Programming Fundamentals- I

Rao Muhammad Umer
Lecturer,
Web: raoumer.github.io
Department of Computer Science & IT,
The University of Lahore.

Loops

For Loop

Printing the counting from 1 to 10

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for (i = 1; i <= 10; i++)
        cout << "i= \n", i;
    getch();
    return 0;
}
```

The for Statement

- The most important looping structure in C/C++.
- Generic Form:

  for (initial; condition; increment)

  statement

- initial, condition, and increment are C++ expressions.
- For loops are executed as follows:
  1. Initial is evaluated. Usually an assignment statement.
  2. Condition is evaluated. Usually a relational expression.
  3. If condition is false (i.e. 0), fall out of the loop (go to step 6.)
  4. If condition is true (i.e. non zero), execute statement
  5. Execute increment and go back to step 2.
  6. Next statement

The for Statement

For statement examples

```
#include <iostream>
using namespace std;
int main () {
    int count;
    /* 1. simple counted for loop */
    for (count =1; count <=20;
    count++)
    cout << "\n", count;
    /* 2. counting backwards */
    for (count = 100; count >0;
    count--)
    cout << "count= ", count;
    /* 3. for loop counting by 5's */
    for (count=0; count<1000;
    count += 5)
    cout << "count= ", count;
    /* 4. initialization outside of
    loop */
    count = 1;
    for ( ; count < 1000; count++)
    cout << " \n", count;
    getch();
    return 0;
}
For ( ; ; )
{
}
```

While Loop

Printing the counting from 1 to 10

```
#include <iostream>
using namespace std;
int main()
{
	int i= 1;
	while (i<= 10)
		{
		cout << "i=\n", i;
		i++;
		}
	getch();
	return 0;
}
```

The while Statement

- Generic Form

```
while (condition)
    statement
```

- Executes as expected:
  1. condition is evaluated
  2. If condition is false (i.e. 0), loop is exited (go to step 5)
  3. If condition is true (i.e. nonzero), statement is executed
  4. Go to step 1
  5. Next statement
- Note:

```
for (exp1; exp2; exp3) stmt;
```

is equivalent to

```
exp1;
while(exp2) { stmt; exp3; }
```

Do while Loop

Counting from 1 to 10

```
#include <iostream>
using namespace std;
int main()
{
    int i= 1;
    do
        {
            cout << "i= \n", i;
            i++;
        } while (i<= 10) ; // can also be written as while ( ++i<= 10)
    getch();
    return 0;
}
```

The do while Loop

- The do/while repetition structure
  - Similar to the while structure
  - Condition for repetition tested after the body of the loop is performed
    - All actions are performed at least once
  - Generic Format:

```
do {
        statement;
} while ( condition );
```

break, continue Statements

The break statement

- **break**
  - Causes **immediate exit** from a **while, for, do/while or switch** structure
  - Program execution **continues** with the first statement after the structure
  - Common uses of the **break** statement
    - Escape **early** from a loop
    - Skip the remainder of a **switch** structure

The continue statement

Used for skipping the remainder of the body of a **while, for or do/while** structure and proceeding with the next iteration of the loop

- **while and do/while**
  - Loop-continuation test is evaluated immediately after the **continue** statement is executed
- **for**
  - Increment expression is executed, then the loop-continuation test is evaluated

Revision

Mind Term Exam Revision

- **Data Types**
  - int
  - float
  - double
  - char
  - etc.
- **Escape Sequences**
  - \n, \t, \\, \', \", etc.
- **Naming Convention**
  - Name of Variables

Mind Term Exam Revision

- **Comments**
  - // (single line), /* */ (multi-lines)
- **Expression and Operators**
  - Logical, Arithmetic, Increment/decrement, Relational
- **Casting of Data types**
- **Control Structures**
  - If , if – else, nest if – else, switch
- **Flow chart, Pseudo code**

Mind Term Exam Revision

- **C++ Program Development Environment**
  - Source code, object code, preprocessor, compiler, assembler, interpreter, loader, linker, debugger
- **Memory Concepts**
- **Continue, Break statements**