

Chapter 11

DECISION CONSTRUCTORS

Q1. What are control structures? Also discuss its different types.

Ans.

Control Structures:

- These are the statements used to control the flow execution in a program or function.
- These instructions enable us to group individual instructions into a single logical unit with one entry point and one exit point.
- All programs use control structures to implement the program logic.

There are three types of control structures:

1. Sequence
 2. Selection
 3. Repetition / Iteration / Loops
- Sequence
 - o The instructions are executed in a sequence in which they are written in the program.
 - o It is default flow of the program.
 - o The execution of the program starts from first instruction and all instructions are executed one by one in a sequence.

Example: compound statement:

```
{  
  Statements  
    St - 1  
    St - 2  
    ----  
    ----  
}
```

- Selection
 - o A structure that selects which statement or block of statements is to execute.

- o In selection structure instruction are divided into two or more groups and the selection is done after testing a condition.

Examples:

1. if – else
 2. switch
- Repetition/ iteration/ loops:
 - o These statements are used to repeat a set of statements as long as condition is true.

Examples:

1. for-loop
2. while loop
3. do-while loop

Q2. What is an if statement? Explain with the help of an example and flow chart.

Ans.

if statement:

- If is a keyword in C language. It is used to execute or ignore a set of statements after testing a condition.
- A condition is a relational or logical expression and it produces either true (means 1) or false (means 0) result. If the condition is true then the block of statement is executed and if the condition is false then the block of statement is ignored and the control is transferred to the next statement after if statement.

Syntax:

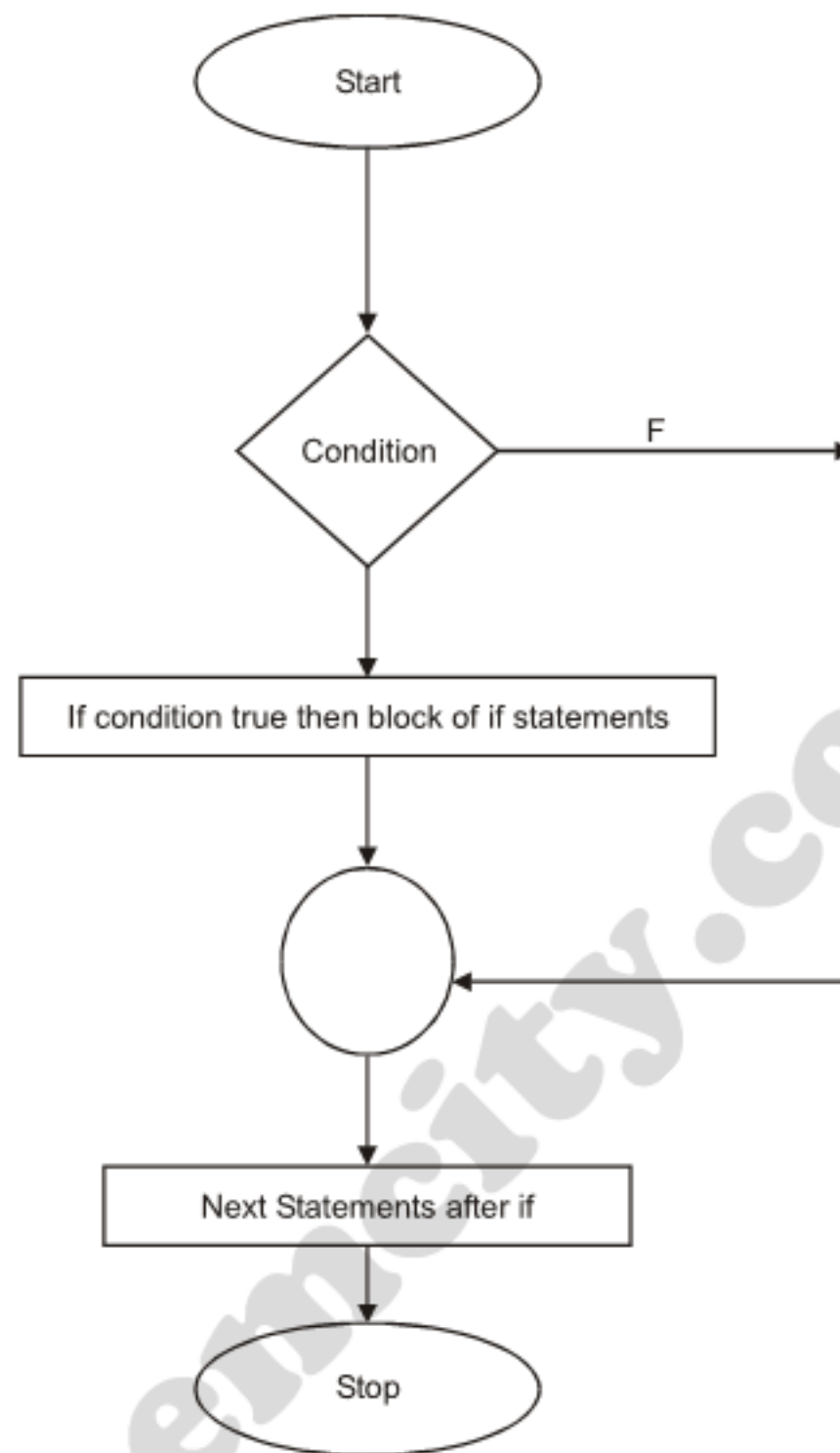
```
if (relational or logical condition)
{
    Block of statements
}
next statement after if
```

Example: Even or Odd Number

```
#include<stdio.h>
void main ( )
{
    int n,r;
    printf("\n Enter any Number:      ");
    scanf("%d", &n);
    r=n%2;
    if(r==0)printf("\n The Number is Even ");
    if(r==1) printf("\n The Number is Odd ");
}
```

}

Flow Chart:



Program:

Write a program in C language that takes marks from user and display “PASS” if marks are greater than or equal to 50.

```
#include<stdio.h>
#include<conio.h>
void main( )
{
    int m;
    clrscr( );
    printf(“Enter your marks\t”);
    scanf(“%d”,&m);
    if(m>=50)
        printf(“PASS”);
    getch( );
}
```

Output:

Enter your marks 60
PASS

Program:

Write a program in C language that takes two number and displays a message if first number is square of second.

```
#include<stdio.h>
#include<conio.h>
void main( )
{
    int x,y;
    clrscr( );
    printf("Enter first number\t");
    scanf("%d",&x);
    printf("Enter second number\t");
    scanf("%d",&y);
    if(x==y*y)
        printf("%d is square of %d", y,x);
    getch( );
}
```

Program:

Write a program to find out the roots of quadratic equation:

```
#include<stdio.h>
#include<math.h>
void main( )
{
    float a,b,c,r1,r2,disc,real,imag;
    printf("Enter value of A?");
    scanf("%f",&a);
    printf("Enter value of B?");
    scanf("%f",&b);
    printf("Enter value of C?");
    scanf("%f",&c);
    disc = b*b-4.0*a*c;
    if (disc<0)
    {
        real = -b/(2.0*a);
        imag = sqrt(-disc)/(2.0*a);
        printf("Roots are imaginary\n");
    }
}
```

```
printf("Root1 = %f +i %f\n", real, imag);
printf("Root2 = %f -i %f\n", real, imag);
}
if (disc == 0)
{
r1 = r2 = -b/(2.0*a);
printf("Roots are real & equal\n");
printf("Root1 = %f\n", r1);
printf("Root2 = %f\n", r2);
}
if (disc > 0)
{
printf("Roots are real and different\n");
r1 = -b/(2.0*a) + sqrt(disc)/(2.0*a);
r2 = -b/(2.0*a) - sqrt(disc)/(2.0*a);
printf("Root1 = %f\n", r1);
printf("Root2 = %f\n", r2);
}
}
```

Q3. What is an if-else statement? Explain with the help of an example and flow chart.

Ans.

if-else statement:

- It is similar to if statement i.e. It is also used to execute or ignore a set of statements after testing a condition. It is a two way decision making statement.
- In if-else statement one condition and two blocks of statements are given. A condition is a relational or logical expression and it produces either true (means 1) or false (means 0) result. If the condition is true then the first block of statement is executed and 2nd is ignored. And after executing the first block, the control is transferred to next statement after if-else structure. If the condition is false then the first block of statement is ignored and the 2nd block of statement is executed. And after executing the 2nd block of statement the control is transferred to the next statement after if-else structure.

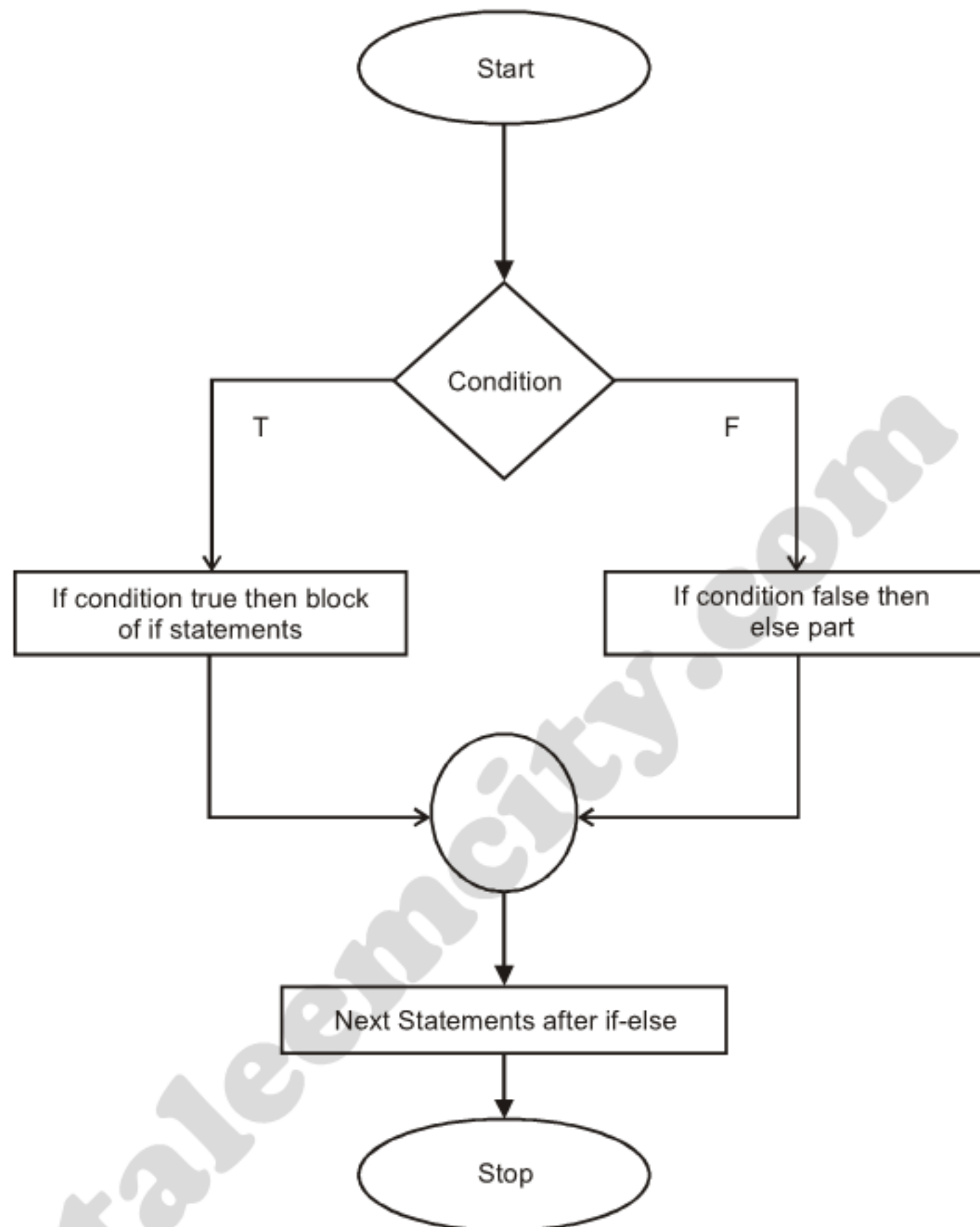
Syntax:

```
if (relational or logical condition)
{
    first Block of statements
}
else
```

```
{  
    2nd block of statements  
}  
Next statements after if-else
```

taleemcity.com

- **Flow Chart:**

**Program:**

Write a program in C language that finds whether a given year is leap year or not. Leap year is a year that is divisible by 4.

```
#include<stdio.h>
#include<conio.h>
void main( )
{
    int y;
    clrscr( );
    printf("Enter Year\t");
```

```
scanf("%d",&y);
if(y%4==0)
    printf("Leap year");
else
    printf("Not leap year");
getch( );
}
```

Output:

Enter year 2008

Leap year

Program:

Even or Odd Number

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

```
#include<conio.h>
```

```
void main ( )
```

```
{
    int n,r;
    printf("\n Enter any Number: ");
    scanf("%d", &n);
    r=n%2;
    if(r==0)
        printf("\n The Number is Even ");
    else
        printf("\n The Number is Odd ");
}
```

Q4. What is conditional or ternary operator? Explain with the help of example.

Ans.

Conditional Operator or Ternary Operator:

- It is used as an alternative to simple if-else statement and it is also called ternary operator.
- It requires three expressions. First expression is relational or logical expression (condition) and it produces either true (means 1) or false (means 0) result.
- If the condition is true then 2nd expression after question mark (?) is executed and if the condition is false then the 2nd expression is ignored and 3rd expression after colon (:) is executed.

Syntax:

Conditional Statement?true case statement: false case statement ;

Example: Even or Odd Number

```
#include<stdio.h>
#include<conio.h>
void main ( )
{
    int n,r;
    printf("\n Enter any Number:    ");
    scanf("%d", &n);
    r=n%2;
    r==0 ? printf("\n Even Number"):printf("\n Odd Number ");
}
```

Q5. What is a switch statement? Explain with the help of example.

Ans.

Switch Statement:

- Switch is also a control structure and it is used to select one option from a set of options.
- It compares the value of an expression against a list of cases.
- The case labels and the value of expression must be an integer or a character.
- It must not be a float or double value. If the value of expression in switch is float or double type then the compiler will generate an error message.

Syntax:

```
switch(expression)
{
    case label - 1:
        statements – 1;
        break;
    case label - 2:
        statements – 2;
        break;
    case label - 3:
        statements – 3;
        break;
    -----
    -----
    case label - n:
        statements – n;
        break;
```

```
    default:  
    statements ;  
}
```

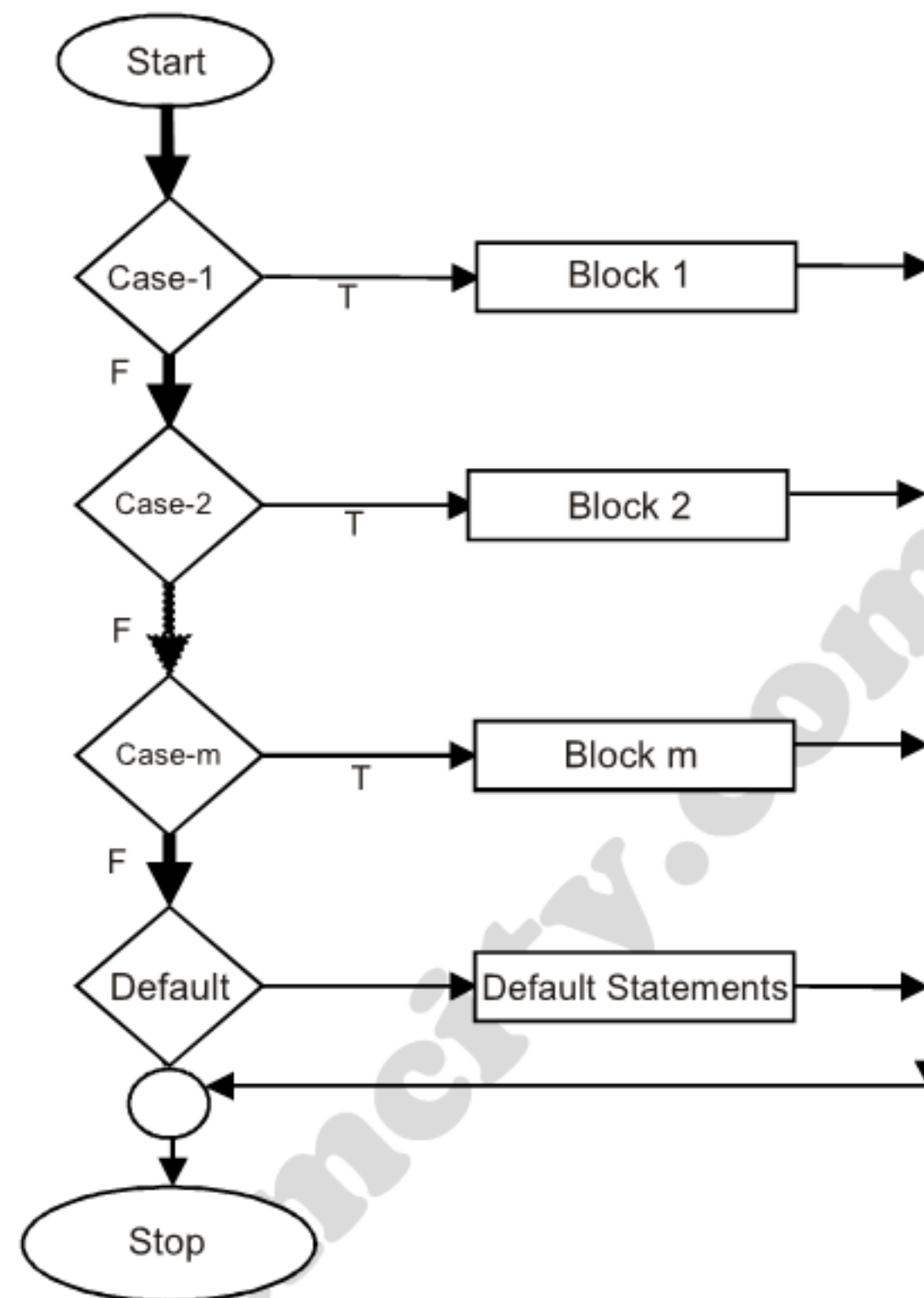
- The value of expression is compared to each case label.
- The case whose case value matches the value returned by the expression is executed.

The use of break keyword:

- It must be included at the end of each case statement and is used to exit from the body of switch.
- If all break statements are omitted then the code of other cases after the matching case will be executed sequentially.

The use of default keyword:

- Its use is optional and if none of the case label is matched. Then the statements under default are executed.
- The position of default is not fixed. It may be placed before the first case or after the last case.



Example: Vowel or a Consonant:

```
#include<stdio.h>
#include<conio.h>
void main ( )
{
    char ch;
    printf("\n Enter any Character:  ");
    scanf("%c", & ch);
    switch(ch)
    {
        case 'a': case 'A':
            printf("\n Vowel Character A ");
            break;
        case 'e': case 'E':
            printf("\n Vowel Character E ");
            break;
        case 'i': case 'I':
            printf("\n Vowel Character I ");
            break;
        case 'o': case 'O':
            printf("\n Vowel Character O ");
            break;
        case 'u': case 'U':
            printf("\n Vowel Character U ");
            break;
        default:
            printf("\n It is a Consonant Character ");
    }
}
```

Q6. What is nested if statemnt? Explain with the help of example and flow chart.

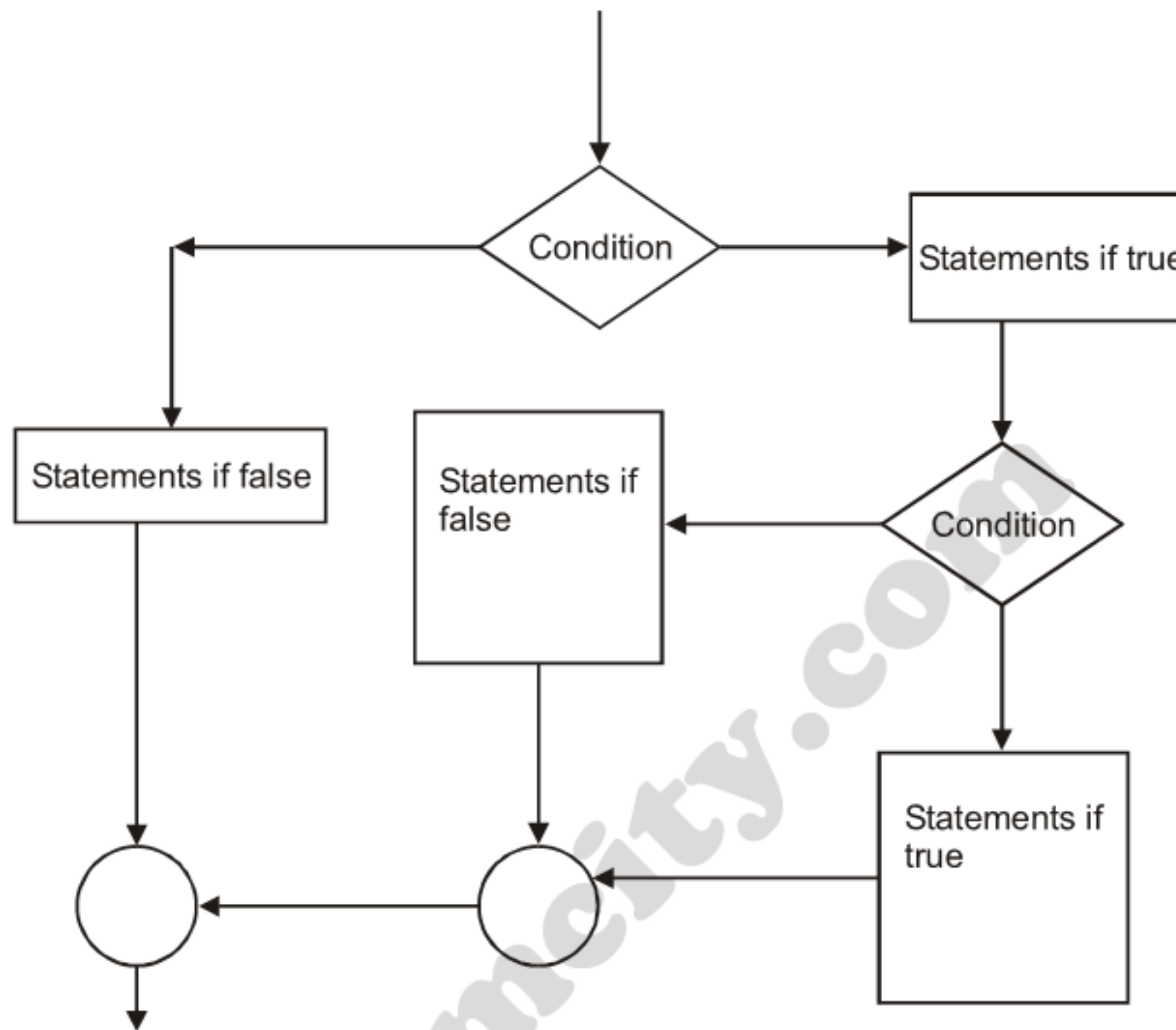
Ans.

Nested-if statement:

- if statement within another if statement is called nested if statement.
- It is used for multi-way decision making.
- Nesting can be done up to any level. Increase in nesting also increase the complexity of program.
- The else statement is optional. It may be used with outer or inner if statement.

Syntax:

```
if(condition-1)
{
    if(condition-2)
    {
        statements of inner if
    }
}
```

Flow Chart:**Program:**

Comparison of 3 values:

```

#include<stdio.h>

void main ( )
{
    int a,b,c;
    printf("\n Enter three Numbers :    ");
    scanf("%d %d %d", &a, &b, &c);
    if(a>b)
    {
        if(a>c)
            printf("\n first value a is largest");
        else

```

```
        printf("\n third value c is largest");
    }
    else
    {
        if(b>c)
            printf("\n second value b is largest");
        else
            printf("\n third value c is largest");
    }
}
```

Q7. What is if-else-if statement? Explain with the help of example and flow chart.

Ans.

if-else-if statement:

- If there are more than two alternatives, then we use if-else-if instead of nested if statement.
- It is used when multiple options are available and only one option is to be selected.
- Nesting can be done up to any level.
- The conditions in if are evaluated in a sequence.
- All the conditions are evaluated until a true condition is reached.
- When a true condition is found, then the statements under true conditions are executed and the remaining conditions are ignored.
- If all conditions are false, then the last statement – k after else is executed.

Syntax:

```
if(condition-1)
    statement – 1
else if(condition-2)
    statement – 2
else if(condition-3)
    statement – 3
-----
else if(condition-n)
    statement – n
else
```

statement – k

Program:

Perform arithmetic operations using if-else-if

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

```
#include<conio.h>
```

```
void main ( )
```

```
{
```

```
float a,b;
```

```
char op;
```

```
printf("\n Enter two Numbers :      ");
```

```
scanf("%f %f", &a, &b);
```

```
printf("\n Enter operator + - * or / :  ");
```

```
scanf("%c", & op);
```

```
if(op=='+') printf("The Result = %.2f",a+b);
```

```
else if(op=='-') printf("The Result = %.2f",a-b);
```

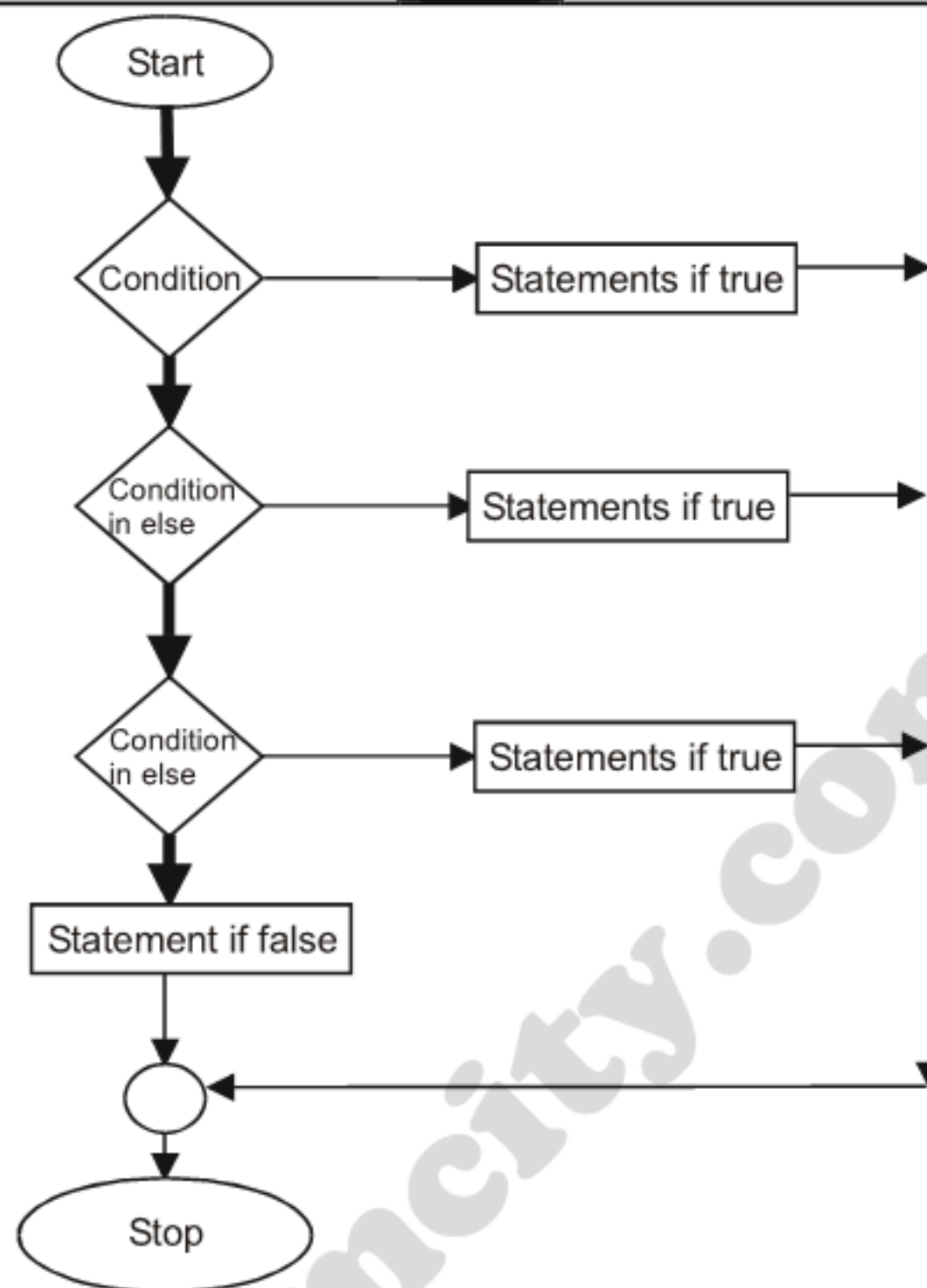
```
else if(op=='*') printf("The Result = %.2f",a*b);
```

```
else if(op=='/') printf("The Result = %.2f",a/b);
```

```
else printf(" Operator is invalid ");
```

```
}
```

Flow Chart:



Comparison of nested if and sequence of if statement:

1)	Sequence of if is simple than nested if statement	Nested if is more complex than sequence of if statement.
2)	All conditions must be tested.	In nested if when a logical decision is true then the remaining conditions are ignored.
3)	In sequence of if statement, mostly CPU time is consumed in testing all conditions.	In nested if mostly statements are executed or ignored by testing the condition in outer structure, therefore, it is a time saving process.

Distinguish between if-else if statement and switch statement:

The “nested if-else” and “switch” Statement:

Both the nested “if-else” and switch statement are used for multiple selections but following are the differences between these statements:

Nested if-else Statement	Switch Statement
(i) It becomes complicated for multiple	It is easy to understand for multiple

selections.	selections.
(ii) It uses an independent expression for each case.	It uses a single expression for all cases, but each case must have a constant value of integer type or character type.
(iii) The test condition can be given in a specified range of values. If the given condition matches then the statements under it will be executed.	Only a single expression is given in the switch Statement which returns a single value. The test condition cannot be given in a specified range. It is the major drawback of the switch statement.

Program:

Write a program to locate a point in the coordinate plane.

```
#include<stdio.h>
void main(void)
{
    int x,y;
    printf("\nEnter x- and y-coordinates?");
    scanf("%d, %d", &x, &y);
    if(x == 0)
    {
        if(y == 0)
            printf("The point is on the origin.");
        else
            printf("The point is on y-axis.");
    }
    else if(x > 0)
    {
        if(y == 0)
            printf("The point is on x-axis.");
        else if(y > 0)
            printf("The point is in 1st quadrant.");
        else
            printf("The point is in 4th quadrant.");
    }
}
```

```
else
{
    if(y == 0)
        printf("The point is on x-axis.");
    else if(y > 0)
        printf("The point is in 2nd quadrant.");
    else
        printf("The point is in 3rd quadrant.");
}
```

taleemcity.com