"If I have seen further it is by standing on the shoulders of giants."
-Isaac Newton

"Words are a pretext. It is the inner bond that draws one person to another, not words."
— Jalaluddin Rumi

Chapter 3

Introduction To C++ Language

Generic Table Printing

Summary

Like human beings, computers have their own languages that we use to interact with / operate them. These languages are also called programming language and C++ is one of the most famous and widely used programming languages. Just like human languages have grammar, computer languages have syntax[1] and logical rules to link the steps of instruction to form a computer program.

Actually the computer only understands the language of the 0's and 1's and this language is called **binary language**. This language is difficult for humans to understand, so computer scientists have made programs which convert an easier human friendly language to binary language. Such a program is known as a **compiler**. Binary language is also called a "low-level" language, in the sense that it is close to the electronics of a computer. The programming languages that are easy for humans to understand are called "high-level" language, and C++ is one such language.

Let us now dive into learning the basics of the language we use in our book, that is C++. *Always remember that the language constructs (its syntactic(grammatical), morphological (keywords) and semantics rules) will only teach you the tool to express your computational thought. Learning those computational ideas will be the most important journey you will experience throughout this book*.

Writing Our First Computer Program

Let us write our first C++ program which should write (we also use the word "print" for this) something on our Console Window(black window). This is the basic means of interaction between the end user and our code. You may think of it as a blackboard where the running program will display some output. Also, the user can give some input to the running program by typing something using the keyboard. This is called **prompting** the user for input, after that input our program will behave as the desired requirement of the user. We will be using CodeBlocks for our book. We have chosen Codeblocks for its primitive nature and being lightweight.
To install CodeBlocks, create and run our program, See **Appendix I**

Example 1: Printing a Welcome Message on the Console Window...!

Let us first write a program that prints a welcome message to display to the user.
Code

[1] Rules that specify what is a valid statement and what is not. Otherwise, we could enter commands in nonsensical ways and the computer will have to reject them.

```
1  #include <iostream>
2  using namespace std;
3
4  //This program should print the message inside the quotes
5  int main()
6  {
7     cout << "Hello Turing... Welcome  :) to the world of C++" << endl;
8     return 0;
9  }
```

Output

```
Hello Turing :) to the world of C++

Process returned 0 (0x0)   execution time : 0.092 s
Press any key to continue.
```

Link

We strongly advise that you should always write your code yourself in your code editor and execute this program to see its output. Still, just in case your code did not compile or run correctly due to some errors, you may download it from here: http://codepad.org/XJOMHbBe

Description

In lines 1 and 2, "header files" are included in our program (for now, write them as is, you will understand them in a while). You can think of these lines as there are a lots of work related to many things like printing, reading from files from user and maths work and so on that are already done by many amazing programmers and they have provided those works in the form of libraries and we can add them in our code, "just like you don't want to reinvent the wheel rather you would use it, that is why we would like to use the prior work done by many programmers in the form of libraries".

TO REMEMBER

header files are included in our program because it is something predefined for us and `cout` statement is one of its built-in functionality. We include header files according to our need. Here we added `iostream` which is used to print on our command prompt like console screen.

Line 5 is our starting of **main**, every C++ program must have one and only one **main**. From line 6 to line 9 the parentheses **{ }** represent the body of our main flow. In line 7 **cout** is standard syntax to display the words or characters inside quotations.The symbol **<<** represents the C++ output operator and **endl** is used for new line(next line). In line 8 **return 0** means our program is ended.
Finally, note that every program statement must end with a semicolon **.**

```
cout << "message 1 " << "message 2 " << "message 3";
```

Now as you can see in the above line of code multiple messages accommodates in single **cout** with the help of symbol **<<** it is also used as a separator.

TO REMEMBER

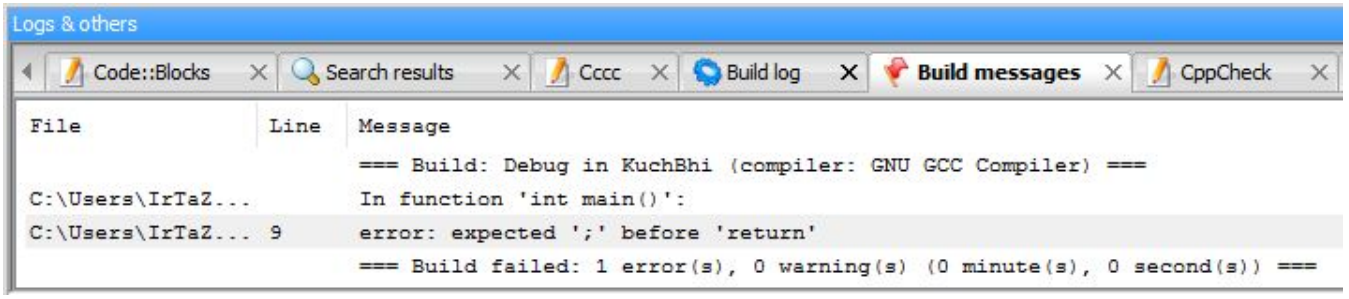
Don't worry about the syntax errors (in case if you miss any semicolon or misspell any thing) much if you forget it again and again, that is quite normal and you will get used to of it after a while.

Syntax Errors in the Program

```
1  #include <iostream>
2  using namespace std;
3
4  //error in line number
5
6  int main()
7  {
8     cout << "Hello world!" << endl
9     return 0;
10 }
```

In above picture there is piece of code with error in line number 8. The red square box indicates that there is error in line 8.

Error: Semicolon is missing



This window from the CodeBlocks shows errors and warnings.
Now let's move to next examples.

Example 2: Basic Arithmetic Operations(+, *, -)

Let's write a program which can do simple arithmetic operations.

Code

```
1  #include <iostream>
2  using namespace std;
3
4  //This program should do different arithmetic operations
5
6  int main()
7  {
8     cout << 2*3 << endl << 2+3 << " " << 2-3 << endl;
9     return 0;
10 }
```

Output

```
6
5 -1
```

Link
http://codepad.org/Yb2q4KsY

Description
In line 8 **cout** is working in the following fashion. Everything which is written in between two **<<** will be printed one by one. If the message inside the two **<<** is within double quotes then it will be printed as it is. But if the message is something like a mathematical expression it will be first evaluated (automatically in the background by the compiler) and that evaluated value will be replaced as the message e.g. **2*3** will be replaced by **6** similarly **2+3** will be replaced by **5** and **2-3** will be replaced by **-1**.

Remember
Don't need to use quotations for arithmetic computation, in above code we use quotations just to give a space between numbers as you see in console output.

Example 3: Table Printing of 23

We want to make a program which should print a table of any number like 23.
We will use what we have learnt in the previous examples to write this very simple program. The idea is to use **cout** and print the table line by line by computing the mathematical expression as well as the message written in the tabular format. Here is the program.

Code

```
1  #include <iostream>
2  using namespace std;
3
4  //This program should print the table of 23 up to 10th multiple
5
6  int main()
7  {
8     cout << "23 x  1 = " << 23*1 << endl;
9     cout << "23 x  2 = " << 23*2 << endl;
10    cout << "23 x  3 = " << 23*3 << endl;
11    cout << "23 x  4 = " << 23*4 << endl;
12    cout << "23 x  5 = " << 23*5 << endl;
13    cout << "23 x  6 = " << 23*6 << endl;
14    cout << "23 x  7 = " << 23*7 << endl;
15    cout << "23 x  8 = " << 23*8 << endl;
16    cout << "23 x  9 = " << 23*9 << endl;
17    cout << "23 x 10 = " << 23*10 <<endl;
18    return 0;
19 }
```

Output

```
23 x 1 = 23
23 x 2 = 46
23 x 3 = 69
23 x 4 = 92
23 x 5 = 115
23 x 6 = 138
23 x 7 = 161
23 x 8 = 184
23 x 9 = 207
23 x 10 = 230
```

Link
http://codepad.org/rXYZZOaf

Description
In this program **cout** is used to print table of 23 upto 10th multiple. Note that whenever you write something in between the double quotes **"MESSAGE"** that **MESSAGE** will be printed on the console screen as it is. But if we write a mathematical expression without quotes, C++ is intelligent enough that the mathematical expression is replaced by the result of the mathematical expression. For example **23*2** will be replaced by **46** and that result will be printed on the screen.

REMEMBER
Now you have to remember that the material inside the quotes "" is printed on console screen as it is and other computation expression (e.g 23*1, 23*2) is written without quotations for arithmetic computation.

Example 4: Table Printing of 73 upto 10th Multiple

We want to make a program which should print a table of 73 upto 10th multiple.
The similar concept as previous example can be modified for printing the table of 73.

code

```cpp
#include <iostream>
using namespace std;

//This program should print the table of 73 up to 10th multiple

int main()
{
   cout << "73 x  1 = " << 73*1 << endl;
   cout << "73 x  2 = " << 73*2 << endl;
   cout << "73 x  3 = " << 73*3 << endl;
   cout << "73 x  4 = " << 73*4 << endl;
   cout << "73 x  5 = " << 73*5 << endl;
   cout << "73 x  6 = " << 73*6 << endl;
   cout << "73 x  7 = " << 73*7 << endl;
   cout << "73 x  8 = " << 73*8 << endl;
   cout << "73 x  9 = " << 73*9 << endl;
   cout << "73 x 10 = " << 73*10<< endl;
   return 0;
}
```

Output

```
73 x 1 = 73
73 x 2 = 146
73 x 3 = 219
73 x 4 = 292
73 x 5 = 365
73 x 6 = 438
73 x 7 = 511
73 x 8 = 584
73 x 9 = 657
73 x 10 = 730
```

Link

http://codepad.org/Z2C30gCL

Exercise

1. Print the table of 35 up-to 20 multiples?
2. Print the table of 43 from 13 to 30 multiple?

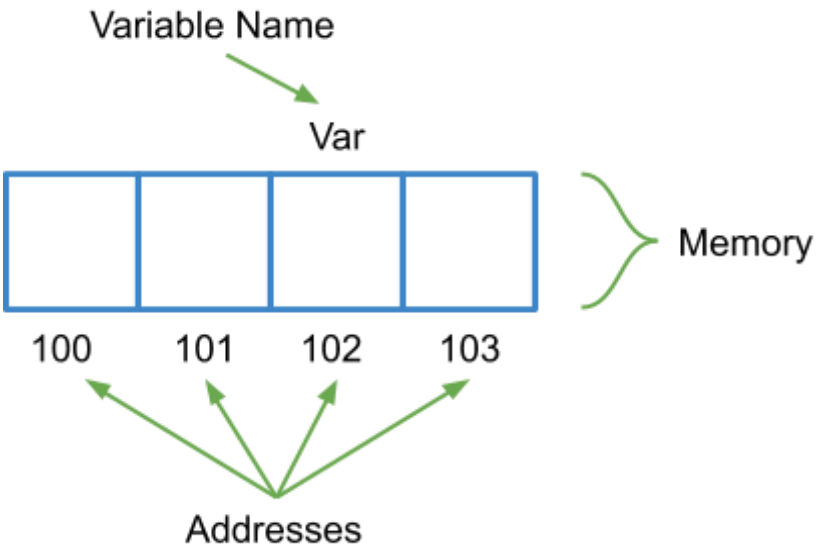
What if we have to print the table of 27 then as you see we change 23 or 73 in every line so it is a very cumbersome act to change the code again and again whenever you want to change the table to some other number's table. **Can we do something so that the change in requirement of a different table will require a minimum cumbersome change in the code?**

Variables, Memory and assignment (=) statement

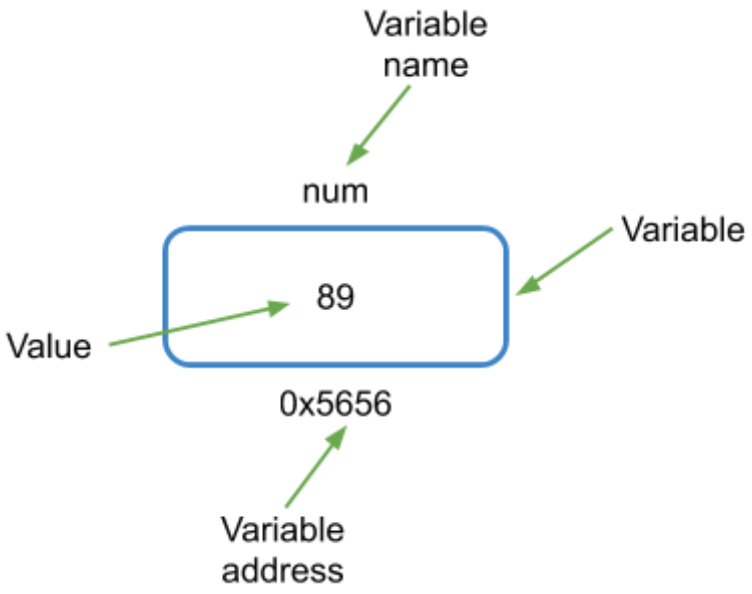
Variables and Memory

Now let's move to a new concept called Variables. Variable is a named location in computer's memory which store some value. You can later use that value by that name of the variable. Variables are of different types which determines the size and layout of variable's memory for example integers, floating point numbers like real values, character(s), bool (true/1 or false/0) etc. In this chapter we will only use integer type variable so don't worry about the rest, you will going to learn these in coming chapters.

For better understanding of variables you need to know a little bit about computer's memory, computer memory is a kind of boxes aligned up in a row each box(memory) numbered sequentially. These numbers are known as memory addresses.



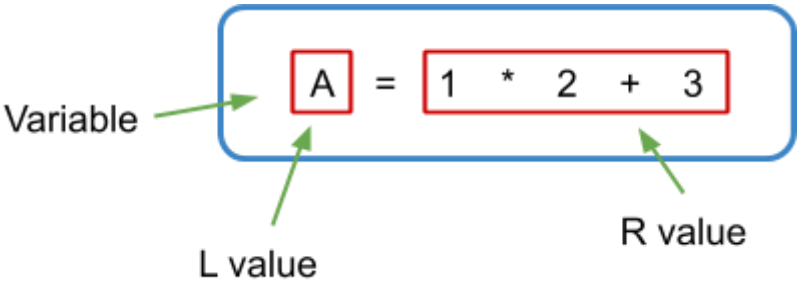
You can also give a name to a memory location which you can later access by its name (compiler in the background manages itself how to replace the value of the named memory). For example, you create a variable named num in which 89 is stored as shown in image below



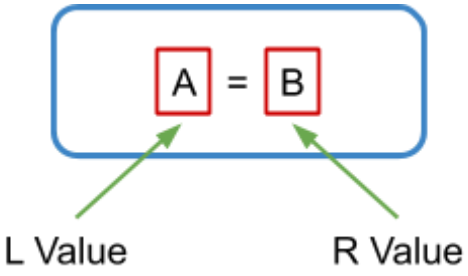
Assignment Statement (=)

The most important and common statement in C++ is Assignment Statement, in C++ it is used for computation and setting the value to a variable. The symbol that is used for computation and assigning the value to a variable is called assignment operator denoted by =. This operator is just like the equal sign of mathematics, but in C++ it has a different meaning.

As stated earlier assignment operator is used for computation and assigning the value to a variable. We can assign a single value or an expression to a variable as shown in image below

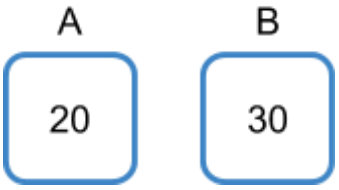


In this image R value(right value) is assigned to variable A(L value / left value), it doesn't mean that A is equal to R value so don't confuse. You can also assign a value of one variable to another as shown in the image below

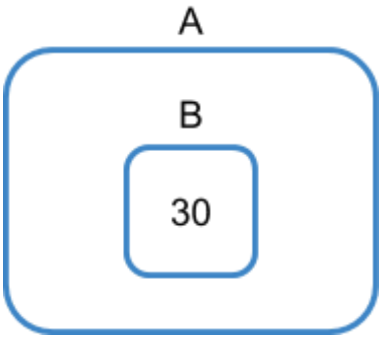


Suppose that variable **A** holds **20** and **B** holds **30**.

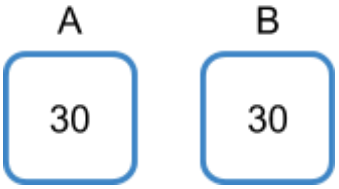
```
int A = 20;
int B = 30;
```



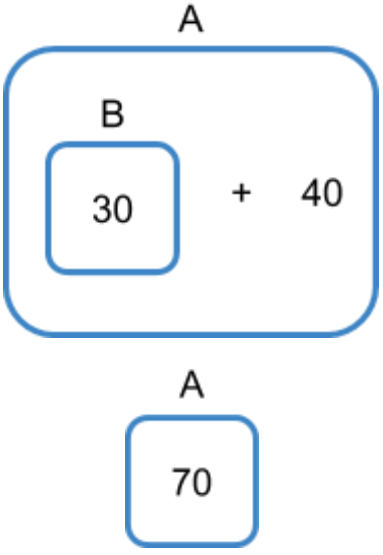
If we have to assign the value of one variable to another we simply write this line of code
To assign value of a variable to another variable simply this line will be written.
A = B;
In this line of code **A** is L-value(left value) and **B** is R value(right value). When this line will execute **A** hold the value of **B** as shown in the image below.



This means **A** holds the value of **B**.



Now, for better understanding we assign arithmetic expression to a variable.
A = B + 40;



Variable Declaration and Initialization

To declare any variable you should first write its data type (i.e. which type of data the variable will hold) then the name of variable. For example:

```
int num = 34;
```

Like in the above example variable `num` is initialized with 34. You can else declare multiple variables by using comma like below

```
int num1, num2, num3;
```

Declaration and initialization process can also be done in one line like this,

```
int a = 5, b = 3, c = 0;
```

Variable have different data types and computer gave them space(just like a box) according to the data type.

In the picture above addresses of variables is not shown because compiler handle it internally and you don't need to worry about it, when you make a variable compiler assign the address automatically.
Integer is a data type in C++ used to store whole numbers.

Remember
integer type variable do not store fractional number (e.g 2.1, 10.6). There is another type used to store such numbers which you can learn in next chapter.

Rules for naming the variables

In C++ there are some rules for naming a variable which are as follows.

- Keywords(reserved words) cannot be used as a variable name.
- Variable name cannot be start with number or special character.
- In C++, variable name must start with alphabets or underscore(_).
- After the first letter initialization of variables, numbers can be used but space and special character is still not allowed.

See the link below for the list of C++ keywords(reserved words).
http://en.cppreference.com/w/cpp/keyword
See the link below for the list of special characters.
https://www.owasp.org/index.php/Password_special_characters

Example 5: Integer Variable Declaration, Initialization and Execution

Let's write a program which declare a variable, initialize and execute it.
Code

```
1  #include <iostream>
2  using namespace std;
3
4  //This program declare and initialize the integer variable with 34
5  //num is the name of variable
6
7  int main()
8  {
9     int num;
10    num = 34;
11
12    cout << "Number = " << num << endl;
13    return 0;
14 }
```

Output

```
Number = 34
```

Link

http://codepad.org/eYWAeVU2

Description

In this example an integer type variable is created in line number 9 and initialize that variable with 34 on line number 10, later on the number is displayed on console screen by using **cout** on line number 12.

Example 6: Arithmetic Operations using Integer Variable

In this example we are going to perform simple arithmetic operations using integer type variable.

Code

```
1  #include <iostream>
2  using namespace std;
3
4  //This program should perform arithmetic operation on variable
5
6  int main()
7  {
8     int num = 34;
9
10    cout << num+1 << endl;
11    cout << num*2 << endl;
12    cout << num-1 << endl;
13    return 0;
14 }
```

Output

```
35
68
33
```

Link
http://codepad.org/C0oWIPhB

Description
As you see in this example from the result we came to know that wherever the variable(**num**) is written the compiler automatically replaces that expression's spot by its value and the mathematical expression solve and that statement replace with a single value .
Now play with variables, change the value of variable and see the results.

Program 7.1: Table printing using Variable(**int**)

This is an example of table printing using integer type variable.
code

```
1  #include <iostream>
2  using namespace std;
3
4  //This program print the table of 73 up to 10th multiple
5
6  int main()
7  {
8     int Table = 73;
9
10    cout << "Table x  1 = " << Table*1 << endl;
11    cout << "Table x  2 = " << Table*2 << endl;
12    cout << "Table x  3 = " << Table*3 << endl;
13    cout << "Table x  4 = " << Table*4 << endl;
14    cout << "Table x  5 = " << Table*5 << endl;
15    cout << "Table x  6 = " << Table*6 << endl;
16    cout << "Table x  7 = " << Table*7 << endl;
17    cout << "Table x  8 = " << Table*8 << endl;
18    cout << "Table x  9 = " << Table*9 << endl;
19    cout << "Table x 10 = " << Table*10<< endl;
20    return 0;
21 }
```

Output

```
Table x 1 = 73
Table x 2 = 146
Table x 3 = 219
Table x 4 = 292
Table x 5 = 365
Table x 6 = 438
Table x 7 = 511
Table x 8 = 584
Table x 9 = 657
Table x 10 = 730
```

Link
http://codepad.org/eUykwczX

Description
We saw that **Table** inside the double quotes do not replace by the value rather just the Label and the table outside the quotes evaluate and replace by the value of variable and then replace by the answer.

Program 7.2: Modified Version of program 7.1

code

```
1  #include <iostream>
2  using namespace std;
3
4  //This program print the table of 73 up to 10th multiple
5
6  int main()
7  {
8     int Table = 73;
9
10    cout << Table << " x  1 = " << Table*1 << endl;
11    cout << Table << " x  2 = " << Table*2 << endl;
12    cout << Table << " x  3 = " << Table*3 << endl;
13    cout << Table << " x  4 = " << Table*4 << endl;
14    cout << Table << " x  5 = " << Table*5 << endl;
15    cout << Table << " x  6 = " << Table*6 << endl;
16    cout << Table << " x  7 = " << Table*7 << endl;
17    cout << Table << " x  8 = " << Table*8 << endl;
18    cout << Table << " x  9 = " << Table*9 << endl;
19    cout << Table << " x 10 = " << Table*10<< endl;
20    return 0;
21 }
22 /*
23 Note: That if now you want to change the Table printing to let us say some other value e.g. 39 then all you
24 need to change is just one line i.e. Line # 8 change it to
25 int Table = #VALUE
26 and you are done with the desired result.
27 */
28
```

Description
We saw in the previous program that the **Table** inside the double quotes do not replace by the value, so this time we put **Table** outside the double quote and there it changes with its value and replace with answer.

Output

```
73 x  1 = 73
73 x  2 = 146
73 x  3 = 219
73 x  4 = 292
73 x  5 = 365
73 x  6 = 438
73 x  7 = 511
73 x  8 = 584
73 x  9 = 657
73 x 10 = 730
```

Link
http://codepad.org/RRtKV1Qt

Program 8: Table printing using Variable(**int**)

This is a program of table printing using integer type variable.

code

```
#include <iostream>
using namespace std;

//This program print the table of 55 upto 10th multiple

int main()
{
   int Table = 55;

   cout << Table << " x 1 = " << Table*1 << endl;
   cout << Table << " x 2 = " << Table*2 << endl;
   cout << Table << " x 3 = " << Table*3 << endl;
   cout << Table << " x 4 = " << Table*4 << endl;
   cout << Table << " x 5 = " << Table*5 << endl;
   cout << Table << " x 6 = " << Table*6 << endl;
   cout << Table << " x 7 = " << Table*7 << endl;
   cout << Table << " x 8 = " << Table*8 << endl;
   cout << Table << " x 9 = " << Table*9 << endl;
   cout << Table << " x 10 = " << Table*10 << endl;
   return 0;
}
```

Output

```
55 x 1 = 55
55 x 2 = 110
55 x 3 = 165
55 x 4 = 220
55 x 5 = 275
55 x 6 = 330
55 x 7 = 385
55 x 8 = 440
55 x 9 = 495
55 x 10 = 550
```

Link
http://codepad.org/rplaFuor

Description
This program is just like the program 7.2 but in this the value of variable is changed and you see the whole table is changed. Also tried changing the variable value and looked at Table of 93, 102, 174 etc.
Now what if we want to print the table of different numbers we have to change the value of variable in the code every time in the next topic you are going to learn a new concept of input from user.

Exercise

1. Print the table of 89 with the help of integer variable up-to 25 multiples?
2. Print the table of 43 with the help of integer variable from 23 to 56 multiple?

Taking Input from User (the use of cin)

In C++ **cin** is used to take input from user and it is predefined in **<iostream>** remember that **cin** is not a keyword. In C++ **cin** and **cout** is standard input and output stream, **cin** is a variable of type istream(input stream), **cin** means standard console input. User can input the data from keyboard, file, screen etc.
cin is a syntax to take input from user at run time and store the input value in variable, arrays etc. As we see in **cout** insertion operator**(<<)** was used to separate the output message, when using **cin** extraction operator**(>>)** will be used to take input separately.

Program 9: Taking Input from User and Print

This program shows you how to take input from user and display the result.
Code

```cpp
#include <iostream>
using namespace std;

//This program take the input from user and display that number

int main()
{
   int num;
   cout << "Enter number: ";
   cin >> num;

   cout << "Number = " << num << endl;
   return 0;
}
```

Output

```
Enter number: 159
Number = 159
```

Link

http://codepad.org/e22Ss3hA

Description

In the above program **cin** is used to take input from user and store that input in variable **num** and at the end the same input in displayed on console screen using **cout**.

Remember

If you are taking input for an integer variable then you are bound to write the same type of input.
For example: ***10, 45, 97 etc.***
if you input fractional numbers like 10.5, 36.9 or 65.7 the integer variable only stores 10, 36 or 65 respectively.

Program 10: Taking Input(number) from User and Print the Table

This is a program of table printing by taking input from user.

Code

```cpp
#include <iostream>
using namespace std;

//This program should print the table according to user input number

int main()
{
   int Table;
   cout << "Which table you want to Print: ";
   cin >> Table;

   cout << Table << " x 1 = " << Table*1 << endl;
   cout << Table << " x 2 = " << Table*2 << endl;
   cout << Table << " x 3 = " << Table*3 << endl;
   cout << Table << " x 4 = " << Table*4 << endl;
   cout << Table << " x 5 = " << Table*5 << endl;
   cout << Table << " x 6 = " << Table*6 << endl;
   cout << Table << " x 7 = " << Table*7 << endl;
   cout << Table << " x 8 = " << Table*8 << endl;
   cout << Table << " x 9 = " << Table*9 << endl;
   cout << Table << " x 10 = " << Table*10 << endl;
   return 0;
}
```

Output

```
Which table you want to Print: 56
56 x 1 = 56
56 x 2 = 112
56 x 3 = 168
56 x 4 = 224
56 x 5 = 280
56 x 6 = 336
56 x 7 = 392
56 x 8 = 448
56 x 9 = 504
56 x 10 = 560
```

Link

http://codepad.org/SR6BJK6X

Description

In this program we use simple **cin** and concept of variables and print the table but this time according to user's input and we don't have to change the table value in our whole program as we did before, now we just enter the value and table is printed.

Program 11: Taking Input(number) from User and Print the Table up to 15th Multiple

This program takes input from user and print the table of that input(number) upto 15th multiple.

Code

```cpp
#include <iostream>
using namespace std;

//This program should print the table from input number up to 15th multiple

int main()
{
   int Table;
   cout << "Which table you want to Print: ";
   cin >> Table;

   cout << Table << " x 1 = " << Table*1 << endl;
   cout << Table << " x 2 = " << Table*2 << endl;
   cout << Table << " x 3 = " << Table*3 << endl;
   cout << Table << " x 4 = " << Table*4 << endl;
   cout << Table << " x 5 = " << Table*5 << endl;
   cout << Table << " x 6 = " << Table*6 << endl;
   cout << Table << " x 7 = " << Table*7 << endl;
   cout << Table << " x 8 = " << Table*8 << endl;
   cout << Table << " x 9 = " << Table*9 << endl;
   cout << Table << " x 10 = " << Table*10 << endl;
   cout << Table << " x 11 = " << Table*11 << endl;
   cout << Table << " x 12 = " << Table*12 << endl;
   cout << Table << " x 13 = " << Table*13 << endl;
   cout << Table << " x 14 = " << Table*14 << endl;
   cout << Table << " x 15 = " << Table*15 << endl;
   return 0;
}
```

Output

```
Which table you want to Print: 56
56 x 1 = 56
56 x 2 = 112
56 x 3 = 168
56 x 4 = 224
56 x 5 = 280
56 x 6 = 336
56 x 7 = 392
56 x 8 = 448
56 x 9 = 504
56 x 10 = 560
56 x 11 = 616
56 x 12 = 672
56 x 13 = 728
56 x 14 = 784
56 x 15 = 840
```

Link

http://codepad.org/0pmZrgLu

Description

We saw that every change in the range(multiples) caused the new addition of lines or removing some lines. Like if we would like to print the table up to 50th iteration or 100'th iteration then we have to write 50 or 100

times the lines with multiplicand changed in each line.

To avoid this repetition of similar kind of code we will learn an extremely powerful construct of repetition called loop.

Repetition and **while** loop

To repeat a process (set of statements) in C++ we use loops. Loops just make our work a lot more easier and make our code smaller. There are different kind of loops, **while** is one of its type.

It's syntax is,

```
while(condition)
{
        Body(statements) of while
}
```

If condition is true then the Body part of the loop will be executed. After completely executing the Body part the condition of the while loop will be checked again if it is true then the body will be executed again, and this process will continue until the condition becomes false.
In this program below the message **Hello** print continuously.

Remember
If the loop repeat it's iteration continuously and never stops this type of loop is called infinite loop.

FOR loop

It's syntax is,

```
for(initialization; condition; increment or decrement)
{
        Body(statements) of for loop
}
```

The initialization expression is used to declare and/or initialize control variable(s) for the loop; it is evaluated first, before any iteration occurs. The condition expression is used to determine whether the loop should continue iterating; it is evaluated immediately after the initialization; if it is true, the statement is executed. The increment/decrement expression is used to increase or decrease the control variable(s); it is evaluated after the statement is executed. So the sequence of events that generate the iteration are:
1. evaluate the initialization expression
2. if the value of the condition expression is false, terminate the loop
3. execute the statement
4. evaluate the update expression
5. repeat steps 2 – 4.

Program 12: Infinite Hello Printing using While loop

This program prints the message continuously(infinitely) using while loop.
Code

```
1  #include <iostream>
2  using namespace std;
3
4  //This program should print the quoted message infinitely
5
6  int main()
7  {
8     while(true)
9     {
10        cout << "Hello" << endl;
11    }
12    return 0;
13 }
```

Output

```
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
```

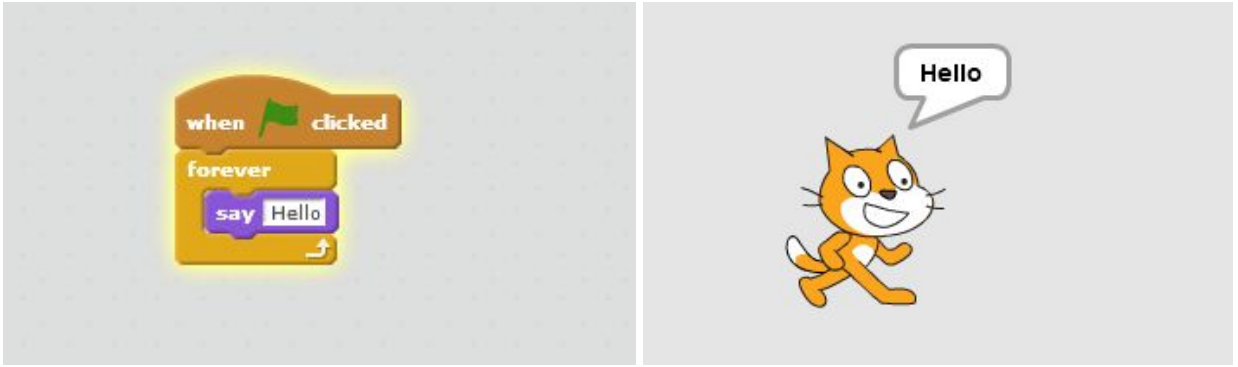
Link: http://codepad.org/xKtC2kvx

Description

You can terminate the infinite loop by pressing **ctrl + c** keys or by closing the console window.

For Those who did Chapter 2

This example of **scratch** is just like above **program 12** of infinite while loop.



Program 13: Printing Hello 10 Times using While loop

This program print the value inside the quotes and also show the incremented value of time.

Code

```
1  #include <iostream>
2  using namespace std;
3
4  /*This program should print the quoted message
5  until the condition of while loop get false.
6  In this program while loop runs 10 times with
7  integer variable*/
8
9  int main( )
10 {
11    int Time = 1;
12
13    while(Time  <= 10)
1     {
4        cout << Time << ": hello" << endl;
15       Time = Time + 1;
16    }
17    return 0;
   }
```

```
#include <iostream>
using namespace std;

/*This program should print the quoted message until the
condition of while loop get false.
In this program for loop runs 10 times with integer variable*/

int main( )
{
   for(int Time=1; Time<10; Time++)
   {
      cout << Time << ": hello" << endl;
   }
   return 0;
}
```

Output

```
1: hello
2: hello
3: hello
4: hello
5: hello
6: hello
7: hello
8: hello
9: hello
10: hello
```

Link

http://codepad.org/JgV6SWic

Description

In this program On line 9, we created a variable **Time** and assign 10 into it and a **while** loop with a condition(**Time <= 10**) and until the condition remains true, when condition become false the statements outside the loop will be execute. In line 12 **while** is started with a condition when the condition will execute the **Time** will be replaced with its value. It means until the value of **Time** becomes 11 the loop will not be terminated. Line number 15 means we are changing the value of **Time** by adding 1 to the current value of **Time** and then again assign a new incremented value to **Time**.

Program 13.1: Printing Hello 10 Times using For loop

This program print the value inside the quotes and also show the incremented value of time.

Code

```
1  #include <iostream>
2  using namespace std;
3
4  /*This program should print the quoted message until the condition of for loop get false.
5  In this program for loop runs 10 times with integer variable*/
6
7  int main()
8  {
9     for(int Time = 1; Time <= 10; Time++)
10    {
11       cout << Time << ": hello" << endl;
12    }
13    return 0;
14 }
```

Output

```
1: hello
2: hello
3: hello
4: hello
5: hello
6: hello
7: hello
8: hello
9: hello
10: hello
```

Link: http://codepad.org/83PwQgCG

Description

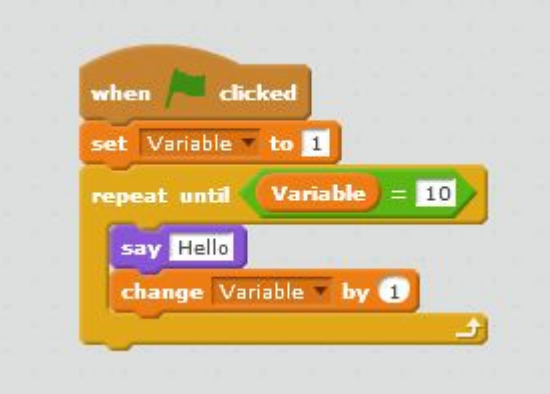
This program will print a message hello 10 times on 10 lines. First it will print a hello message then goto next line then again repeat this process 10 times in total.

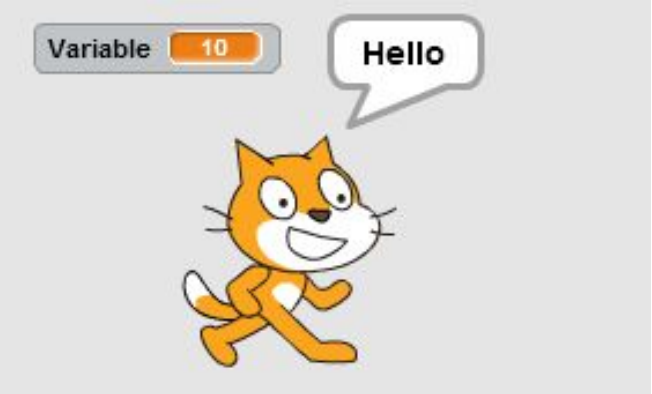
On line number 9 of this program there is a for loop written. In this loop at line number 9 first of all an integer variable will be assigned a value of 1. Then at very next the condition part will execute it checks either the value stored in Time is less than 10 or not(our loop is dependent on the condition loop will iterate until the condition becomes false). Then the body of the loop will execute the hello message and **endl** instruction and then the increment part will execute it means after printing message there will be an increment of 1 in Time so the value of time will become 2. Loop repeats the process until the condition become false (until the value of time becomes greater than 10, it means after the loop iterate completely the value of time will be 11 .

For Those who did Chapter 2

This example of **scratch** is just like above **program 13 and 13.1** of while and for loop that runs 10 times.

It is just like we did in chapter 2 scratch we studied the **repeat until** construct that were same like **while and for** loop.





Program 14: Table Printing using While Loop

This program print the table of user input number upto 15th multiple using while loop.

code

```
#include <iostream>
using namespace std;

//This program should print the table up to 15th multiple
//Table printing using while loop according to user input

int main()
{
   int Table;
   cout << "Which Table: ";
   cin >> Table;          //Let us say user entered 12

   int i=1;
   while(i <= 15)
   {
      cout << Table << " x " << i << " = " << Table*i << endl;
      i = i + 1;
   }
   return 0;
}
```

Output

```
Which Table: 12
12 x 1 = 12
12 x 2 = 24
12 x 3 = 36
12 x 4 = 48
12 x 5 = 60
12 x 6 = 72
12 x 7 = 84
12 x 8 = 96
12 x 9 = 108
12 x 10 = 120
12 x 11 = 132
12 x 12 = 144
12 x 13 = 156
12 x 14 = 168
12 x 15 = 180
```

Link
http://codepad.org/6PayfOwd

Description
What we want to do in this program is to print table by the user's choice(input) upto 15. At line 16 we write a specific format to print table and then we multiply table num with the number of iteration and on very next line we just increment in the iteration variable **i** so that every time the table multiplies with incremented value of **i**.

Program 14.1: Table Printing using For Loop

This program print the table of user input number upto 15th multiple using for loop.

Code

```
1  #include <iostream>
2  using namespace std;
3  //This program should print the table up to 15th multiple
4  //Table printing using while loop according to user input
5
6  int main()
7  {
8     int Table;
9     cout << "Which Table: ";
10    cin >> Table;        //Let us say user entered 12
11
12    for(int i =  1; i <= 15; i++)
13    {
14       cout << Table << " x " << i << " = " << Table*i << endl;
15    }
16    return 0;
17 }
```

Output

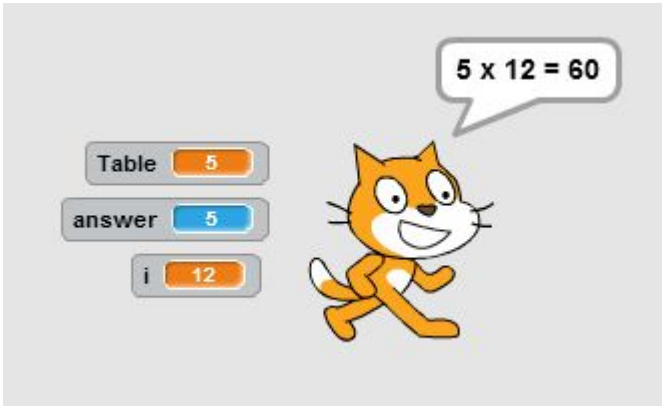
```
Which Table: 12
12 x 1 = 12
12 x 2 = 24
12 x 3 = 36
12 x 4 = 48
12 x 5 = 60
12 x 6 = 72
12 x 7 = 84
12 x 8 = 96
12 x 9 = 108
12 x 10 = 120
12 x 11 = 132
12 x 12 = 144
12 x 13 = 156
12 x 14 = 168
12 x 15 = 180
```

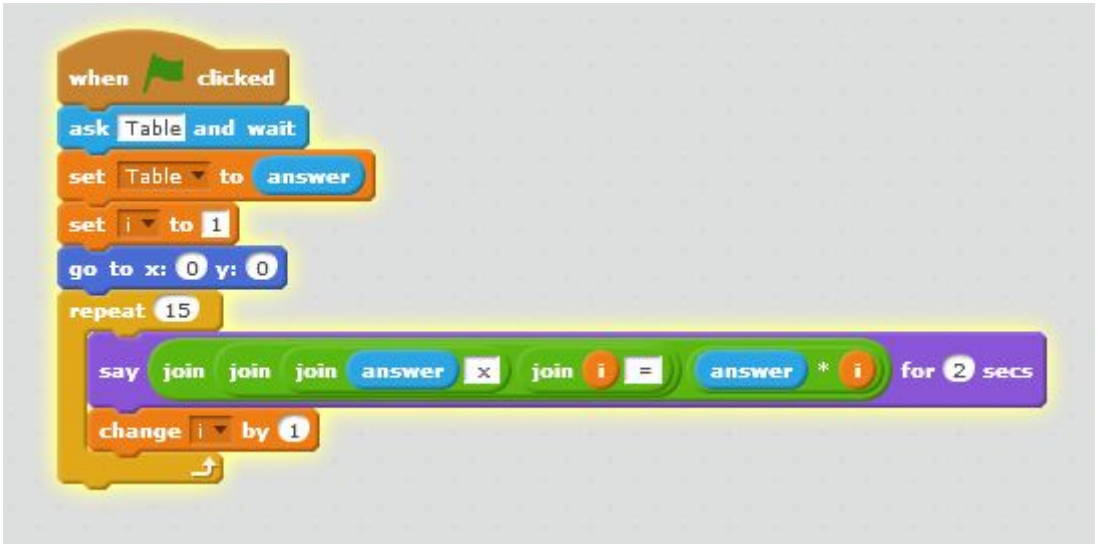
Link: http://codepad.org/wyluAOqK

Description

For Those who did Chapter 2

This example of **scratch** is just like above **program 14 and 14.1** of while and for loop that print the table the only difference is the loop used is for loop.





Program 15: Table Printing up to User Entered Limit using While Loop

This program takes two input from user about the table and limit(multiples) and print the table of user entered number upto user entered limit(multiples).

code

```cpp
#include <iostream>
using namespace std;

//Table printing up to the limit(multiples) entered by user
//While loop that print table according to user input

int main()
{
   int Table;
   int Limit;

   cout << "Which Table: ";
   cin >> Table;          //Let us say user entered 5
   cout << "Limit: ";
   cin >> Limit;          //let us say user entered 16

   int i=1;
   while(i <= Limit)
   {
      cout << Table << " x " << i << " = " << Table*i << endl;
      i = i + 1;
   }
   return 0;
}
```

Output

```
Which Table: 5
Limit: 16
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
5 x 11 = 55
5 x 12 = 60
5 x 13 = 65
5 x 14 = 70
5 x 15 = 75
5 x 16 = 80
```

Link

http://codepad.org/lssSef3n

description

This program is slightly different from the previous one. The main difference is that in this program two variables are used, one is as a counter and the other one is as a limit that is exactly how many time the loop will be executed. It means now user will tell us that exactly for how many times the loop will be executed.

In line 9 and 10 table and the limit variable is declared.
In line 12, 13, 14 and 15 we are getting value for table and limit.
In line 17 the counter variable **i** of integer data type is declared and assigned a value to 1.
In line 18 **while** and it's condition(**i <= Limit**) is written the condition actually means the loop will executed until the value of counter become greater than **Limit**.

Program 15.1: Table Printing up to User Entered Limit using For Loop

This program takes two input from user about the table and limit(multiples) and print the table of user entered number upto user entered limit(multiples).

Code

```
1  #include <iostream>
2  using namespace std;
3  //Table printing up to the limit(multiples) entered by user
4  //For loop that print table according to user input
5
6  int main()
7  {
8     int Table;
9     int Limit;
10    cout << "Which Table: ",
11    cin >> Table;          //Let us say user entered 5
12    cout << "Limit: ";
13    cin >> Limit;          //let us say user entered 16
14
15    for(int i = 1; i <= Limit; i++)
16    {
17       cout << Table << " x " << i << " = " << Table*i << endl;
18    }
19    return 0;
20 }
```

Output

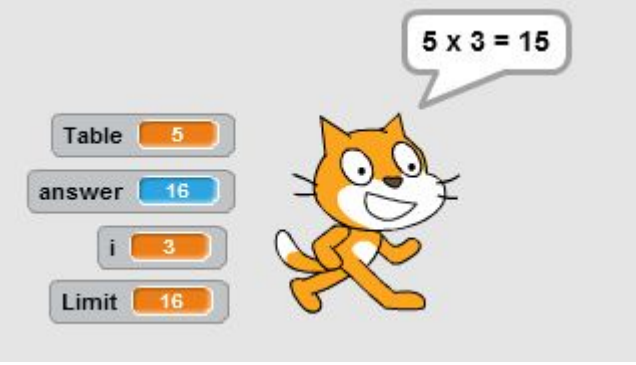
```
Which Table: 5
Limit: 16
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
5 x 11 = 55
5 x 12 = 60
5 x 13 = 65
5 x 14 = 70
5 x 15 = 75
5 x 16 = 80
```

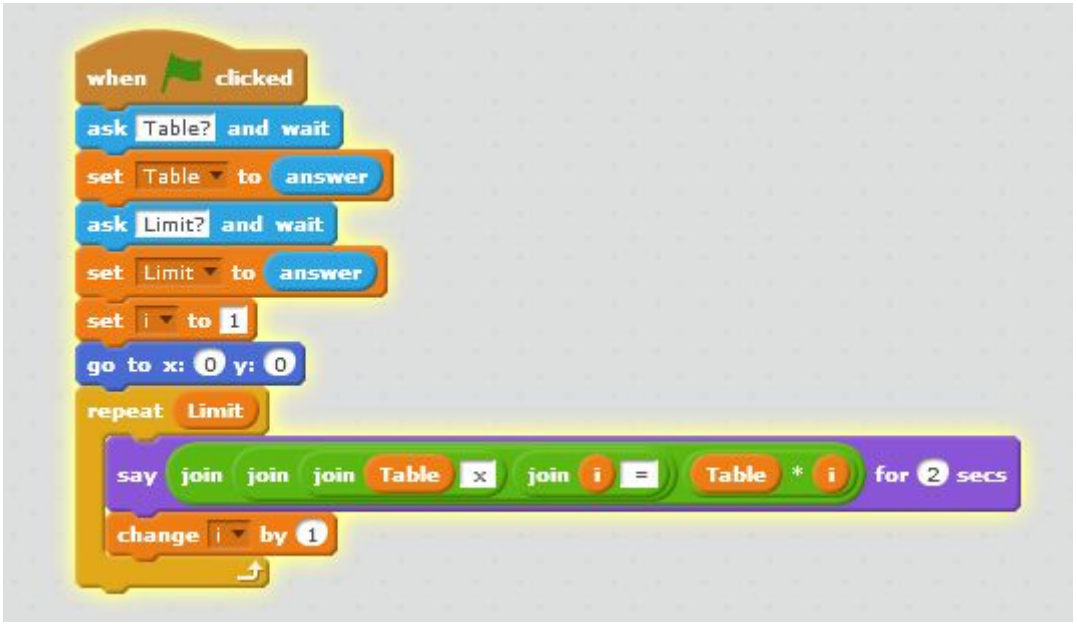
Link: http://codepad.org/qup4EWKf

Description

For Those who did Chapter 2

This example of **scratch** is just like above **program 15 and 15.1** of while and for loop that prints the table up to limit(multiple).





Program 16: Generic Table Printing (User Entered Table and Limits from start to end)

This program is slightly different from previous one this program takes three inputs from the user about table, starting multiple and ending multiple. In short it is generic program.

Code

```
#include <iostream>
using namespace std;

//Table printing within the user entered limit(multiple)
//example of while loop that print table according to user input

int main()
{
   int Table;
   int Start, End;

   cout << "Which Table: ";
   cin >> Table;            //Let say user entered 7
   cout << "Start: ";
   cin >> Start;            //Let say user entered 8
   cout << "End: ";
   cin >> End;              //Let say user entered 15
                    // printing should begin from Start
   int i=Start;

   while(i <= End)
   {
      cout << Table << " x " << i << " = " << Table*i << endl;
      i = i + 1;
   }
   return 0;
}
```

Output

```
Which Table: 7
Start: 8
End: 15
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
7 x 11 = 77
7 x 12 = 84
7 x 13 = 91
7 x 14 = 98
7 x 15 = 105
```

Link

http://codepad.org/y0MJVVPb

Description

This example is slightly different from previous one in this program we take input from user about table, starting and ending, now the while loop starts from starting value and ends at ending value.At last you are able to use while loop, input from user, show anything on screen.

Now, you can use these concept and make anything you want which can be done by these.

Remember

Generic programming means that you can easily reuse that program without changing the code. In short user friendly program.

Program 16.1: Generic Table Printing (Using For loop)

Let us do the same program using **for** loop instead of **while**.

Code

```cpp
#include <iostream>
using namespace std;
//Table printing within the user entered limit(multiple)
/*example of for loop that print table according to
user input within given limits*/

int main()
{
   int Table;
   int Start, End;
   cout << "Which Table: ";
   cin >> Table;            //Let say user entered 7
   cout << "Start: ";
   cin >> Start;            //Let say user entered 8
   cout << "End: ";
   cin >> End;              //Let say user entered 15

   // printing should begin from Start till End
   for(int i=Start; i<=End; i++)
   {
      cout << Table << " x " << i << " = " << Table*i << endl;
   }
   return 0;
}
```

Output

```
Which Table: 7
Start: 8
End: 15
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
7 x 11 = 77
7 x 12 = 84
7 x 13 = 91
7 x 14 = 98
7 x 15 = 105
```

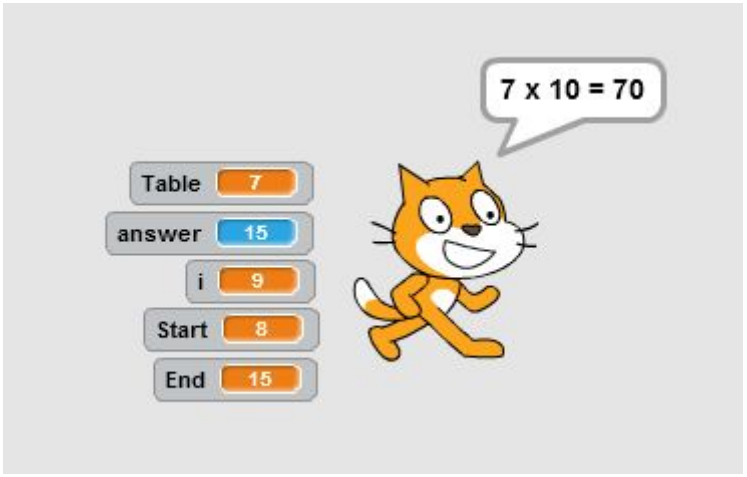
Link: http://codepad.org/vXlJ8y65

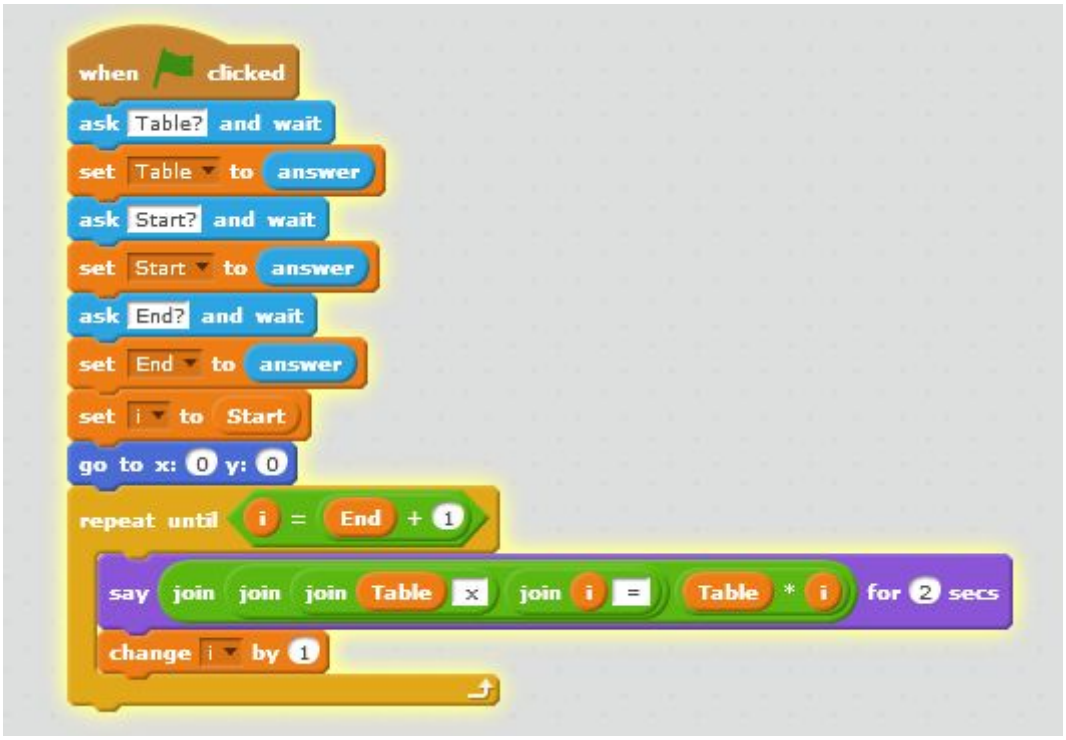
Description

It is the exact same Program 16 but the done using for loop.

For Those who did Chapter 2

This example of **scratch** is just like above **program 16 and 16.1** of while and for loop that prints the table within limits(multiples).


7 x 10 = 70
Table 7
answer 15
i 9
Start 8
End 15


when clicked
ask Table? and wait
set Table to answer
ask Start? and wait
set Start to answer
ask End? and wait
set End to answer
set i to Start
go to x: 0 y: 0
repeat until i = End + 1
say join join join Table x join i = Table * i for 2 secs
change i by 1