Chapter-1 Review of C++ Programming

Introduction to C++

C++ is an Object Oriented Programming Language(OOP) developed by Bjarne Stroustrup.It was early called C with Classes.It follows bottom- up approach.

Character Set

A character set is a set of characters a language can identify .It is also known as the alphabet of a programming language.Most C++ Compiler follows ASCII character set.They include letters,digits,special symbols,white spaces etc.

Tokens

Tokens are the fundamental building blocks of a program.They are also known as Lexical unit.There are five types of tokens in C++.They includes,

Keywords,Identifiers,Literals,Punctuators and Operators.

Keywords

Keywords convey a special meaning to the compiler.They are also known as reserved words.They are 63 keywords in C++.

Identifiers

An identifier refers to the name of variable,function,array,class etc.The identifier assigned to memory location is called a variable.The identifier assigned to a statement is called a label.

Rules for naming an identifier

1)Keywords cannot be used as identifier.

2)It must begin with a letter or underscore.

3)Special symbols should not be used (except underscore).

4)Upper case and lower case letters are treated separately(case sensitive).

5)only characters,digits and under score are allowed in the name of identifier.

Literals

Literals are data items whose value do not change during the execution of a program.They are also called constants.There are four types of constants in C++

Integer constants

Character constants

Floating point constant

String constant

Integer constants

An interger constant represents a whole number(ie , a number without fractional part).There are three types of integer constants,Decimal constant,Octal constant and Hexadecimal constant.

Character constant

A character constant is enclosed within single quotation mark.For example 'a','B' etc.

Floating point constant

Floating point constants are also know as Real constant.Floating point constants are represented in Mantissa and Exponent Parts.

String constant

A string constant is enclosed within double quotation mark.Every string in C++ is terminated by a Null character(\0).For example "abc","Object" etc.

Escape sequence

C++ allows certain characters that cannot be directly typed or entered from keyboard (Non-graphic characters) such as new line,back space ,.etc.These characters begins with backslash(\) and are called escape sequence.

Escape Sequences

Escape Sequences	*Character Constants*
\a	Alert (bell)
\b	Backspace
\f	Form feed
\n	Newline (Linefeed)
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
\0	Null
\'	Single quote
\"	Double quote
\\	Backslash
\?	Question mark
\C	Octal constant (C is a three-digit octal constant)
\xC	Hexadecimal constant (C is a three-digit hexadecimal constant)

Operator

An operator is a symbol used to perform a specific task.The data on which an operator act upon is called operand.Based on the number of operands operators are classified into three, Unary operator,Binary operator and Ternary operator.

A unary operator acts on a single operand.A binary operator acts on two operands.A ternary operator(? :) acts on three operands.

Based on the nature of operation operators are classified into three Arithmetic,Relational and Logical operator.

Arithmetic operators

Arithmetic operator are used for addition,substraction,multiplication and division(+,-,*,/).C++ provides a special operator modulus operator(%) for getting remainder of division operation.For example 10 % 2 returns 0.

Symbol	Meaning
*	Multiplication
/	Division and Integer Division
+	Addition
–	Subtraction
%	Modulus or Remainder

Relational operators

Relational operators are used to compare values.They produces only 'True' of 'False' values.The following are the relational operators in C++.

Operator	Meaning
<	*Less than*
>	*Greater than*
<=	*Less than or equal to*
>=	*Greater than or equal to*

Logical operator

Logical operators allows us to combine two or more conditions.They are often referred to as Boolean operators.The logical operators are &&(logical AND),||(logical OR),and !(logical NOT).

OPERATOR	MEANING
&&	AND
||	OR
!	Not

Type conversion

Type conversion means converting one data type to another data type.There are two types of type casting,**Implicit typeconversion**

Explicit typeconversion(Type casting)

Implicit type conversion

Implicit type conversion also known as automatic type conversion is performed by the compiler.The conversion is always from lower type to higher type,it is also known as ***type promotion***.

Example:

Int x=5.7 automatically changes to 5.

Explicit type conversion

Type casting refers to conversion that is performed explicitly using cast operator.

The operator used for this purpose is known as cast operator.The cast operator takes on the format,

(cast type) expression

or

cast type (expression)

For example :

cast type expression

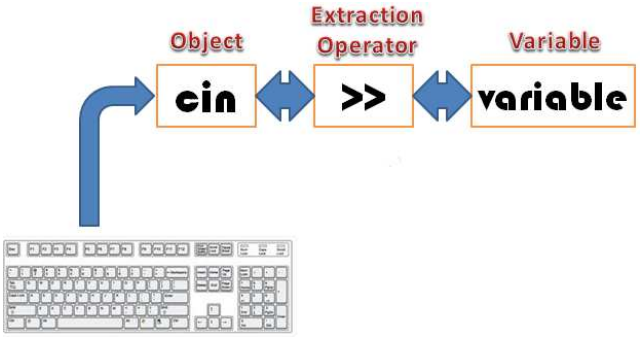
int a = (int) 10.5 ,Here the value 10.5 is converted to integer type.It can also be written as

Input-Output operators

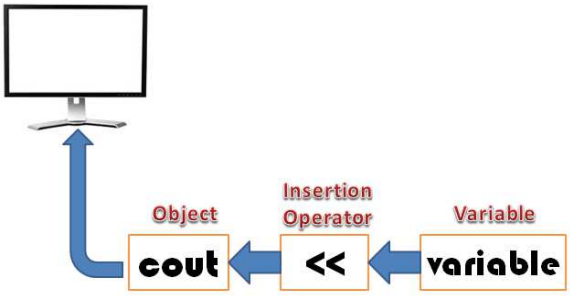
The Input-Output operators are used to take input and display output.The operator(**>>**) is used for taking input,and the operator(<<)is used to display the output.The **>>** operator is called **extraction** or **get from** operator.The operator **<<** is called **insertion** or **put to operator**.It is a binary operator.

Input and output statements

The input operator is used with standard input stream,cin and the output operator is used with standard output stream,cout.The input operator takes the value through cin and stores in a variable.



Cin >> number;



Cout << number+2;

Assignment operator (=)

The assignment operator is used to assign a value to a variable.

For example a=10 Assigns the value 10 to a

The symbol = assigns a value,where as the symbol == compare two values.

Expressions

An expression consists of operator and operand.There are different types of arithmetic expressions in C++:

Integral expression : All operands in the expressions are integers.An integral expression yields an integral result.

Floating point (decimal) expression: All operands in the expression are floating points(decimals).A floating point expression yields a floating point result.

Relational expression

The relational expression consists of relational operators.They are also called conditions.

Logical expression

A logical expression is an expression whose value is either True or False.

For example x>y is a logical expression,since it can be either 'True' or 'False'.

Cascading of I/O operators

The multiple use of input or output operators in a single statement is called cascading of I/O operators.

Assignment Statement

Assignment operator is represented by the symbol **'='**.The assignment operator takes the value on the right and stores it in the variable on the right side.

The assignment statement can be simple or compound type.

Compound Assignment statements use the following assignment operators :

*=

/=

%=

+=

/=

The following are the valid expression :

a+=b (equivalent to a=a+b)

a–=b (equivalent to a=a–b)

a*=*b* (equivalent to a=a*b)

a/=b (equivalent to a=a/b)

a%=b (equivalent to a=a%b)

program shows the use of compound assignments.

```
#include<iostream.h>
main()
{
int number=30;
number +=100;
cout<<endl<<number;
number –=10;
cout<<endl<<number;
number /=3;
cout<<endl<<number;
number *=4;
cout<<endl<<number;
number %=9;
cout<<endl<<number;
}
```

Output

```
130
120
40
160
7
```


C++ Shorthands

C++ shorthands are used to simplify the coding of assignment statements.

For example x=x+20; can be written as

X+=20;

Increment and Decrement Operators

In C++ there are two operators increment**(++)** and decrement **(--)**.The increment operator increases the value by one and the decrement operator decreases the value by one.

Prefix increment and decrement operator.

Postfix increment and decrement operator.

Prefix increment and decrement operator

The Prefix increment operator increases the value by one,whereas the pefix decrement operator decreases the value by one.Here incremented or decremented value is used for other operations.So this method is called change,then use method.

For example a=++b here the value of b is incremented by one and assigned to a.

Postfix increment and decrement operator

The Prefix increment operator also increases the value by one,

whereas the pefix decrement operator decreases the value by one.In the postfix notation the increment is done after the value is assigned. So this method is called use,then change method.

For example a=b++ here the value of b is assigned to a then the value of b is incremented by one.

Control Statements

Control statements are used to alter the normal sequence of execution of a program.They are classified into two types,

1. Decision Statements

2. Iteration Statements

Decision statements

The decision statements are based on a condition.They are also known as branching statements.The flow of control depends on the result of a particular condition.C++ supports three types of decision statements,

a) if statement
b) if – else statement
c) Switch statement

The if statement

The if statement is a single path statement, which executes statement only if the condition is true.The syntax is,

if(condition or expression)

statement1;

If the condition is true then the statement1 will be executed.

Example :**Program to demonstrate the use of if... statement.**

```
# include <iostream.h>
void main( )
{
 Int mark;
Cout <<"Enter your mark" ;
Cin >> mark ;
If (mark >=18)
   Cout <<"You have Passed " ;
```

}

In the above example the message You have passed will be displayed only when the condition ie, mark >=18 is true.

if – else statements

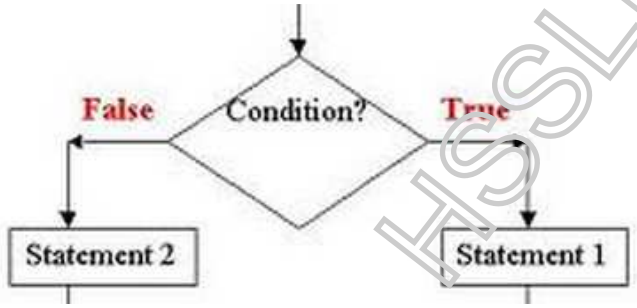
The if... else statement is used to test a logical condition and choose one of the alternatives based on the result of logical condition.The syntax of if – else statement is

If (condition or expression)

statement 1;

else

statement 2;



Example:**Program to Find largest of two numbers.**

```
# include <iostream.h>
Void main( )
{
Int a,b;
Cout << "Enter the numbers: " ;
```

```
Cin >> a >> b;
   If (a > b)
      Cout << "First number is largest: " ;
     else
    cout <<"Second number is largest: " ;
}
```

Example:**Program to find if a number is Even or Odd**

```
#include<iostream.h>
void main()
{
  int num,r;
 cout<<"Enter the number";
 cin>>num;
 r=num%2;
 if(r==0)
 cout<<"Number is Even";
 else
 cout<<"Number is Odd";
}
```


Nested if

An if – else statement can be placed inside another if – else statement,this is called nesting of if.

Example

If (Mark >=80)

{

If (age>18)

Cout<<"Eligible for higher studies" ;

}

else

{

cout <<"Not Eligible" ;

}

The else-if ladder

The ***else-if*** ladder helps to select one out of many alternative block of statements for execution depending on mutually exclusive conditions.The syntax of the **else... if** *ladder* is,

If (test-expression***)***

{

Statement 1;

}

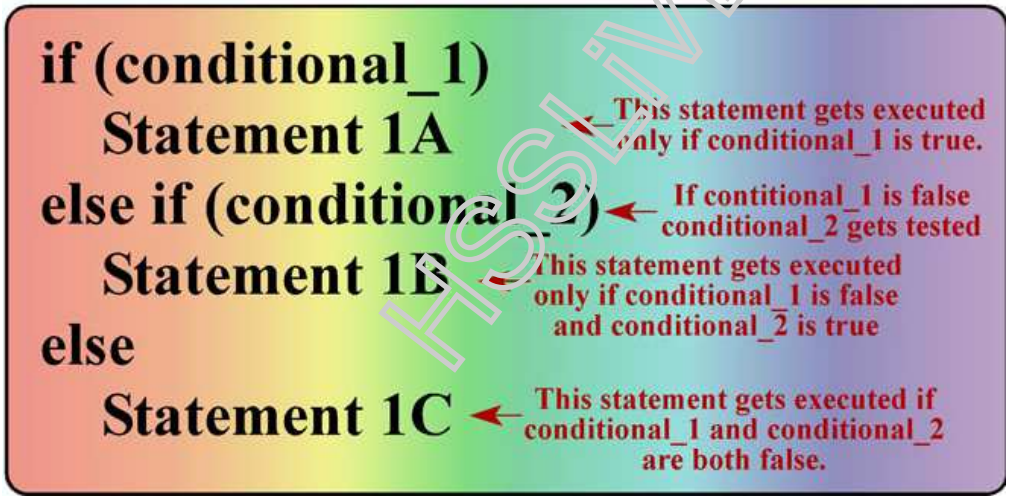
else if (test-expression)

{

Statement 2;

```
  }
else if (test-expression)
  {
    Statement 3;
  }
 else
{
Statements;
}
```

Working of if...else if



Example:Program to find Largest of three numbers

```
#include <iostream.h>
void main()
{
 Int a,b,c;
```

```
Cout<<"Enter the numbers" ;
Cin>>a>>b>>c;
If (a>b && a>c)
 {
Cout <<"The largest Number is a" ;
 }
If (b>c &&& b>a)
 {
Cout <<"The largest  Number is b" ;
 }
else
 {
Cout <<"The largest number is c" ;
 }
 }
```


Switch statement

The Switch enables to select one among several alternatives.It is a multi-path statement.The synatx for switch statement is

```
Switch (expression)
 {
  case value1 :Statement 1;break;
  case value2 :Statement 2;break;
```

case value3 :Statement 3;break;

default : Statement n;

}

The expression is evaluated and statements are executed according to the value .If no match is found,then default statement is executed.

Note :Switch can be used instead of multiple if else if statement.The expression in a switch statement cab be either integer or a character.

Example :Program to accept a digit and display it in words.

```
#include<iostream.h>
void main()
{
 int digit;
 cout <<"Enter a Number" ;
 cin >>digit;
  switch (digit)
   {
    case 0: cout<<"Zero";break;
    case 1: cout<<"One";break;
    case 2:cout<<"Two";break;
    case 3:cout<<"Three";break;
    case 4:cout<<"Four";break;
    case 5:cout<<"Five";break;
```

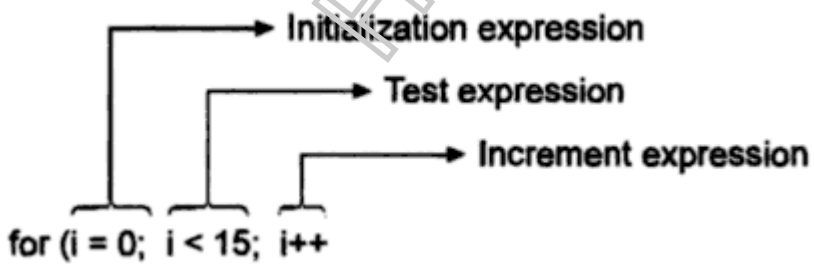
```
    case 6:cout<<"Six";break;
    case 7:cout<<"Seven";break;
    case 8:cout<<"Eight";break;
    case 9:cout<<"Nine";break;
   }
}
```

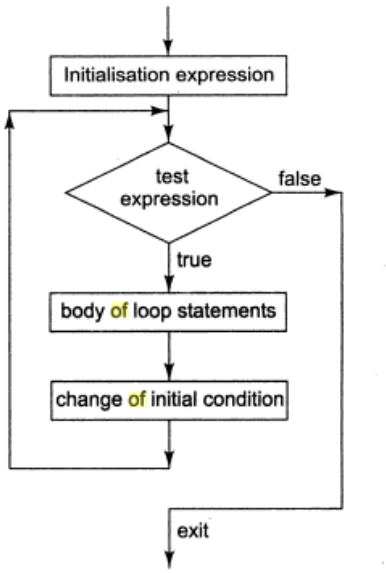
for statement

The for statement is the most commonly used statement in C++.It is an entry controlled loop.The syntax of for ... loop is

```
For(initialisation ;test expression ;updation)
    {
     Body of loop;
    }
```

Example : **for(i=0;i<15;i++);**





Example :Program to Print first 10 natural numbers

```
#include<iostream.h>
void main()
{ int i;
 for(i=0;i<10;i++)
 {
  cout<< i;
  cout<<" ";
 }
}
```


While Statement

The while ... loop is an entry controlled loop.The condition is checked first and if it is true the loop will be executed.The synatx is

```
While (expression)
```

```
    {
    Body of loop;
    Update expression;
   }
```

Example:Program to print first 10 even numbers

```
     #include<iostream.h>
void main()
  {
        int i;
        i=2;
       while(i<=10)
         {
            cout<<i;
               i=i+2;
           }
      }
```

do ... while loop

The do while loop executes statement before the expression is tested. The syntax is

do

{

Statements ;

}

While (expression)

Note: The do...while loop is an exit controlled loop in C++.

Difference between do .... while and while Loop

In the while loop the condition is tested and the body gets executed.

Whereas in do..while loop the condition is checked at the end of the loop.The do...while loop executes atleast one time even if the condition

is false.

Definition

The do-while loop is a post-test or bottom-test loop i.e, it executes its body at least once without testing specified conditional expression and then makes test. The do-while loop terminates when the text expression is evaluated to be 0.

Conditional Operator (?:)

The conditional operator is aternary operator(ie,it takes three operands).The general form is **Condition?true-part:false-part;**

The condition is a boolean expression.If it is true the result is the true-part,else the result is the false-part .

Note:The conditional *Operator can be used as an alternative to if... else.*

Nested Loop

A loop placed inside another loop is called a nested loop and the process of placing a loop inside another loop is called nesting.In a nested loop the inner loop will be executed as long as the condition in the outer loop is True.

Example:

```
for(i=0;i<=5;i++) // outer loop
  {
  for(j=0;j<=5;j++) // inner loop
     {
cout<<i<<"\t"<<j;
}
}
```

Explanation of the program :

In the outer for loop when the value of i becomes 0 then control enters into the inner loop where j will take value from 0 to 5 considering the same value of i = 0

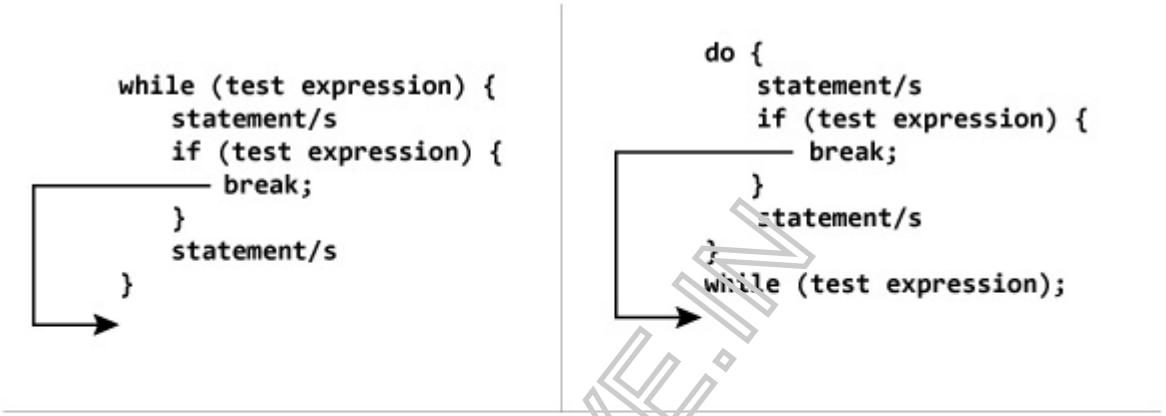
Value of i	Value of j in inner loop
i = 0	j = 0 1 2 3 4 5
i = 1	j = 0 1 2 3 4 5
i = 2	j = 0 1 2 3 4 5
i = 3	j = 0 1 2 3 4 5
i = 4	j = 0 1 2 3 4 5
i = 5	j = 0 1 2 3 4 5

Jump statements

Jump statements are used to jump unconditionally to a different statement.It is used to alter the flow of control unconditionally.There are four types of jump statements in C++,break,continue,goto and return.

Break

The break statement is used to terminate a loop or switch statement.



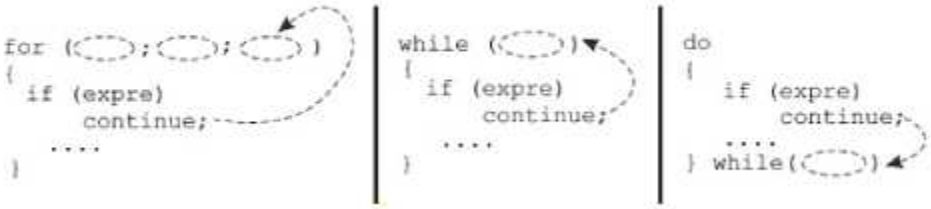
Note

*The **break** statement exits only in the most current loop. If you have a program with a **while** loop that is within another **while** loop, **break** exits only the internal loop.*

Continue

A continue statement is used to continue to the beginning of a loop.When a continue statement is executed in a loop it skips the remaining statements in the loop and proceeds with the next iteration of the loop.In a while and do.. while loop,the condition is evaluated immediately after the

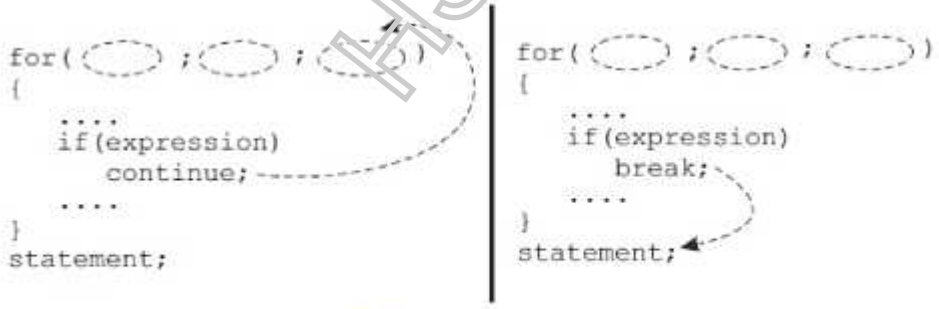
continue statement.In a for loop ,the update statement is executed after the continue statement.



For example:

```
for(i=0;i<=10;i++)
{
    statement 1;
    if(i==5)
    continue;
    statement 2;
}
```

In the above example, when the value of the variable **i** is equal to five, the **continue** statement will be executed. The **statement 2** will be skipped and the control will be transferred back to the increment part of the **for** statement for the next iteration.



Control flow for continue and break

Goto

A goto statement is used for unconditional jump.It transfers the control from one part of the program to another.The syntax is,

Goto label;

A label is an identifier followed by a colon.