Programming Language

Using C++

control structure of C++ program

Lecture #10

Characters with special purposes

There are some letters used with "\" for a specific purpose, as shown in the following table:

character	Purpose
endl	Indicates the end of writing on the current line
\n	Indicates the end of writing on the current line and moving to a new line
\t	Indicates making a space (8 spaces by pressing the ruler)
\a	Indicates the sound of a beep during program execution
\b	Indicates going back one space after execution
\'	Indicates printing a single quote.
\r	Indicates to return the cursor to the beginning of the line and resume printing
\f	Indicates to leave a blank page and continue printing

control structure of C++ program

Three methods of processing a program (controlling the structure of program execution)

- ✓ **In sequence (sequentially)**
- ✓ **Looping**
- ✓ **Branching**

Loop: Altering the flow of program execution by repetition of a particular block of statement(s).

Branch: Altering the flow of program execution by making a selection or choice.

What kind of loop

- ✓ When you know how many time you want to execute the block of code use a counting loop
- ➢ A counting while loop, you set up increment and test your counter
- ➢ A for loop, give the critical values the for loop.
- ➢ for loop sets up, increments and tests for you
- ✓ When you have no idea how many times the loop will be executed.

Basic Loops

- When one action is to be repeated a number of times a loop is used. There are two common types of loops:

✓ **while loop or do...while loop**

- Used to continue repetition while a condition holds
- Can also be used to repeat a particular number of times

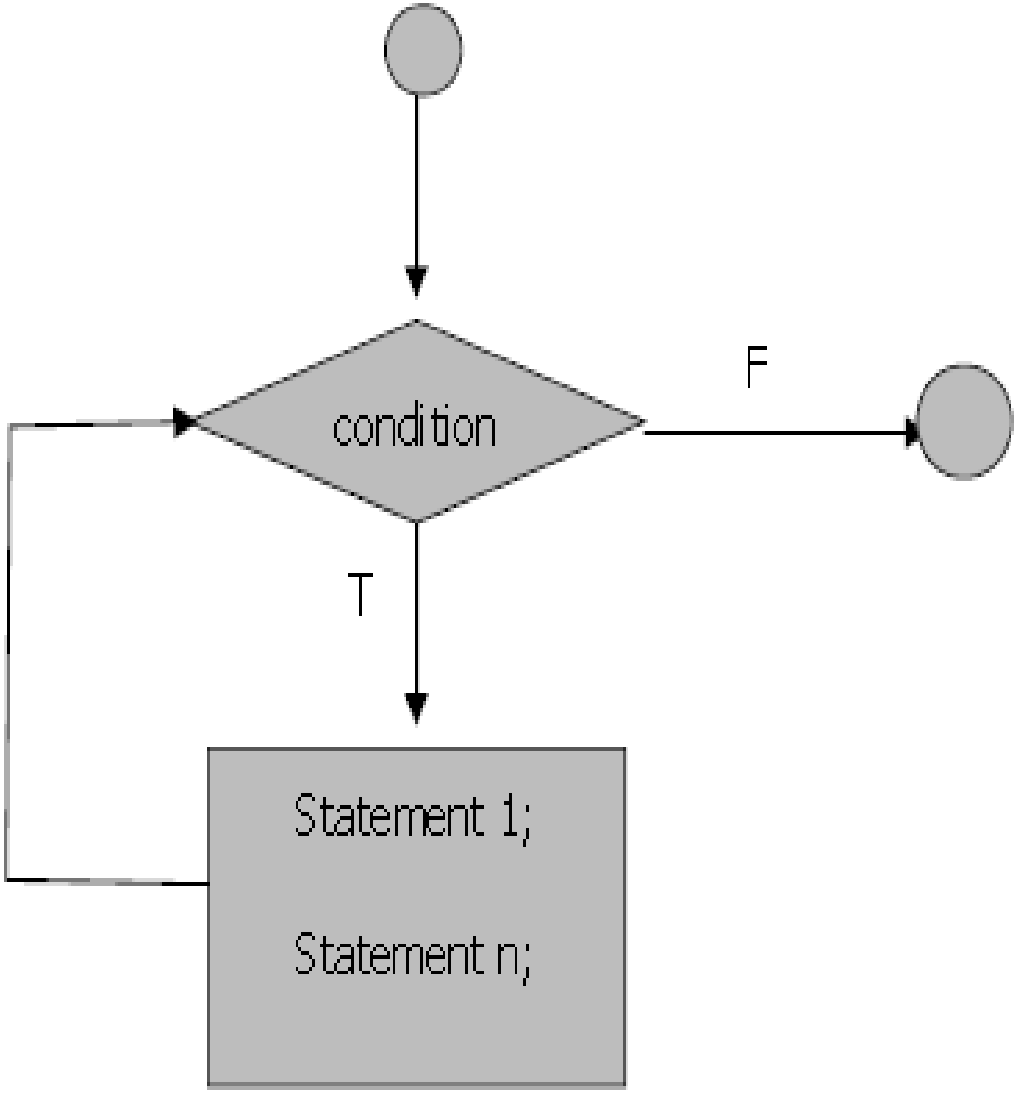
✓ **for loop**

- Specifically designed to repeat a particular number of times
- Loop will stop executing when some condition becomes false.

while Loop

```
while (condition) {
    // Series of actions to be taken
    // each time the loop is executed
    // loop is executed when condition is True
    Action(s);
}
// When condition is false execute following actions
Actions(s);
```

Structure of while loop : flowchart



Example1: while Loop

```
// Sum the Integers from 1 to 25 inclusive
#include <iostream>
using namespace std;
int main( ){
        int x = 1;
        int sum = 0;
        while (x <= 25)
        {
                sum += x;
                x++;
        }
        cout << sum;
        return 0;
}
```

Example2: while Loop

```
// Class average program with counter-controlled repetition.
#include <iostream>
using namespace std;
int main()
   {
      int total;        // sum of grades input by user
      int gradeCounter; // number of grade to be entered next
      int grade;        // grade value
      int average;      // average of grades
// initialization phase
     total = 0;         // initialize total
     gradeCounter = 1; // initialize loop counter
// processing phase
      while (gradeCounter <= 10) {       // loop 10 times
           cout << "Enter grade: ";          // prompt for input
           cin >> grade;                    // read grade from user
            total = total + grade;          // add grade to total
            gradeCounter = gradeCounter + 1;  // increment counter
    } // termination phase
```

The counter gets incremented each time the loop executes. Eventually, the counter causes the loop to end.

```
   average = total / 10;   // integer division
// display result
   cout << "Class average is " << average << endl;
   return 0;   // indicate program ended successfully
   } // end function main
```

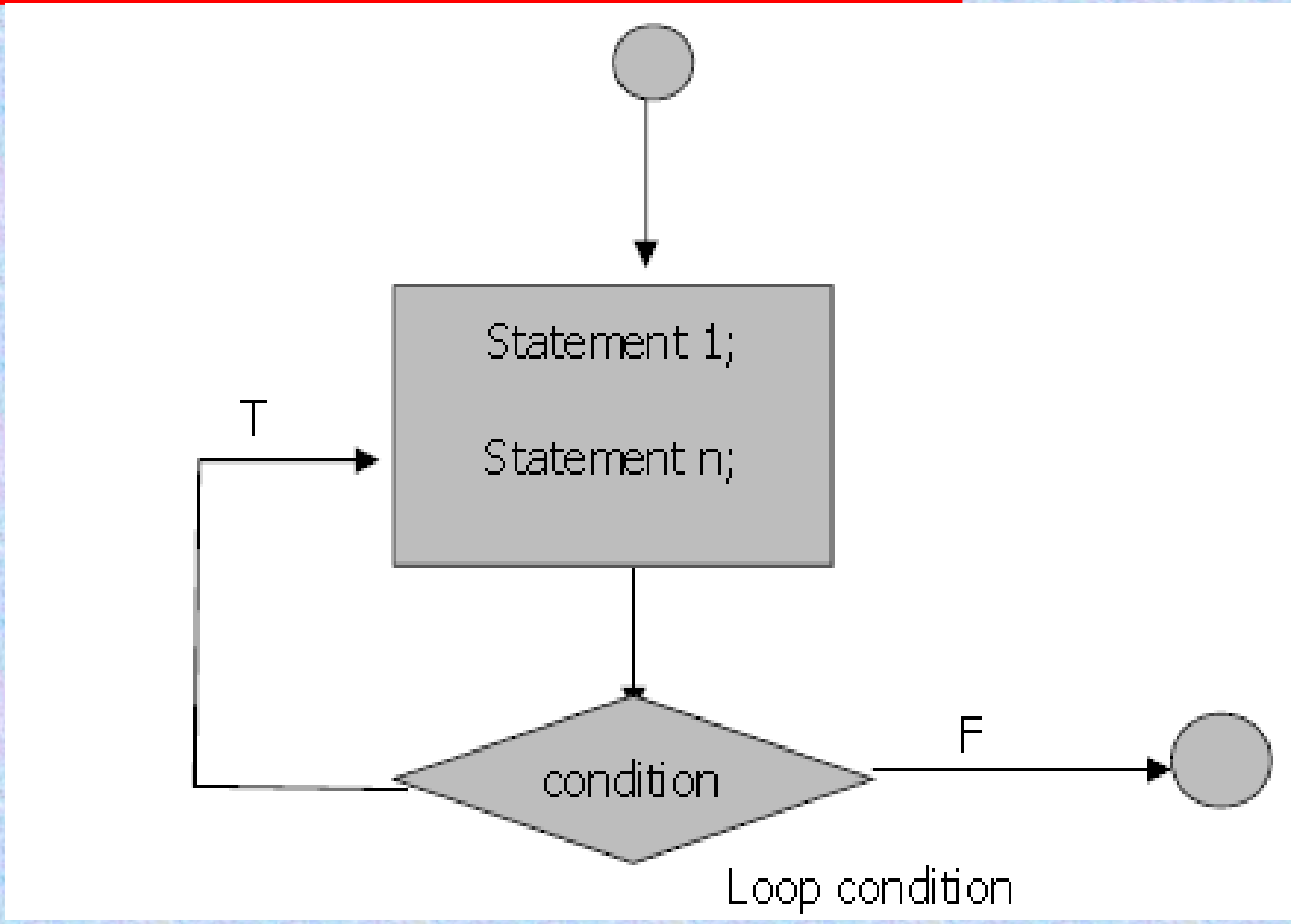
// running of program

```
Enter grade: 98
Enter grade: 76
Enter grade: 71
Enter grade: 87
Enter grade: 83
Enter grade: 90
Enter grade: 57
Enter grade: 79
Enter grade: 82
Enter grade: 94
Class average is 81
```

do…while Loop in C++

```
do
{
// Series of actions to be taken
// each time the loop is executed
// Always executed on first pass
// Subsequently executed if condition is true
Action(s) ;
} while (condition);
// When condition is false execute following actions
actions;
```

Structure of do…while Loop



Example do…while Loop

```
// Sum the Integers from 1 to 25 inclusive
int x = 1;
int sum = 0;
do
{
   sum += x;
   x++;
} while (x <= 25);
cout << sum;
```

Differences between *do…while* and *while*

```
// Sum Integers from n to m
n = 4;
m = 3;
sum = 0;
do
{
        sum += n;
        n++;
}
while ( n <= m );
cout << sum;
```

After code sum = 4
Body of loop executes once

```
// Sum Integers from n to m
n = 4;
m = 3;
sum = 0;
while ( n <= m )
{
        sum += n;
        n++;
}
cout << sum;
```

After code sum = 0
Body of loop does not execute

for Loop

for (*initial statement; loop condition; update statement*)

{

//Series of actions to be taken

//each time the loop is executed

Action(s);

}

- **A for loop is a counting loop, you must know how many times the loop will be executed**

Example for Loop

```
// program to read 11 random integer numbers and determines their sum
#include <iostream>
using namespace std;
int main( ){
        int sum = 0; int number;
        for (int i = 1 ; i <= 11 ; i++){// Loop reads 11 integers and determines their sum
                cout << "enter an integer ";
                cin >> number;
                sum += number;
        }
        cout <<"the sum of 11 integers is =: "<<sum<<end;
        return o;
}
```

The nested for Statement

The *Nested for* statement used to allow processing multiple loops .

The general syntax of the *nested for* statement can be expressed as:

```
for (initialization1; loop cond1; update1)
    {
      for (initialization2; loop cond2; update2)
          {
            for (initialization3; loop cond3; update3)
                {
                  :
                  statement(s)
                }
          }
}
```

The *initialization (1,2,3,…)* expressions initialize the loops; initiliazation1 executed one once when the first loop (outer loop) begins, other initializations (inner loop) executed every time when the outer loop checked.

The *loop cond (1,2,3,…)* expression terminate the loops when it is evaluates to false.

The *update (1,2,3,…)* expressions invoked after each iteration through the loops; it is acceptable for this expression to *increment* or *decrement* a value of the loop variable.

Example

Write C++ for the multiplication tables (1-12).

```
#include <iostream>
using namespace std;
void main( ) {
    for (int i=1; i<=12; i++) {
        cout<<"Table  " << i<<endl;
        for (int j=1; j<=12; j++) {
            cout<<i < "  ×  " << j<< " =  "<<i*j<<endl; }
        cout<<"\n";
        }
    }
}
```

Break

Basic Branches

Ability to control whether a statement list is executed

- **There are two common types of loops**

✓ **if statement**

- **if**
- **if-else**
- **if-else-if**

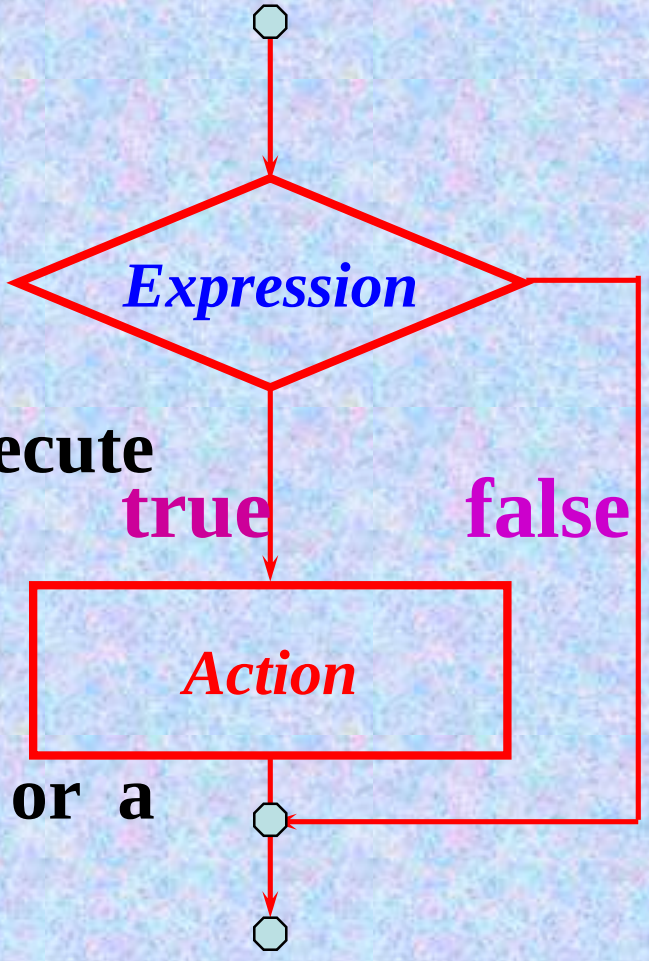
✓ **switch statement**

The Basic If Statement

- **Syntax**

  if (*Expression*)

  Action



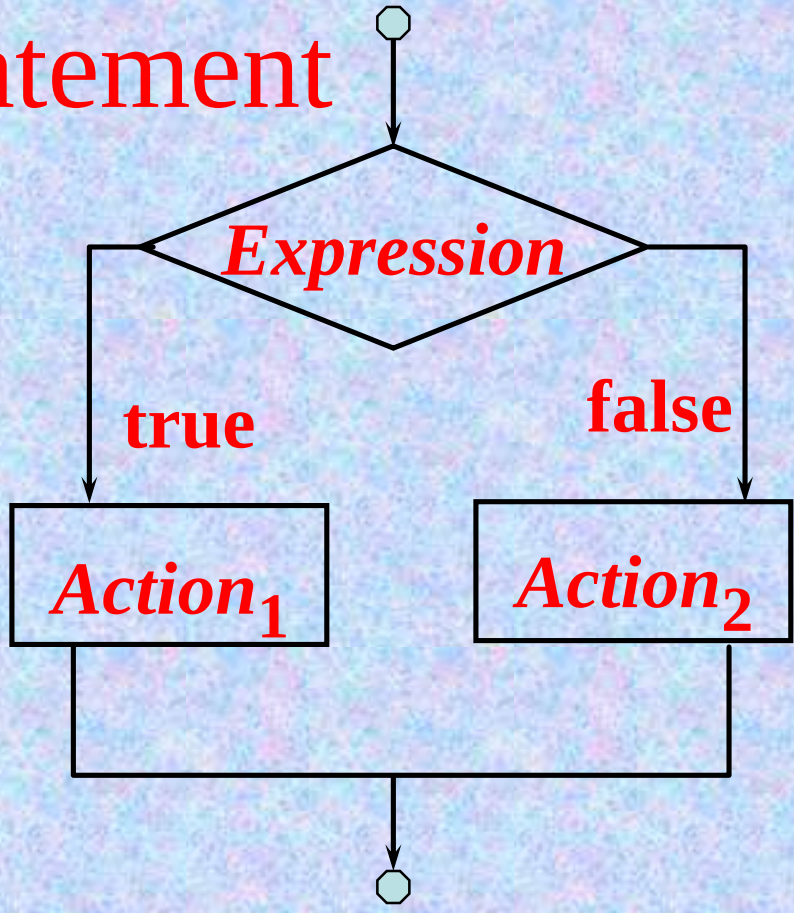
- **If the *Expression* is true then execute *Action***
- ***Action* is either a single statement or a group of statements within braces**

```
// program to sort two numbers ascendingly
#include <iostream>
using namespace std;
int main( ) {
    int  value1, value2, remembervalue1;
    cout << "Enter two integers: ";
    cin >> Value1 >> Value2;
    if (Value1 > Value2) {
        Value1 = Value2;
        Value2 = rememberValue1;
    }
    cout << "The input in sorted order: “   << Value1 << " " << Value2 << endl;
    return 0;
}
```

The If-Else Statement

- **Syntax**

```
if (Expression)
   Action1
else
   Action2
```



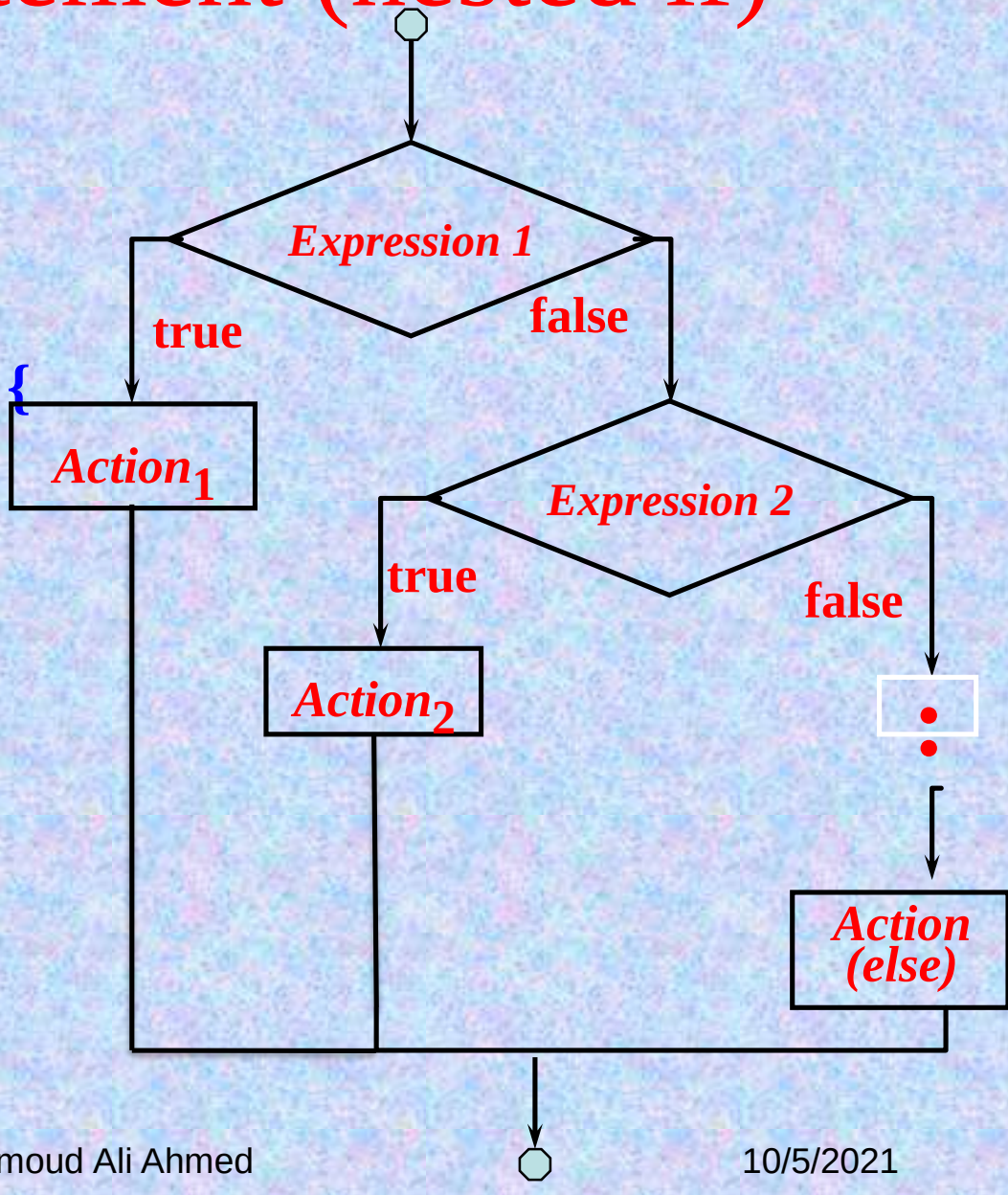
- **If *Expression* is true then execute *Action1* otherwise execute *Action2***

```
if (v == 0) {
 cout << "v is 0";
}
 else {
  cout << "v is not 0";
 }
```

```
//Program to finding the Max of two integers
#include <iostream>
using namespace std;
int main( ) {
    int Value1, Value2, max;
    cout << "Enter two integers: ";
    cin >> Value1 >> Value2;
    if (Value1 < Value2) {
            max = Value2;
    }
    else {
            max = Value1;
    }
    cout << "Maximum of inputs is: " << max << endl;
    return 0;
}
```

if-else-if Statement (nested if)

```
if (Expression1/condition1){
    actions(1);
}
else if (Expression2/condition2) {
    actions(2);
}
:
else {
    actions(else);
}
```



```
//program to determine whether the number is neg, pos, or zero
include <iostream>
using namespace std;
int main( ) {
    int nbr;
    cin>>nbr;
    if ( nbr < 0 ){
        cout << nbr << " is negative" << endl;
    }
    else if ( nbr > 0 ) {

        cout << nbr << " is positive" << endl;
    }
    else {

        cout << nbr << " is zero" << endl;
    }
    return 0;
}
```

Switch Statement

General Syntax:

```
switch (variable) {
   case value1:
                //action(s); break;
   case value2:
                //action(s); break;
   :
   case valuen:
                 //action(s); break;
   default:
        //action(s); }
```

```
//program to test whether the letter is vowel letter or not
include <iostream>
using namespace std;
int main( ) {
    char ch;
    cin>>ch;
    switch (ch) {
        case 'a': case 'A':
        case 'e': case 'E':
        case 'i': case 'I':
        case 'o': case 'O':
        case 'u': case 'U':
                cout << variable << " is a vowel" << endl;
                break;
        default:
                cout << ch << " is not a vowel" << endl;
    }
    return 0;
}
```

```
//program to carry mathematical operations
include <iostream>
using namespace std;
int main( ) {
    int Left,  Right;
    char Operator;
    cout << "Enter simple expression: ";
    cin >> Left >> Operator >> Right;
    cout << Left << " " << Operator << " " << Right  << " = ";
    switch (Operator) {
        case '+' : cout << Left + Right << endl; break;
        case '-' : cout << Left - Right << endl; break;
        case '*' : cout << Left * Right << endl; break;
        case '/' : cout << Left / Right << endl; break;
        default: cout << "Illegal operation" << endl;
}
return 0;
}
```

Quiz # 2

Write C++ program to swap any two integer numbers.

Modify your program if the swapping need to carried for ten pairs of integers numbers.

```
#include <iostream>
using namespace std;
int main(){
	int a, b, temp;
	cout<<"Enter values of a and b:\t“;
	cin>>a>>b;
	cout << “\n Before swapping." << endl;
	cout << "a = " << a << ", b = " << b << endl;
	temp = a;	a = b;	b = temp;
	cout << "\n After swapping." << endl;
	cout << "a = " << a << ", b = " << b << endl;
  return 0;
}
```