Decision Control Structure

Dr. Chayan Halder
Assistant Professor
Ramakrishna Mission Vivekananda Centenary College, Kolkata

Introduction

- In different situations different actions need to follow.
- In programming the order of execution of instructions may have to be changed depending on certain conditions.
- This involves a kind of decision making to see whether a particular condition has occurred or not and then direct the computer to execute certain instructions accordingly.
- Executes some different set of instructions depending on different conditions

Decision making structures of C

- if statement
- switch statement
- Conditional operator statement
- goto statement

If Statement

- The IF statement is the powerful decision making statement and is used to control the flow of execution of the statements.
- When decision making in a program they are simply the result of computation in which the final result is either TRUE or FALSE.
- The value zero (0) is considered to be FALSE in program. Any positive or negative value is considered to be TRUE.

Levels of complexity for **if**:

Simple **if** statement

if…..**else** ladder statement

if……**else if**……**else** ladder statement

Nested **if**…..**else** statement

Simple If Statement

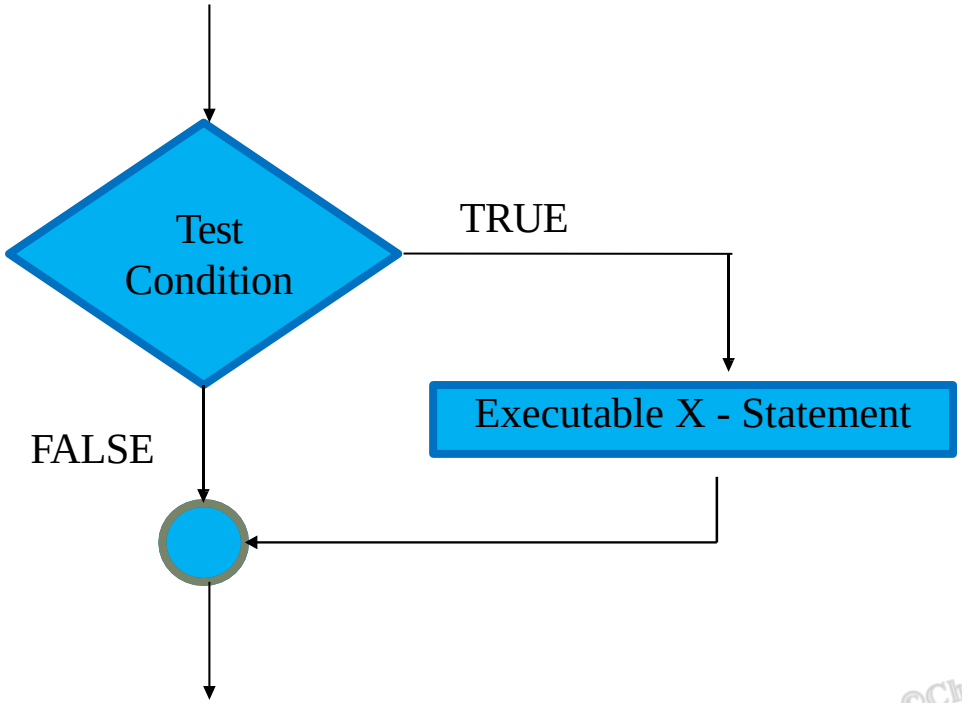
- The expression must evaluate to true or false.

- The “statement” can be a group of statements in braces

Syntax

```
if(expression)
{
   Statement 1;
   Statement 2;
   ….
}
```

• If the condition is true then the statement following the “if” is executed if it is false then the statement is skipped.



Flow chart of simple **if** statement

Example

```
#include<stdio.h>
int main( )
{
      int n ;
      printf ( "Enter your age: " ) ;
      scanf ("%d", &n ) ;
      if ( n>=20 )
      {
            printf (“\nYou are in your adulthood. ” ) ;
      }
}
```

```
Enter your age: 25
You are in your adulthood.
```

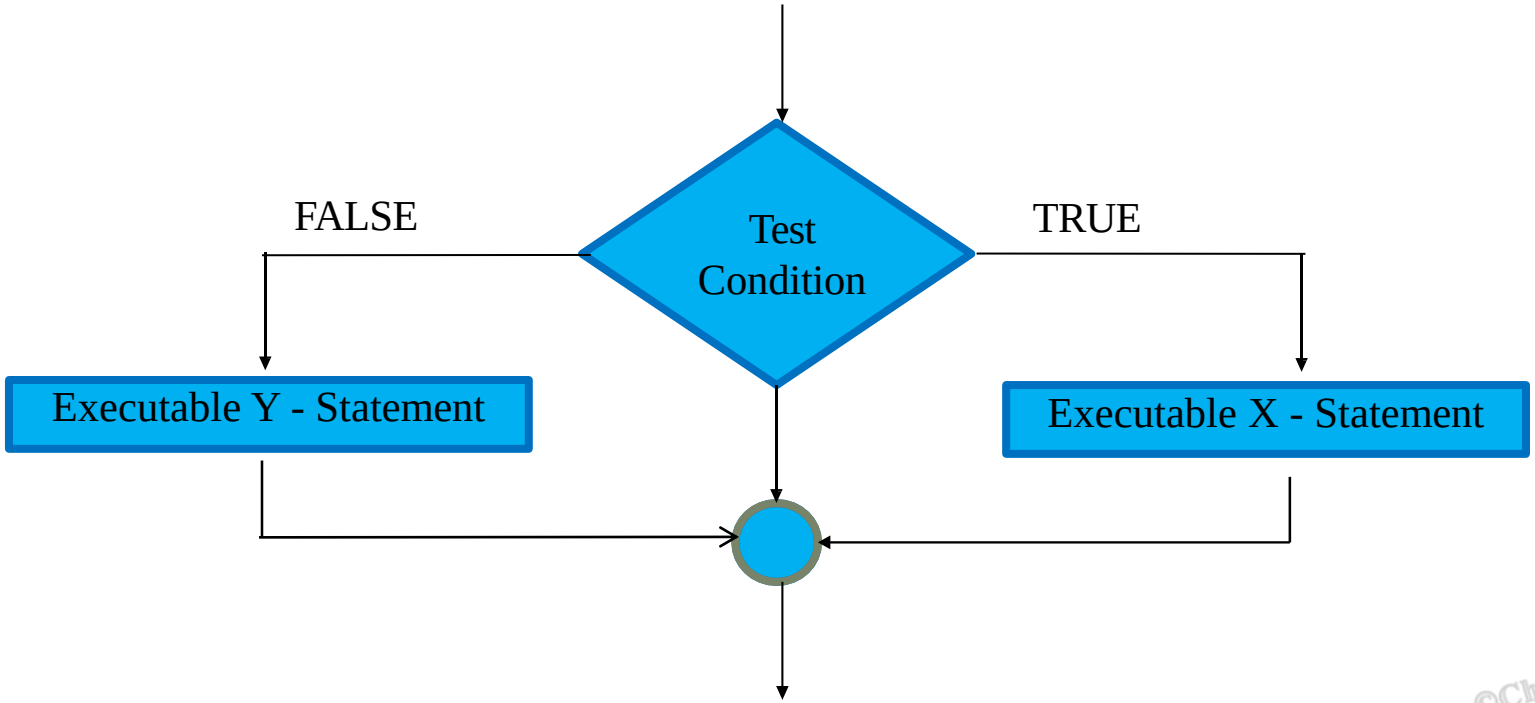
If… else ladder Statement

- An extension version of **if** statement.
- Group of statements can be executed for the false part of **if**.

Syntax

```
if (test expression)
{
        true block statement(s)
}
else
{
        false block statement(s)
}
```

• It is used to execute some statements when the condition is true and execute some other statements when the condition is false depending on the logical test.



Flow chart of **if**… **else** ladderstatement

Example

```
#include<stdio.h>
int main( )
{
    int n ;
    printf ( "Enter your age: " ) ;
    scanf ("%d", &n ) ;
    if ( n>=18 )
    {
        printf (“\nYou are eligible for citizenship ” ) ;
    }
    else
    {
        printf (“\n NOT ELIGIBLE” ) ;
    }
}
```

```
Enter your age: 13
NOT ELIGIBLE


Enter your age: 22
You are eligible for citizenship
```

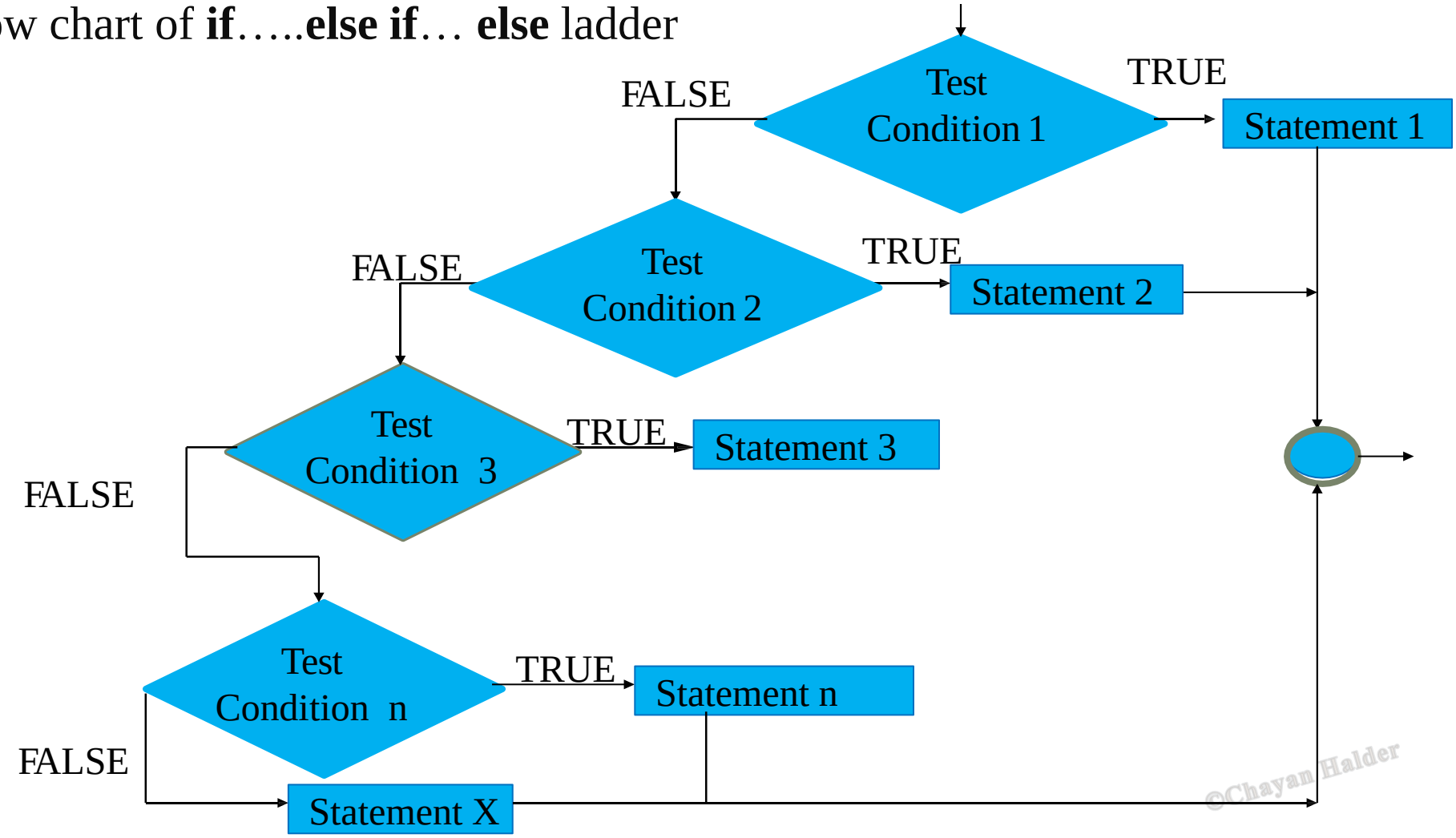
If… else if… else ladder Statement

- It is used to give series of decision

Syntax

```
if (condition)
{
//statement 1
}
else if (condition 2)
{
//statement 2
}
else
{
// statement when all condition are false
}
```

Flow chart of **if**…..**else if**… **else** ladder



Example

//Program to find the roots of the quadratic equations ax^2+bx+c=0

```
#include<stdio.h>
#include<math.h>
int main()
{
        float a,b,c,d,r1,r2;
        printf("Enter coefficients a, b and c: ");
        scanf("%f%f%f",&a,&b,&c);
        d=b*b-4*a*c;
        if(d==0)
        {
            r1=-b/(2*a);
            printf("\n Roots are equal and is %.3f",r1);
        }
        else if(d>0)
        {
                d=sqrt(d);
                r1=(-b+d)/(2*a);
                r2=(-b-d)/(2*a);
                printf("\nRoots are real and r1=%.3f, r2=%.3f",r1,r2);
                }
        else
        {
                d=sqrt(fabs(d));
                printf("\n Roots are imaginary");
                printf("\n r1=%.3f + i%.3f",-b/(2*a),d/(2*a));
                printf("\n r2=%.3f - i%.3f",-b/(2*a),d/(2*a));
        }
}
```

```
Enter coefficients a, b and c: 1 2 1
Roots are equal and is -1.000
```

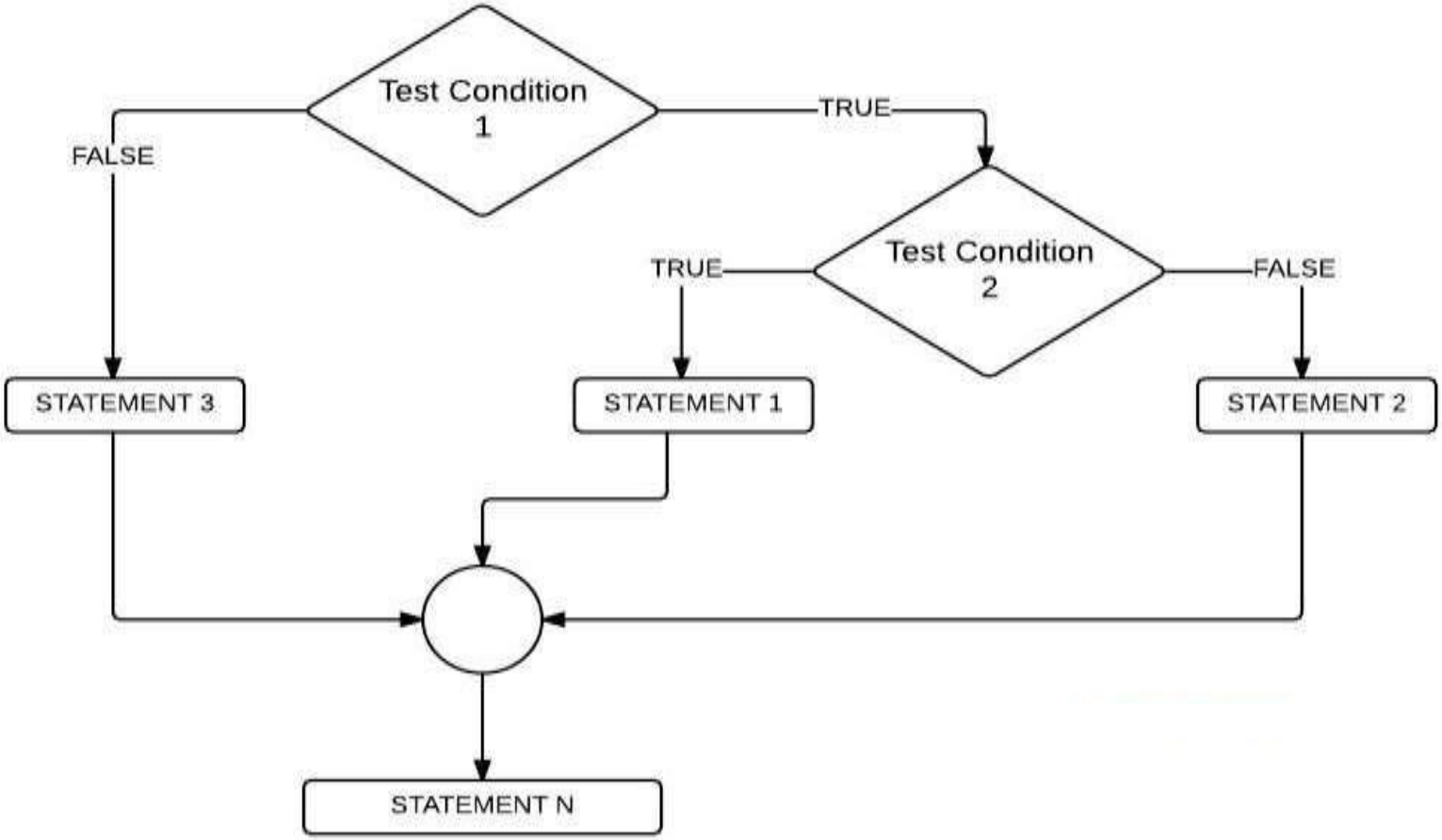
Nested if… else statement

- New block of if-else statement defined in existing if or else block statement.

Syntax

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

Flow chart of Nested if… else statement



Example

```
#include<stdio.h>
int main( )
{
  int a,b,c;
  printf("Enter 3 numbers: \n");
  scanf("%d%d%d",&a,&b,&c);
  if(a>b)
  {
    if( a > c)
    {
       printf("%d is greatest",a);
    }
    else
    {
       printf("%d is greatest",c);
    }
  }
  else
  {
    if( b> c)
    {
       printf("%d is greatest",b);
    }
    else
    {
       printf("%d is greatest",c);
    }
  }
return 0;
}
```

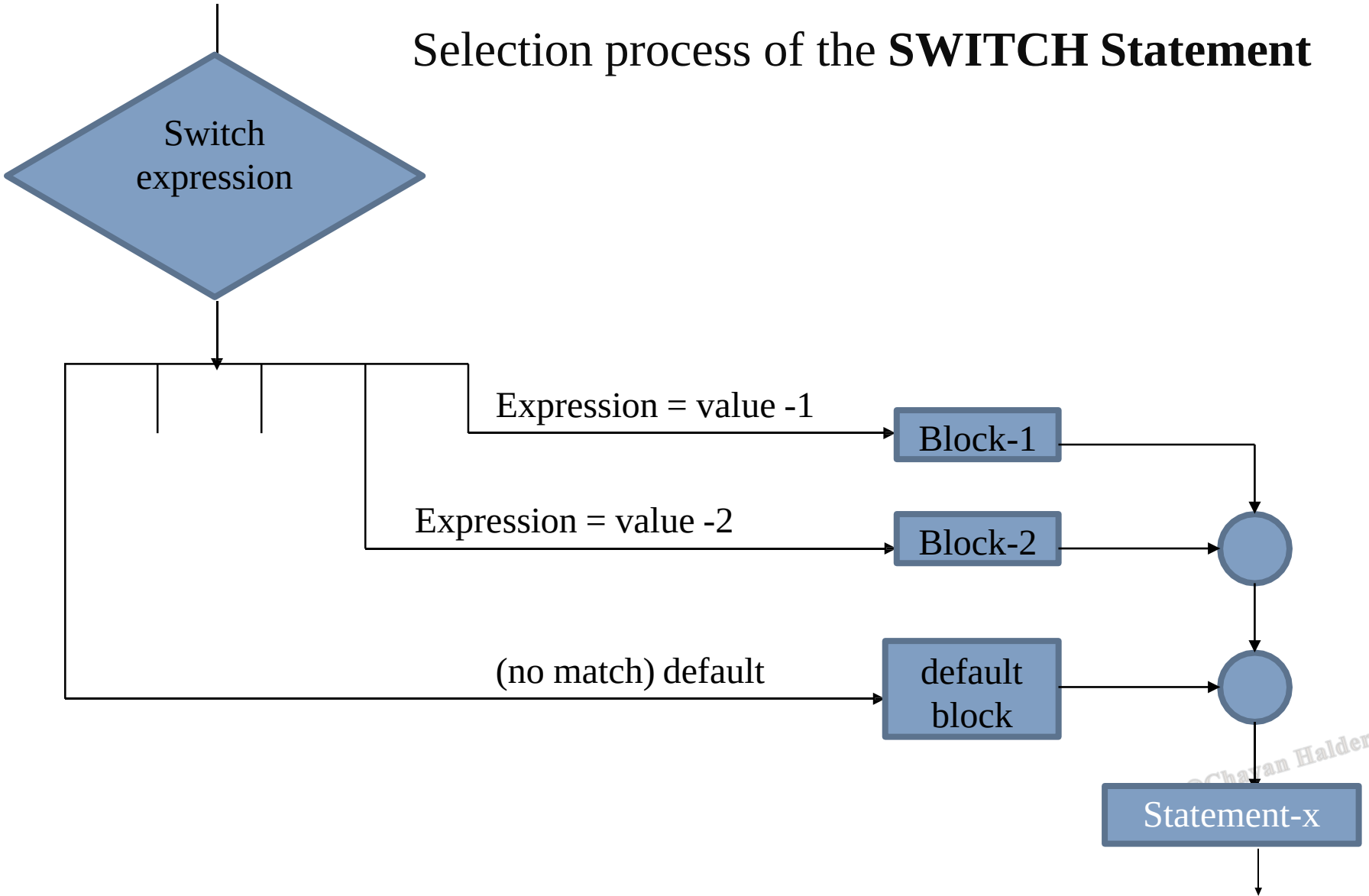
```
Enter 3 numbers:
9
11
5
11 is greatest

Enter 3 numbers:
49
55
88
88 is greatest
```

Switch Statement

- The control statement which allows us to make decision from the number of choices is called switch or switch case statement.
- It is a multi way decision statement, it test the given variable or expression against a list of case values.

```
switch (expression)
{
   case constant 1:
                        block 1
                        break;
   case constant 2:
                        block 2
                        break;
   default:
                    default block
                     break;
}
Statement –x;
```

Selection process of the **SWITCH Statement**



Example

```
Enter an operator(+,-): -
Enter two operands: 6
4
6.000000 - 4.000000 = 2.000000
Enter an operator(+,-): /
Enter two operands: 9
3
Error! operator is not correct
```

```
#include<stdio.h>
int main()
{
    char operator;
    float firstNumber,secondNumber;  printf("Enter an operator (+, -): ");
    scanf("%c", &operator);
    printf("Enter two operands: ");
    scanf("%f %f",&firstNumber, &secondNumber);
    switch(operator)
    {
            case '+':
                    printf("%f + %f = %f",firstNumber, secondNumber, firstNumber +  secondNumber);
                    break;

            case '-':
                    printf("%f - %f = %f",firstNumber, secondNumber, firstNumber -  secondNumber);
                    break;

            default:
                    printf("Error! operator is not correct");
     }
        return 0;
 }
```

Conditional Operator

- This operator is a combination of ? And : , and takes three operands.
- General form:

  conditional expression ? exp-1:exp-2;

- This concept says that if this expression is true then whatever we are writing after this question mark statement, it will be executed.

  And if this expression is false then whatever we are writing after this colon statement, it will be executed.

Example

```
Enter a number: 9
Odd
Enter a number: 8
Even
```

```
#include<stdio.h>
#include <math.h>

void main ()
{
        int a;

        printf("Enter a number: ");
        scanf("%d",&a);
        (a%2==0?printf("Even"):printf("Odd"));
        getch();
}
```

//If True first exp is executed

//If False second exp is executed

Goto Statement

- Like many other languages, C supports the GOTO statement to branch unconditionally from one point to another in the program.
- Although it may not be essential to use the GOTO statement in a Highly Structured Language.
- General form:

```
goto label;
…….
…….
…….
label:
    statement;
```

- Not recommended but may be used occasionally.

Example:

```
#include <stdio.h>
#include <math.h>
int main ()
{
        float y,x;
        read:
        printf("\nEnter a number: ");
        scanf("%f",&y);
        if(y<0)
        {
             goto read;
        }
        x=sqrt(y);
        printf("\nSquare root of %f---->%f\n",y,x);
        goto read;

        return 0;
}
```

```
Enter a number: 9
Square root of 9.000000---->3.000000
```

```
Enter a number: 25

Square root of 25.000000---->5.000000
```