PROGRAMMING FUNDAMENTALS

Let's explore technology together to live in the future

Checkout more on https://github.com/Sy-hash-collab

Sy-hash-collab

'if' Statement:

'if' is a key word in C++ languag

- The if Statement is primary slection control str.
- It is used to execute or skip a statement or set of statements.

=> The condition is given as relational expression.

Syntax:

```
if (condition)
    statement;
```

The syntax for compound statement is as;

```
if (condition)
{
   statement 1;
   statement 2;
   :
   statement N;
}
```

Program

```
# include <iostream>
using namespace std;
int main()
{
 int marks;
 cout << "Enter your marks:";
 cin >> marks;
 if (marks >= 40)
{cout << " You have passed";
 }
```

Output:

```
Enter your marks: 50
 You have passed.
```

'if else' Statement:

- Used to make two way decisions.

=> It executes one block of statement when the condition is true and other when it is false.

- In any situation, one block is executed and other is skipped.

Syntax:

```
          if (condition)
              Statement;
          else
              statement;
```

e.g Write program that inputs a year and finds whether it is a leap year or not using if-else.

```
  int main ()
{
  int y
  cout <<"Enter a year";
  cin>>y
  if (y%4==0)
   {cout <<y<<"is a leap year";
    }
  else {cout<<y<<"is not a leap year";
       }
  }
```

Multiple 'if else if' Structure:

It is used when there are many options and only one block of statements should be executed on the basis of a condition.

Syntax:

```
          if (condition)
          { block1;
           }
          else if (condition)
          {block2;
           }
          else
          { block N
           }
```

Q) Write a program that inputs radius and user's choic
It calculates area of circle if user enters 1,
circumference if user enters 2 as choice.

```
 int main ()
{
  float area, radius, circumference;
  int choice
  cout <<"Enter radius:";   cin>>radius;
  cout <<" Enter 1 for area and 2 for circumference'
   cin>> choice;
   if (choice==1)
   { area = radius*radius* 3.14
     cout << "Area:" <<area;
    }
   else if (choice==2)
    {
      circumference = 2.0*3.14*radius;
     cout <<"Circumference:" << circumference;
    }
     else
     { cout <<"Invalid choice";
      }
    }
```

Nested 'if' Structure:

An if statement within an if statement is calle
nested if statement.
In nested structure, the control enters into
inner if only when outer condition is true.

Syntax:

```
                    if (condition)
                            if (condition)
Outer If →                  {
                                    statement;
                            }
                            else
Inner if      →             {
                                    statement;
                            }
                    else
                    {
                            statement;
                    }
```

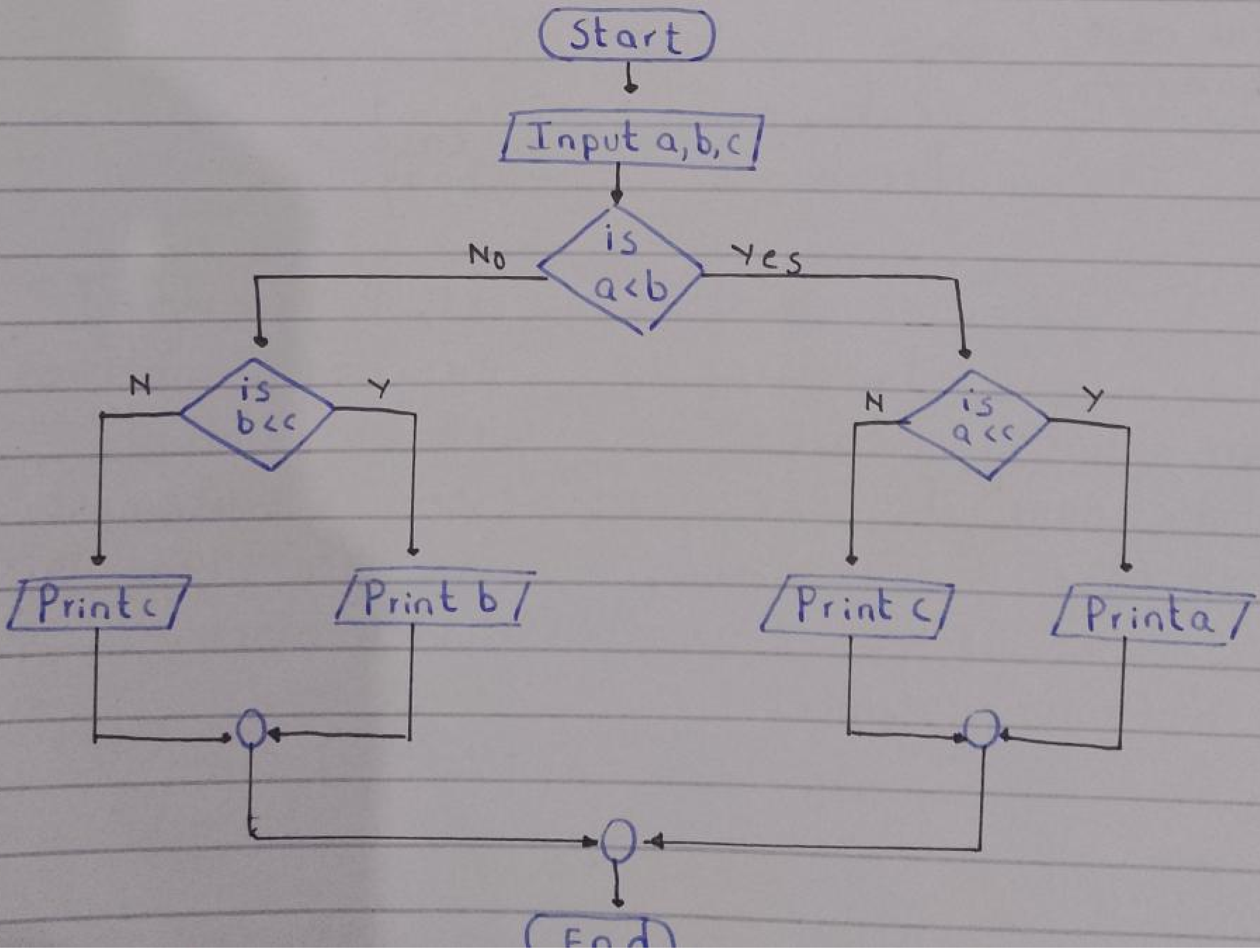
Working of Nested if:

In nested if statement, the condition of outer if is evaluted first. If it is true, the control enters in inner if block. If the condition is false, the inner if is skipped and control directly moves to the else part of outer if. If outer if is true then control enters in the innerif statement. The inner if evaluated according to simple if statement.

e.g Write a program that inputs three numbers and displays the smallest number by using nested if condition.

```
int main ()
{
    int a, b, c;
    cout << "Enter three numbers:";
    cin >> a >> b >>c;
if (a =<b)
    if (a =< c)
    { cout <<a<< "is smallest number.";
    else}
    { cout << c << "is smallest number";
     }
else if (b =< c)
   {
    cout<<b<< "is smallest number.";
   }
 else
   {cout <<c<< "is smallest number.";
   }
  }
```

Flow Chart:



Switch Statement:

- Select one of several alternatives when selection is based on the value of a single variable , an expression.
- In C++, the value of this expression may be of type int or char.
- The Switch statement is better way of writing a program when series of if-else-if occurs.

Syntax:

```
                                      → integer or charachter
                                                   variable.
                   switch (expression)
               {                          → integer or charachter
                   case constant 1:                   constant.
                   statement;              ] First
cause exit  ←      break;                  ]  case
 from case                                 ]  body
   body.           case constant 2:
                   statement;              ] Second
                   break;                  ]   case
                                           ]  body
                   :
                   case constant n:  → if the result of
                   statement;          expression matches
if the result      break;              with case label,
matches none  →  { default:            execute statement
of all case      { statement;          and then break this
label, execute  }                      Switch block.
the default statement.
```

'goto' statement:

=> The goto statement is used to move the control directly to a particular location of the program by using label. A

=> A label is a name given to particular line of program.

=> A label is created with valid identifier followed by a colon (:).

Syntax:

```
goto Label;
```

- The 'lable' indicates the label to which the control is transferred.

Q) Write a program that displays 'C++' five times using goto statement.

```
#include<iostream>
using namespace std;
int main ()
{
  int n = 1;
  loop:
  cout <<n <<"C++" <<endl;
  n ++;
  if (n <= 5) goto loop;

  cout <<"End of Program";
 }
```

Output:

```
C++
C++
C++
C++
C++
End of
  Program.
```

Compound Condition:

A type of comparison in which more than one conditions are evaluated is called compound condition.

=> Suppose a program iputs two numbers to display 'OK' if one number is greater than 50 and less than 100.

logical operator

(Number > 50 && Number < 100)

relational operator.

Logical Operator → "The logical operators are used to evaluate compound conditions."

AND Operator (&&):

- It is used to evaluate two conditions.
- It produces true result if both condition are true.
- It produces false result if any one condition is false.

Example:

Two variables A = 100 and B = 50.

=> Compound condition (A > 10) && (B > 10) is true.

=> Compound condition (A > 50) && (B > 50) is false.

OR Operator (||):

- Used to evaluate conditions.
- Produces true result if either condition is true.
- Produces false result if both conditions are false.

Example:

Two variables A=100 and B=50.

- Compound Condition (A >10) || (B >10) is true.
- Compound Condition (A >50) || (B >50) is true.
- Compound Condition (A >200) || (B >100) is false.

NOT Operator (!):

Reverse the result of condition

- Produces true results if condition is false.
- Produces false result if condition is true.

Example:

Two variables A=100 and B=50.

Condition !(A==B) is true.

└→ Result of condition (A==B) is false but NOT operator converts it into true.

Condition !(A >B) is false.

└→ Result of condition (A >B) is true but NOT operator converts it into false.

Conditional Operator:

Conditional Operator is a decision-making or selection control structure.

=> It can be used in place of simple if - else -> structure.

=> It is also called ternary operator as it uses three operands.

Syntax:

```
(condition)? true : false;
```

condition : The condition is specified as
t relational or logical expression. The condition is evaluated to true or false.

true-case : It is executed if expression evaluates to true.

false-case : It is executed if expression evaluates to false.

Example:

```
X =(A > 50) ? 1 : 0;
```

This statement will assign 1 to X if the condition (A > 50) is true.

It will assign 0 to X if condition is false.

It can be written using if-else statement

```
if (A>50)
    X = 1;
else
    X = 0;
```