SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

COURSE NAME : 23ITT101- PROBLEM SOLVING & C PROGRAMMING

I YEAR /I SEMESTER

Unit II – C PROGRAMMING BASICS

Topic : Decision making and Branching

INTRODUCTION

- Decision-making statements are used to control the flow of the program based on conditions. They are also known as **control statements**.

1. **if** statement
2. **switch** statement
3. Conditional operator statement

DECISION MAKING AND BRANCHING

- Introduction
- Decision Making with if Statements

  1.Simple if-statement

  2.if…...else statement

  3.Nested if…..else statement

  4.else if ladder

SIMPLE IF STATEMENT

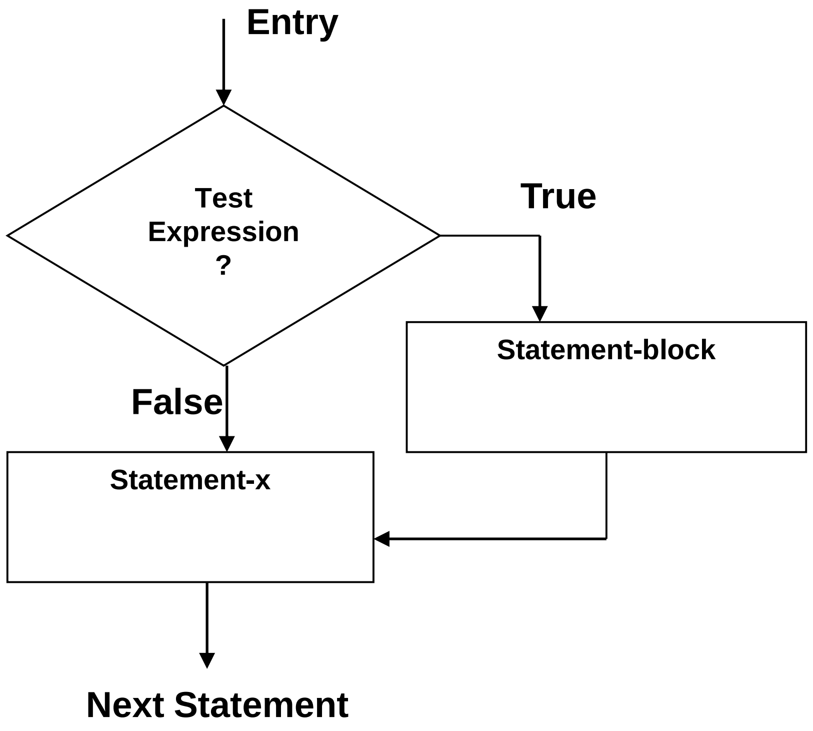
- The general form of simple if statement is.

```
if (test expression)
  {
       statement-block;
   }
       statement-x;
```

if Statement

- It is basically **a two-way decision (true / false)** statement and is used in conjunction with an expression.
- It **evaluate the expression first** and then, depending on whether the value of the **expression (relation or condition) is 'true' (or non-zero) or 'false' (zero),** it transfers the control to a particular statement.

Flowchart of simple if control



Example:

```
#include <stdio.h>
int main()
{
int a= 50;
int b = 100;
if (a<b)
{
printf("%d is less than %d",a,b);
}
return 0;
}
```

Output

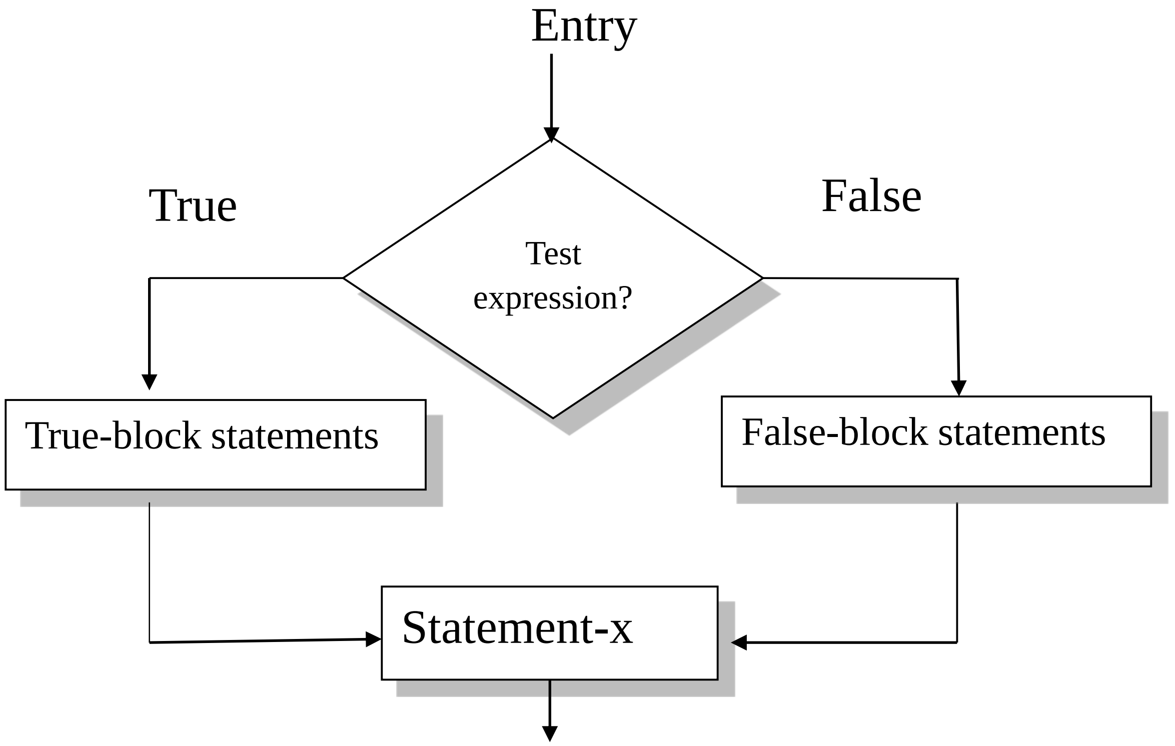
```
50 is less than 100
```

THE IF…..ELSE STATEMENT

The ‘if…else’ statement is the **extension of simple if statement**. The general form is,

```
if(test_expression)
  {
            True-block statement(s)
  }
else
 {
            False-block statement(s)
 }
statement-x;
```

THE IF…..ELSE STATEMENT



THE IF…..ELSE STATEMENT

```
#include <stdio.h>
int main()
{
int age;
printf("Enter your age:");
scanf("%d",&age);
if(age >=18)
  printf("You are eligible for voting");
else
  printf("You are not eligible for voting");
return 0;
}
```

Output

```
Enter your age:17
You are not eligible for voting
```

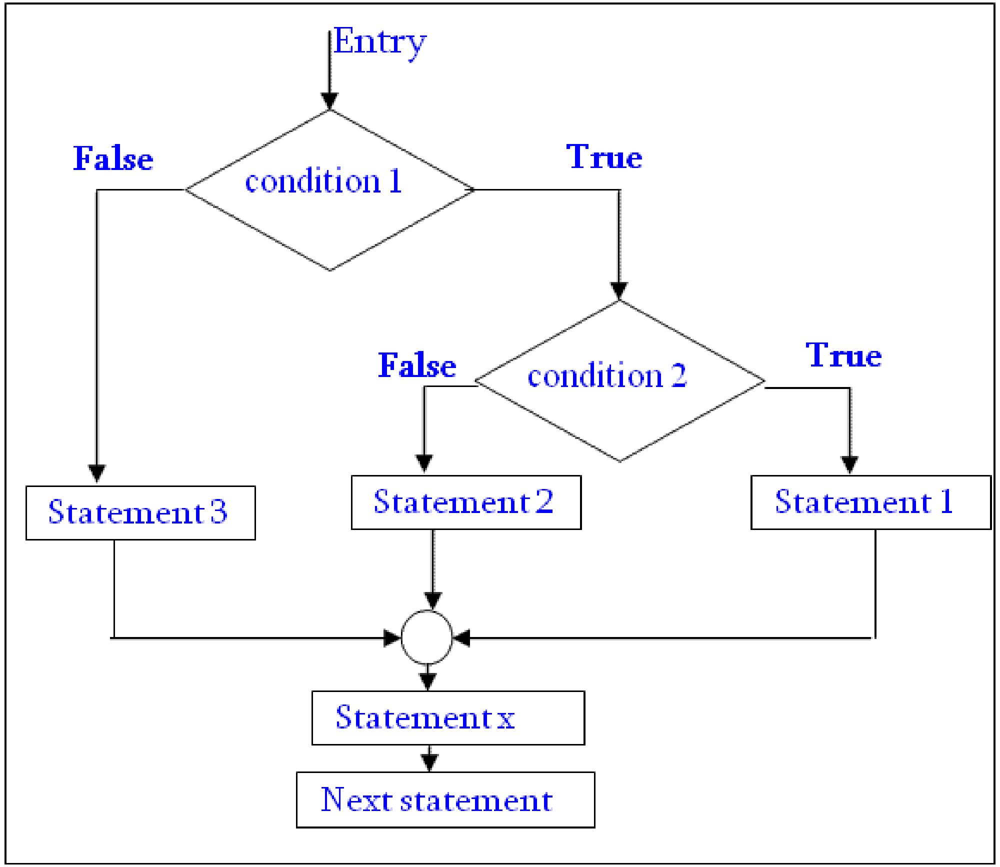
Output

```
Enter your age:18
You are eligible for voting
```

NESTING OF IF …ELSE STATEMENTS

General Form

```
if (test_condition1)
{
   if (test_condition2)
   {
       statement-1;
   }
   else
   {
        statement-2;
   }
}
else
{
        statement-3;
}
statement-x;
```



NESTING OF IF …ELSE STATEMENTS

```
#include<stdio.h>
int main( )
{ int n;
  printf("enter a number: ");
  scanf("%d",&n);
  if(n<=0)
{
  if(n<0)
{
  printf("%d is negative",n);
}
else
{
  printf("zero");
}
}
  else
{
printf("%d is positive",n);
}
  return 0;   }
```

<u>Output</u>

```
enter a number: 0
Zero
```

<u>Output</u>

```
enter a number: -7
-7 is negative
```

<u>Output</u>

```
enter a number: 10
10 is positive
```

ELSE IF LADDER

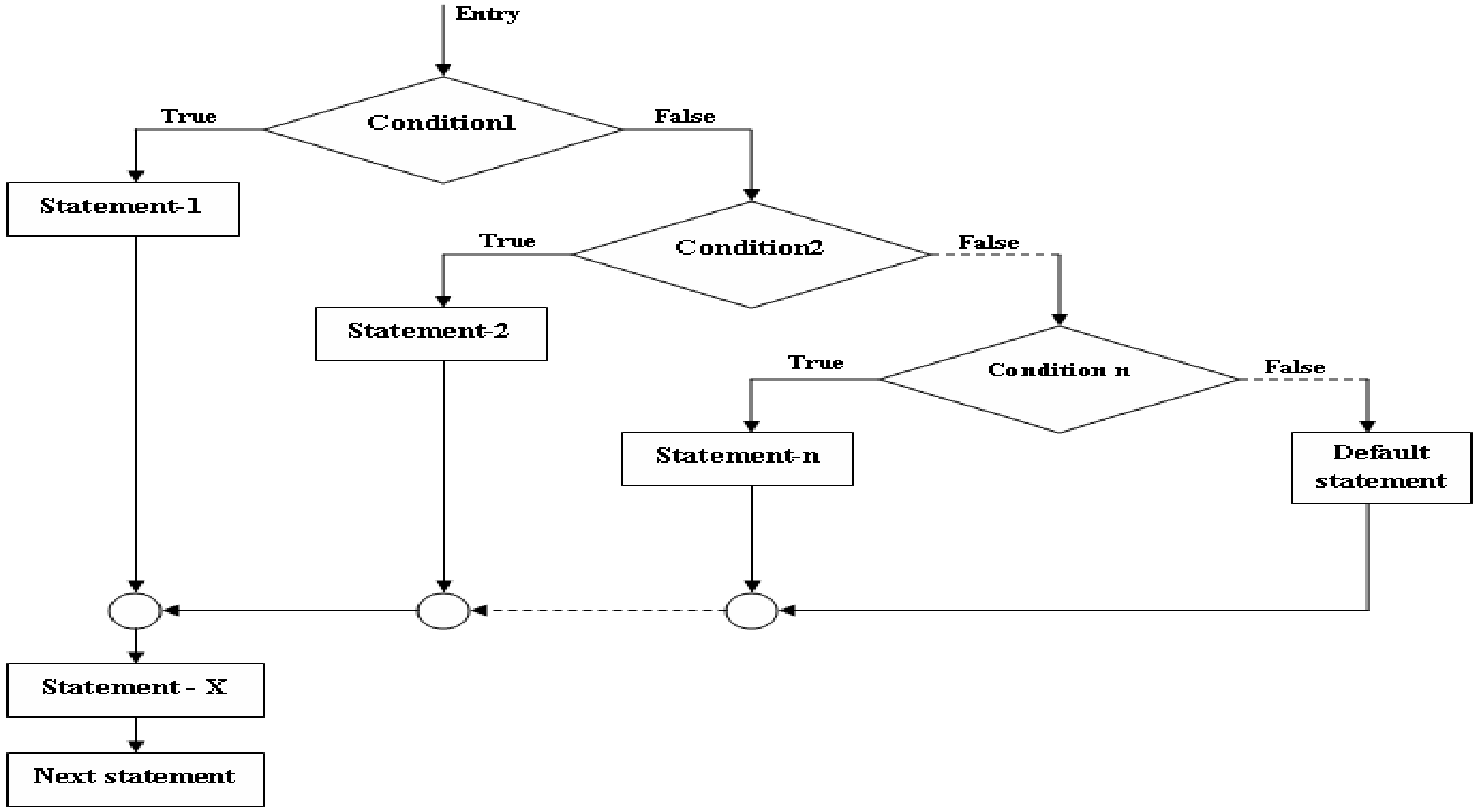
- A multi-path decision is a chain of if statement in which the statement associated with each else is an if statement.
- As soon as a true condition is found, the statement associated with it is executed and the control is transferred to the statement-x
- The conditions are evaluated from the top.
- When all the n conditions become false, then the final else containing the default-statement will be executed.

ELSE IF LADDER

- General Form

```
if ( condition 1)
            statement 1;
   else if (condition 2)
      statement 2;
   else if  (condition 3)
                  statement 3;
   else if  (condition n)
                  statement n;
else
            default statement;
statement-x;
```

ELSE IF LADDER



ELSE IF LADDER

```
#include<stdio.h>
int main()
{
 int marks;
printf(" Enter the marks for C Programming:\n");
scanf("%d",&marks);
 if(marks>=95){
   printf("O Grade");
 }
 else if(marks>=85){
   printf("S Grade");
 }
 else if(marks>=75){
   printf("A Grade");
 }
```

```
else if(marks>=65){
    printf("B Grade");
  }
else if(marks>=50){
    printf("C Grade");
  }
else if(marks>=40){
    printf("P Grade");
  }
  else{
    printf("Fail");
  }
  return 0;
}
```

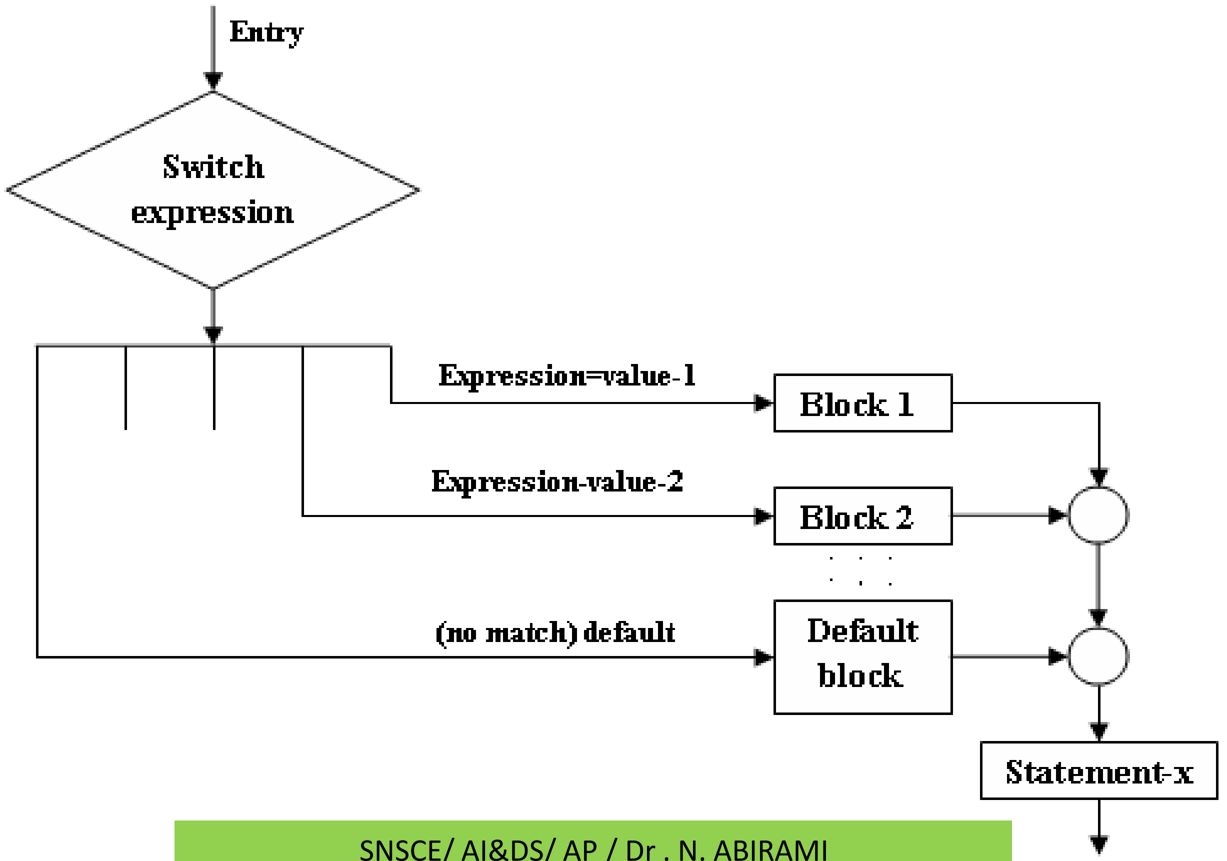
Switch Statement a multi-way decision statement.

Syntax:

```
switch (expression)
        {
  case value1 : block1;
                      break;
  case value2:  block2;
                      break;

  …….
  default:
                   default block;
                    break;
        }
```

Switch Statement



Rules for Switch Statement

- The **Switch expression** must be **an integral type**.
- **Case labels** must be **constants or constant expressions**.
- **Case labels** must be **unique**.
- **No two labels** can have the **same value**.
- **Case labels** must **end with colon**.
- The **break statement** is **optional**.

Rules for Switch Statement

- The **break statement** **transfers the control out of the switch statement.**
- **If present**, it will be **executed when the expression does not find a matching** case label.
- The **default** may be **placed anywhere but usually placed at the end**.
- It is **permitted to nest switch statements.**

Switch Statement - Example

```
#include<stdio.h>
int main( )
{
int c;
printf("Enter a number :");
scanf("%d",&c);
switch(c)
{
case 1:
    printf(“Blue");
    break;
```

```
case 2:
printf(“Green");
    break;

default:
    printf(“Invalid");
    break;
}
return 0;
}
```

Output

```
Enter a number :0
Invalid
```

Output

```
Enter a number :1
Blue
```

Output

```
Enter a number :2
Green
```

Switch Statement - Example

```
#include <stdio.h>
int main ()
{/* local variable definition */
char grade;
printf ("Enter a Grade A,B,C,D in
CAPS: \n");
scanf("%c",&grade);
switch(grade) {
case 'A' :
printf("Excellent!\n" );
break;
```

```
case 'B' :
case 'C' :
printf("Well done\n" );
break;
default :
printf("Invalid grade\n" );
}
printf("Your grade is
%c\n", grade );
return 0;
}
```

Output

```
Enter a Grade A,B,C,D in CAPS:
A
Excellent!
Your grade is  A
```

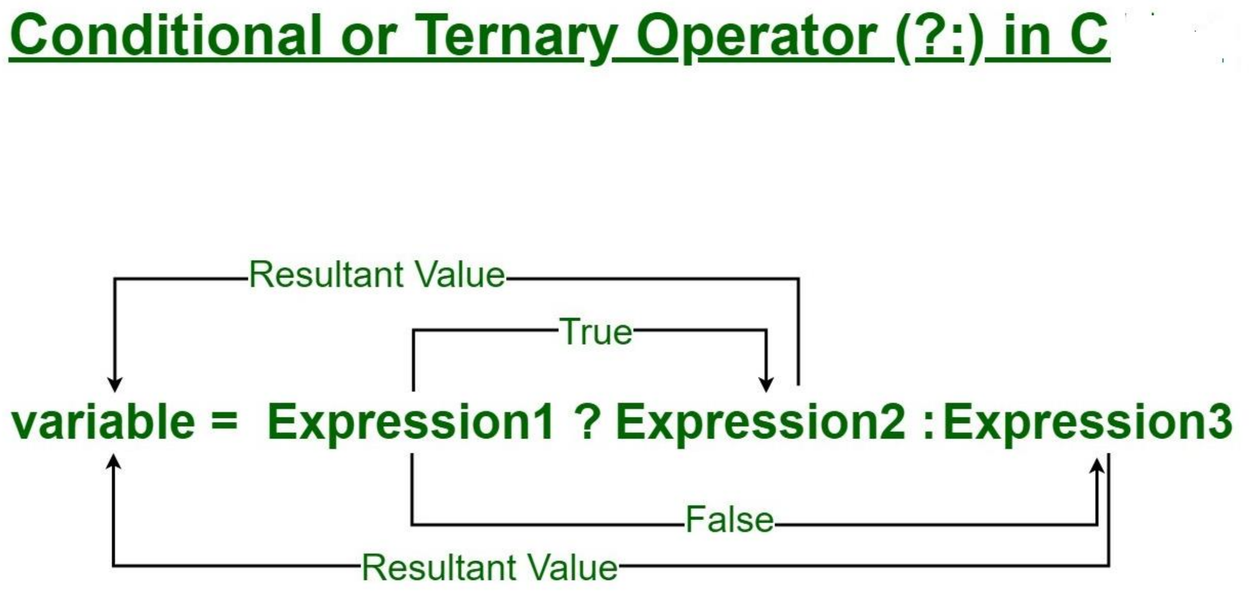
Output

```
Enter a Grade A,B,C,D in CAPS:
B
Well done
Your grade is  B
```

The Conditional Operator

- The C language has an **unusual operator**, useful for **making two-way decisions.**
- This operator is a **combination of ‘?’ and ‘:’** and **takes three operands.**
- This operator is popularly known as the **conditional operator**.

The Conditional Operator


Conditional or Ternary Operator (?:) in C
Resultant Value
True
variable = Expression1 ? Expression2 : Expression3
False
Resultant Value

The Conditional Operator

Syntax:

```
Conditional expression ? expression 1: expression 2;
```

- The **conditional expression is evaluated first**.
- If the **result is nonzero**, **expression 1 is evaluated** and is **returned as the value of the conditional expression**.
- Otherwise **expression 2 is evaluated** and **its value is returned**.

The Conditional Operator

```
#include <stdio.h>
int main() {
int y;
int x = 2;
y = (x >= 6) ? 6 : x;
/* This is equivalent to: if (x >= 5) y = 5; else y = x; */
printf("y =%d ",y);
return 0;}
```

Output

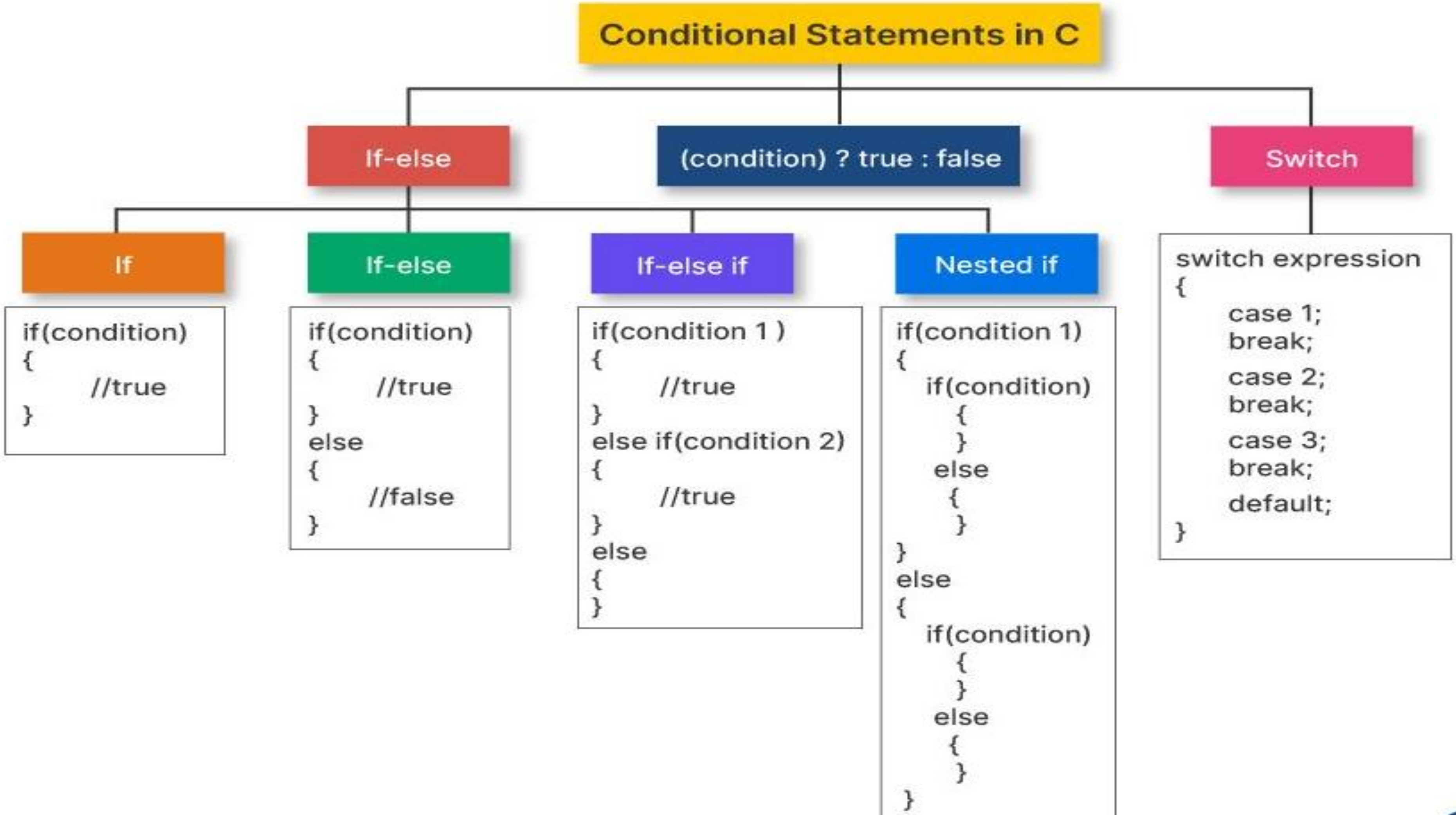
```
y =2
```

The Conditional Operator

```
#include <stdio.h>
int main()
{
// variable declaration
int n1, n2, max;
printf ("Enter a integer number: \n");
scanf("%d%d",&n1,&n2);
// Largest among n1 and n2
max = (n1 > n2) ? n1 : n2;
// Print the largest number
printf("Largest number between %d and %d is %d. ", n1, n2, max);
return 0;
}
```

Output

```
Enter a integer number:
112    56
Largest number between 112
and 56 is 112.
```



```
if(condition)
{
        //true
}
```

```
if(condition)
{
        //true
}
else
{
        //false
}
```

```
if(condition 1 )
{
        //true
}
else if(condition 2)
{
        //true
}
else
{
}
```

```
if(condition 1)
{
   if(condition)
      {
      }
    else
     {
      }
}
else
{
   if(condition)
      {
      }
    else
     {
      }
 }
```

```
switch expression
{
      case 1;
      break;
      case 2;
      break;
      case 3;
      break;
      default;
}
```

Thank you