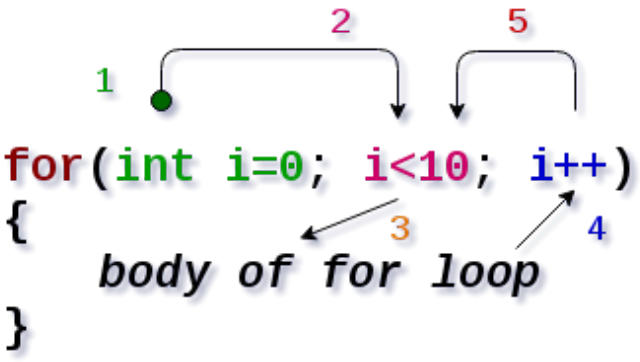
# For Loop

The **for** loop is one of the three main forms of looping you will come across. It takes three arguments. For example, The first section (**green**) is declaring and initializing a local variable **i** to act as a counter. This local variable is also used to keep track of elements in arrays by acting as the element's index. The second argument is the sentinel value (**magenta**), which means the value where the loop will end. In this case, the loop ends when **i** is **>= 10** and only runs so long as this statement evaluates to true. The last argument (**blue**) is popular for incrementing/decrementing the variable from the first argument. Other statements, such as a cout statement, are allowed here. For a general runthrough of a **for** loop:




**Step 1: Declare/Initialize a variable (usually i set to 0).**
**Step 2: Check condition. True = proceed/False = exit**
**Step 3: Execute the body of the for loop.**
**Step 4: Execute last parameter of for loop (increment i by one).**
**Step 5: Go to step 2**

The variable in the first part (**green**) can be a variable that has already been declared. Remember, if you initialize and declare a variable, it is local only to the **for** loop and is invalid elsewhere. If you use an existing variable, be cautious that the variable will contain a different number than when it did going into the loop. It is recommended to use local a variable in a **for** loop, namely **int i = 0**. Here is an example filling an array as well as using a non-local variable:

```
int array[5];
int index; //Potential logic error using broader scope variable
for(index = 0; index < 5; ++index)
    array[index] = index + 1;
```

We declared an array of size 5 and an integer. The loop sets **index** to zero and runs until **index** is equal to five. The first run can be looked at as **array[0] = 0 + 1**, the second run can be looked at as **array[1] = 1 + 1**, and so forth. At the end, array contains the numbers 1,2,3,4,5, and **index** now contains 5 and can be used after the loop.

If you put a semicolon immediately after the closing parenthesis of a **for** loop with no body, the loop will just run through the parameters. Example: **for(int i = 0; i < 10000; ++i);**

**for( ; ; )** is the equivalent of an infinite loop.