Control Structures

A program is usually not limited to a linear sequence of instructions. During its process it may repeat code or take decisions. For that purpose, C++ provides control structures : selection and Iteration, that serve to specify what has be done by our program, when and under which situation.

Compound (Block of) Statements

With the introduction of control structures we are going to have to introduce a new concept: the ***compound statement*** Or ***block***.

A block is a group of statements which are separated by semicolons (;) , like all C++ statements, but grouped together in a block enclosed in braces: { }:

{ statement1; statement2; statement3; }

- A statement can be either a simple statement (a simple instruction ending with a semicolon) or a compound statement (several instructions grouped in a block).
- In the case that we want then statement to be a simple statement, we do not need to enclose it in braces ({}), but in the case that we want the statement to be a compound statement it must be enclosed between braces ({}), forming a block.

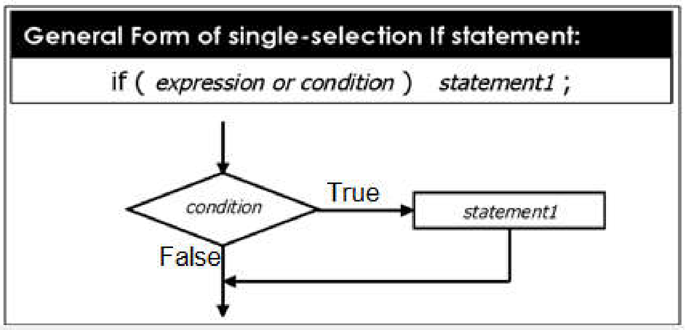
Example1	**Example2**

`{ int a, b ;`
`a= 10 ;`
`b= a % 3;`
`}`
	int x, y;
	`{`
	`cout<<"enter x, y :"`
	`cin>> x>>y ;`
	`int z;`
	`z=x; x=y; y=z;`
	`}`

Selection

single IF statement structure

The **If** statement is used to express conditional expression. If the given condition is true then it will execute the statements; otherwise it will execute the optional statements .



General Form of single-selection If statement:

if (*expression or condition*) *statement1* ;

Note: The single Block IF statement are enclosed in { } to group declaration and statements into a compound statement or a block. These blocks are always considered as a single statement.

General Form of single block selection if statement:

```
if ( condition )
    {  statement1 ;
       statement2 ;
       statement3 ;
          .
          .
    }
```

Example1:
`If (avrg >= 3.5)`
`  cout << "good";`
	Example2:
	`if (deg1 >= 50 && deg1 <= 59)`
	`  cout << "m" << endl;`

Example3:
`cin >> ch ;`
`if ( ch == 'y' || ch == 'Y')`
`{`
`    z= x+10 ;`
`    cout << "z=" << z << endl;`
`}`
	Example4:
	`if (a !=b && c == 6 )`
	`{`
	`  int x;`
	`  cout << "Enter x:";`
	`  cin >> x;`
	`  if (x >=0)`
	`    cout << "positive :" << endl;`
	`}`

Exercises:

1. Write a C++ program to read any two numbers and print the largest value.

2. Write a C++ program that found the z value where z=x / 10 , x >=6.

3. Write a C++ program to read a number and check if its positive , if it so print it and decrement it by 2 then print it after that.

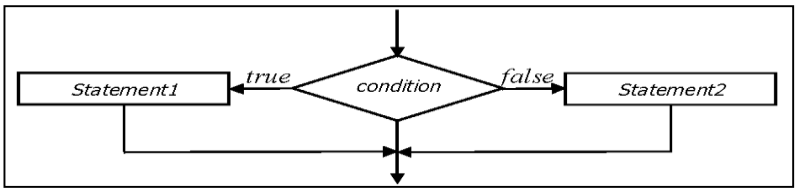
4. Write a C++ program to read a number and check if it's Odd or Even , if it odd so print the number and its square , when it Even print the number and its cubic.

if /else statement structure

General Form Of if /else statement	

If (condition)
statement1 ;
else
statement2 ;
	If (condition)
	{ statements }
	else
	{ statements }

In this case, either of the two statements are executed depending upon the value of the expression. there is a semicolon after each of the statement but not after the IF-expression.



<u>Example 1:</u>

```
cin >> value;
if ( value >= 0 )
        cout << "positive";
else
        cout << "negative";
```

<u>Example 2:</u>

```
cin >> num1 >> num2;
if ( num1 > num2 )
        cout << num1;
else
        cout << num2;
```

Exercises:

1. Write a c++ program to read a number, and check if it's <u>positive</u> or <u>negative</u>.

2. Write a c++ program to read a number, and check if it's <u>odd</u> or <u>even</u>.

3. Write a C++ program to read a student degree , and check if it's degree greater than or equal to 50 , then print <u>pass</u> , otherwise print <u>fail</u> .

Else IF Statements Structure

General Form of else if statement:

```
if ( expression or condition 1 )   statement1 ;
else  if ( expression or condition 2 )  statement2 ;
else  if ( expression or condition 3 )  statement3 ;
:
else  if ( expression or condition n )  statement-n ;
else  statement-e ;
```

<u>Example 1:</u>

```
if ( value == 0 )    cout << "grade is A";
else if ( value == 1 )    cout << "grade is B";
else if ( value == 2 )    cout << "grade is C";
else  cout << "grade is X";
```

Exercises:

1. **Write a C++ program to read a umber , and print the day of the week .**

2. Write a C++ program to compute the value of Z according to the following equations:

$$Z = \begin{cases} x+5 & : x<0 \\ \cos(x)+4 & : x=0 \\ \sqrt{x} & : x>0 \end{cases}$$

Nested IF statements Structures

Some of the samples of Nested if-else constructions are shown below:

```
If (exp.)
  {
     Statements
   }
Else
 {
     Statements
   }
```

```
If (exp.)
  { if (exp.)
     { Statements }
    Else
     { Statements }
  }
Else
  { Statements }
```

```
If (exp.)
  { if (exp.)
     { Statements }
    Else
     { Statements }
  }
Else
  { if (exp.)
     { Statements }
    Else
     { Statements } }
```

Example: C++ program to find a largest value among three numbers.

```
# include <iostream.h>
  float x , y , z;
  void main () {
    cout << "Enter numbers : \n ";
    cin >> x >> y >> z;
    if ( x > y )
      {
         if (x > z)
           cout << " The largest value is  " << x << endl;
         else
             cout << " The largest value is  " << z << endl;
      }
   else if ( y > z )
      cout << " The largest value is " << y << endl;
   else
     cout << " The largest value is " << z << endl;
}
```

Switch selection statement (Selector)

The switch statement is a special multi way decision maker that tests whether an expression matches one of the number of constant values, and braces accordingly.

General Form of Switch Selection statement:

```
switch ( selector )
{
    case label1  :  statement1 ;  break;
    case label2  :  statement2 ;  break;
    case label3  :  statement3 ;  break;
       :
    case label-n  :  statement-n ; break;
    default  :  statement-e ;   break;
}
```

Example 1: Write C++ program to read integer number, and print the name of the day in a week:

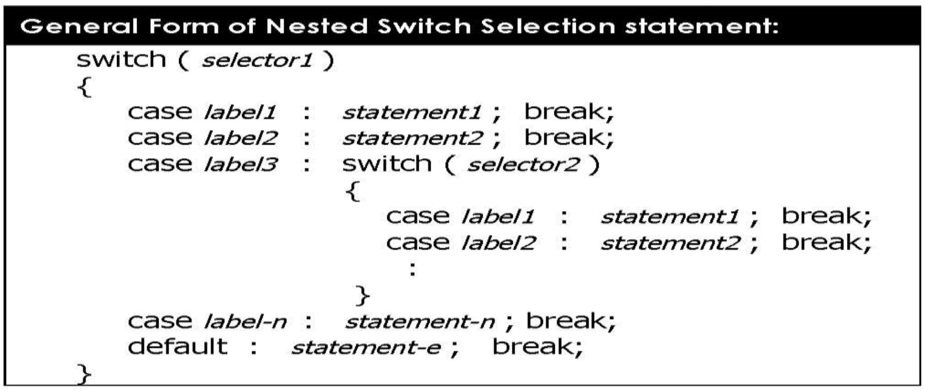
```
# include <iostream.h>
     int day;
   void main ()
   {    cout << "Enter day number : \n ";
        cin >> day;
        switch ( day )
         {
          case 1:  cout << "Sunday"<< endl;         break;
          case 2:  cout << "Monday"<< endl;        break;
          case 3:  cout << "Tuesday"<< endl;       break;
          case 4:  cout << "Wednesday"<< endl;  break;
          case 5:  cout << "Thursday"<< endl;      break;
          case 6:  cout << "Friday"<< endl;           break;
          case 7:  cout << "Saturday"<< endl;       break;
         default: cout << "Invalid day number"<< endl; break;
         }
}
```

Example2: Write C++ program to read two integer numbers and read the operation code number(1 for addition, 2 for subtraction, 3 for product and 4 for division) to perform on these numbers.

Example3: Write C++ program to read two integer numbers and read the operation (+ , -, * or /) to perform it on these numbers.

```
    # include <iostream.h>
        int a , b;
        char x ;
      void main ()
       {
        cout << “Enter two numbers : \n ”;
        cin >> a >> b;
        cout << “ + for addition \n “;
        cout << “ - for subtraction \n “;
        cout << “ * for multiplication \n “;
        cout << “ /  for division \n “;
        cout << “ enter your choice \n”;
        cin >> x;
        switch ( x )
         {
          case ‘+’:  cout << a+b;        break;
          case ‘-’:   cout << a-b;        break;
          case ‘*’:  cout << a*b;        break;
          case ‘/’:  {
                      if (b != 0)
                        cout << a/b;
                      else
                        cout << “ Error.. Division by Zero”;
                      break;
                    }
            default: break;
          }
  }
```

Nested Switch selection statement



General Form of Nested Switch Selection statement:

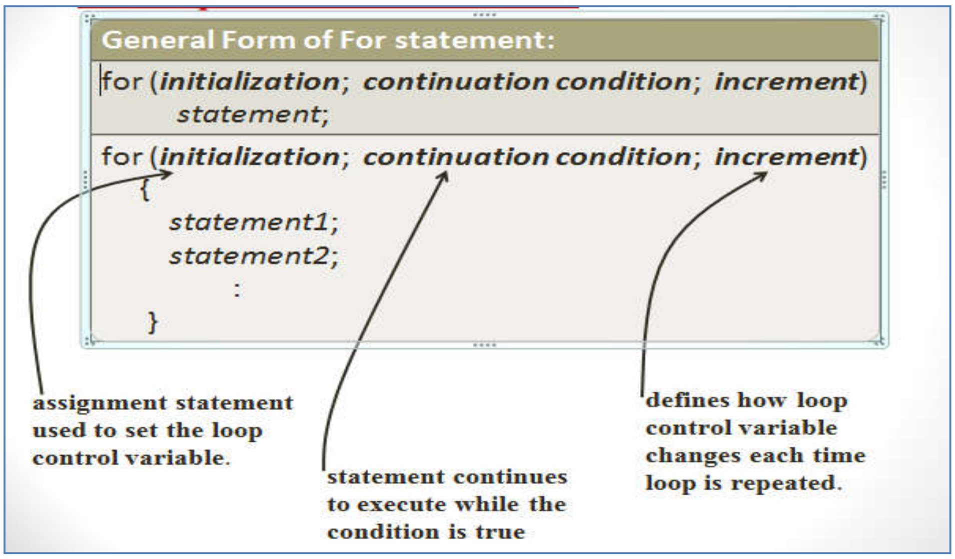
```
switch ( selector1 )
{
    case label1   :   statement1 ;  break;
    case label2   :   statement2 ;  break;
    case label3   :   switch ( selector2 )
                      {
                          case label1   :   statement1 ;   break;
                          case label2   :   statement2 ;   break;
                            :
                       }
    case label-n  :   statement-n ; break;
    default   :   statement-e ;    break;
}
```

Loop Statements (Repetition)

The loop statements are essential to construct systematic block style programming C++ provides three iteration structures:

- For loop statement
- While loop statement
- do/while loop statement

For loop statement structure


General Form of For statement:

```
for (initialization; continuation condition; increment)
        statement;
```

```
for (initialization; continuation condition; increment)
    {
        statement1;
        statement2;
             :
    }
```

assignment statement used to set the loop control variable.
statement continues to execute while the condition is true
defines how loop control variable changes each time loop is repeated.

Examples	Output

int i;
for (i=0 ; i <10; i ++)
cout<< i;
	0123456789

for (int j=0 ; j <10; j +=2)
cout<< j<<'\t';
	0 2 4 6 8

int i;
for (i=1 ; i <10; i +=2)
cout<< I<<endl ;
	1
	3
	5
	7
	9

for (int i=10 ; I >0 ; i --)
cout << i<<" ";
	10 9 8 7 6 5 4 3 2 1

for (int i= 8 ; i>=2 ; i-=2)
cout << i <<'\n';
	8
	6
	4
	2

Examples :

1. Write C++ program to add the numbers between 1 and 10
2. Write C++ program to add even numbers between 1 and 10
3. Write a program that read integer number and check if it prime or not prime
4. Write a program to read integer and find its factorial where
5. n!= 1*2*….* n
6. Write a program that find max number between N integer numbers
7. Write a program to find the minimum mark among many positive integer numbers
8. Write a program to prints the summation of the following series(for n terms)

$$s = \frac{1}{2} + \frac{2}{4} + \frac{3}{6} + \frac{4}{8} + \cdots \qquad + \frac{10}{12}$$

9. Write a program to prints the summation of the following series(for n terms)

$$s = \frac{1}{3} + \frac{2}{5} + \frac{3}{7} + \frac{4}{9} + \cdots \qquad + \frac{10}{21}$$

10. Write a program to prints the summation of the following series(for n terms)

$$s = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} \cdots$$

11. Write a program to generate and print the following series(for n terms)

0 1 1 2 3 5 8 13 21 …….

Although you can use any loop statement to create an infinite loop, **for** is traditionally used for this purpose. Since none of the three expressions that form the **for** loop are required, you can make an endless loop by leaving the conditional expression empty:

```
    for( ; ; )
```

Actually, the **for(;;)** construct does not guarantee an infinite loop because a **break** statement, encountered anywhere inside the body of a loop, causes immediate termination. **When the conditional expression is absent, it is assumed to be true.**

```
// Expample    using infinite loop  continue entering character until press S or s
#include <iostream.h>
 char ch;
 void main( )
  {
    for (  ;  ; )
      {
        cout <<" Enter any char to continue or Press S or s to stop:"<<endl;
        cout<<"Enterthe Char :";
        cin>>ch;
        if (ch=='s' || ch=='S')
           break;
      }
  }
```

Nested for Loop

A for loop can be nested inside of another loop. C++ allows at least 256 levels of nesting. The syntax for a **nested for loop** statement in C++ is as follows:

```
for ( init; condition; increment )
   {
      for ( init; condition; increment )
         {
            statement(s);
         }
       statement(s); // you can put more statements.
   }
```