UNIT 3 : DECISION MAKING STATEMENTS

3.1 if statements :

- 3.1.1 simple if statements
- 3.1.2 if…else statements
- 3.1.3 if…else if….else statements
- 3.1.4 Nested if statements.

3.2 Switch..case statements

- 3.2.1 Use of break and default
- 3.2.2 Difference between switch and if statements

Control Statement or Control Structure

- Control Statements or Control Structures are used to control the flow of Program.
- Control structures are the statements that control the flow of the source code.

- There are three categories of flow controls:
  1. Branching or Selection Statement
  2. Looping or Iterative Statement
  3. Jumping

3.1. Branching or Conditional Statement or if Statement

- Branching are decision making statements.
- The **C language** programs follows a sequential form of execution of statements. Many times it is required to alter the flow of sequence of instructions. **C language** provides statements that can alter the flow of a sequence of instructions.

- □ 3.1.1. Simple if Statement
- □ 3.1.2. if…else Statement
- □ 3.1.3. Nested if Statement
- □ 3.1.4. Ladder if or elseif Statement

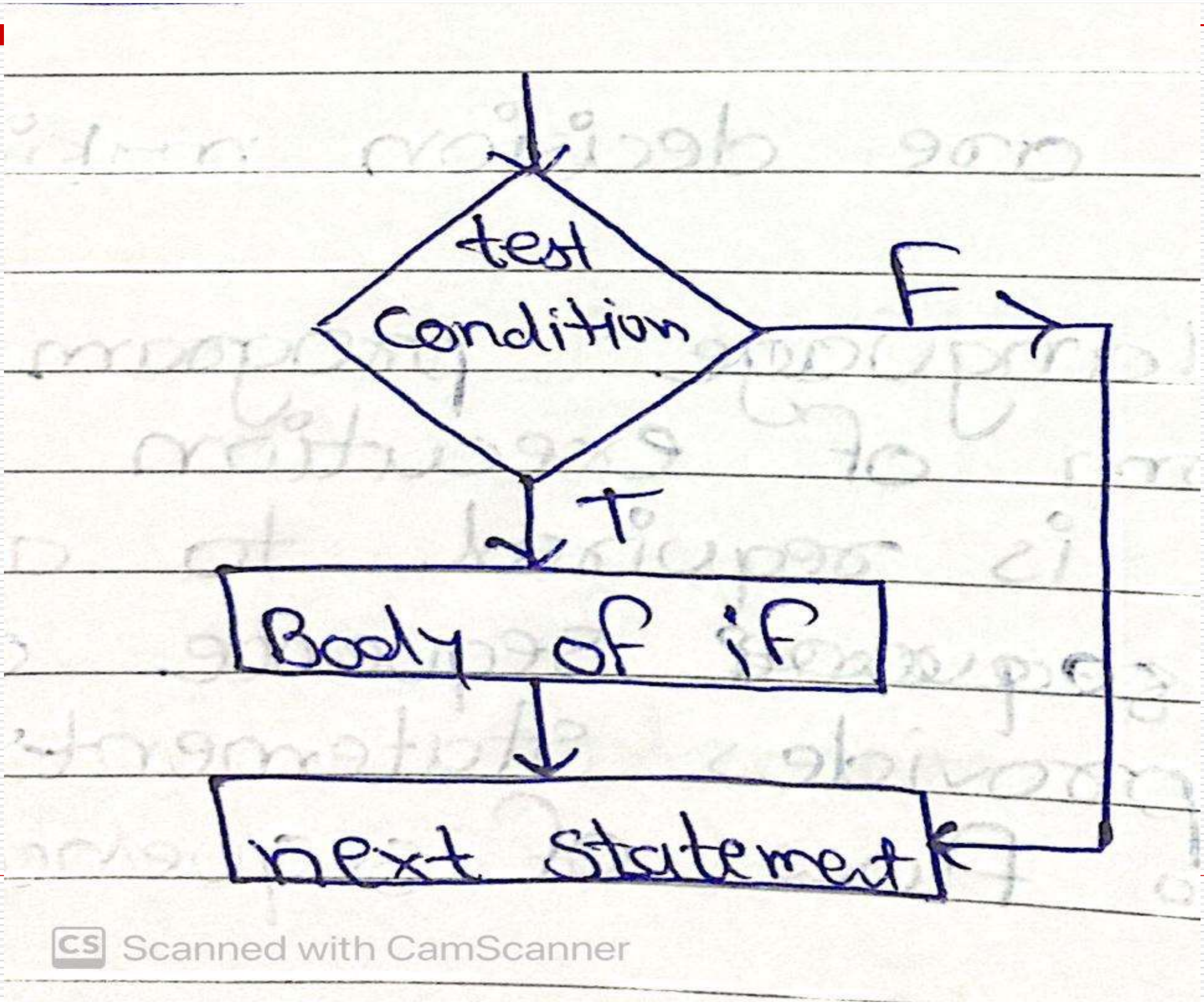
3.1.1. Simple if Statement

□ When we need to execute a block of statements only when a given condition is true then we use if statement.

Syntax:

```
if (test - condition)
{
     //Statement
}
Next statements
```

Flowchart:



Example:

```
void main()
{
        int a=10,b=20;
        clrscr();
        if(a<b)
        {
                printf("Inside IF");
        }
        printf("\nOutside IF");
        getch();
}
OUTPUT:
Inside IF
Outside IF
```

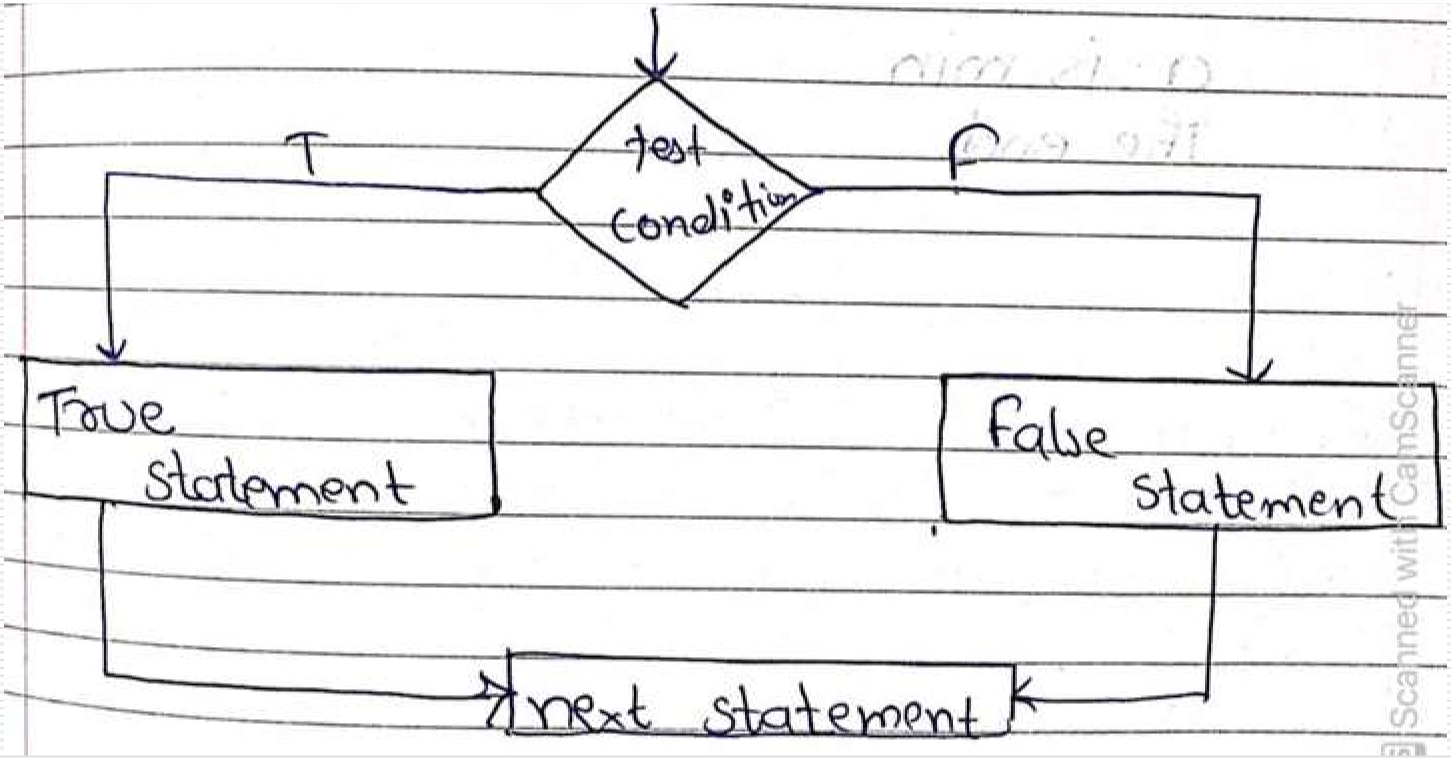
3.1.2. if...else Statement

- An **if** statement can be followed by an optional **else** statement, which executes when the Boolean expression is false.

Syntax:

```
if(test - condition)
{
   // True Statements
}
else
{
   // False Statements
}
```

Flowchart:


test condition
T
F
True Statement
False Statement
next statement

Example:

```
void main()
{
        int a=10,b=20;
        clrscr();
        if(a<b)
        {
                printf("Inside IF");
        }
        else
        {
                printf("\nInside ELSE");
        }
        getch();
}
```

OUTPUT:

```
Inside IF
```

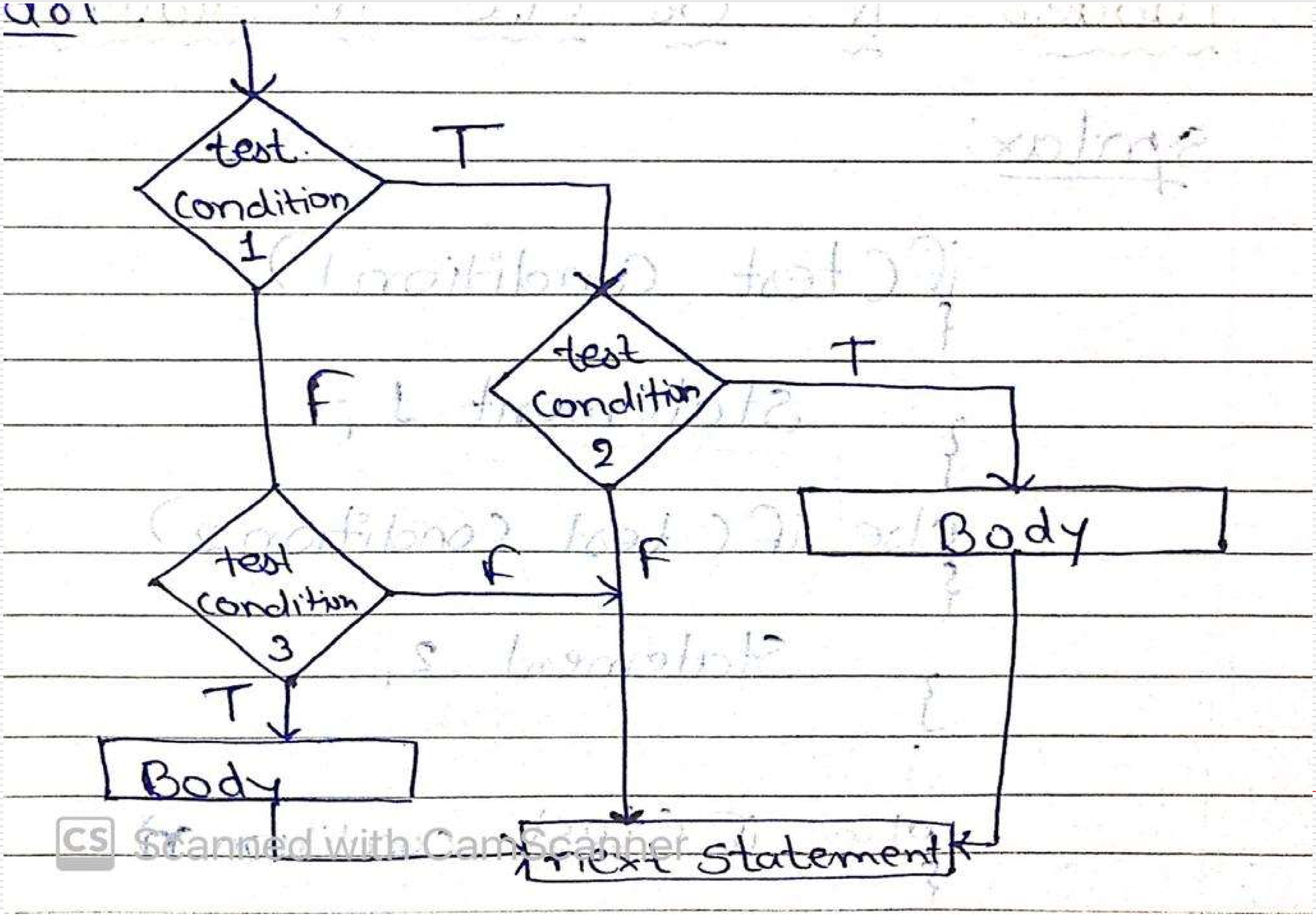
3.1.3. Nested if Statement

- If statement within if statement is known as nested if statement.

Syntax:

```
if(test - condition 1)
{
  if(test - condition 2)
  {
         // True Statement
  }
}
else
{
    if(test - condition 3)
    {
         // False Statement
    }
}
```

Flowchart:


test condition 1
T
F
test condition 2
T
F
Body
test condition 3
F
T
Body
next statement

Example:

```
void main()
{
    int a=10,b=20,c=30;
    clrscr();
    if(a<b)
    {
        if(b<c)
                printf("Inside IF");
    }
    else
    {
        printf("\nInside ELSE");
    }
    getch();
}
```

OUTPUT:

```
Inside IF
```

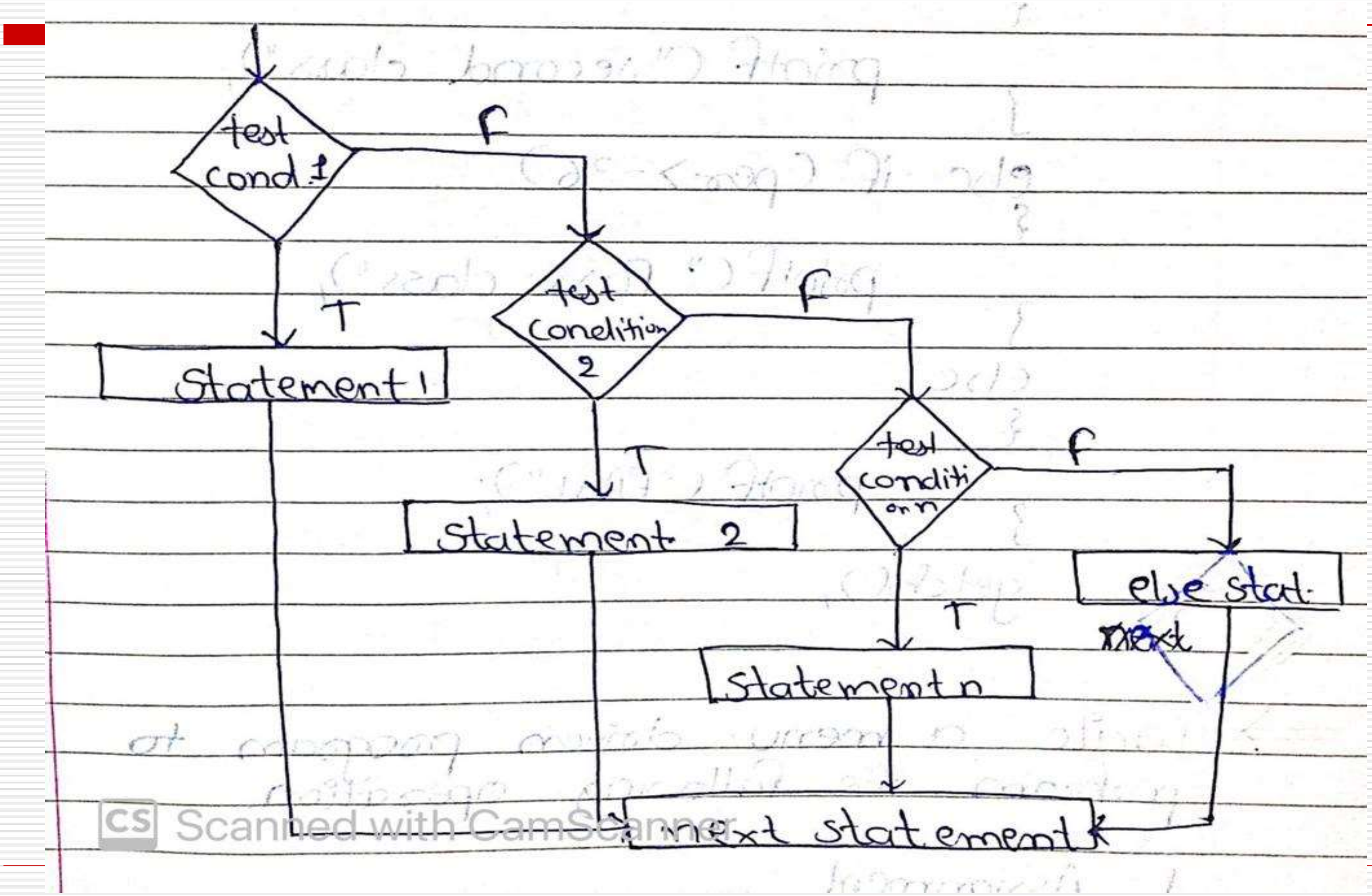
3.1.4. Ladder if or elseif Statement

- else-if statements in C is like another if condition, it's used in a program when if statement having multiple decisions.

Syntax:

```
if(test - condition 1)
{
   // Statement 1
}
else if(test - condition 2)
{
          // Statement 2
}
...
...
...
else if(test - condition n)
{
          // Statement n
}
else
{
          // Else Statement
}
next- statement
```

Flowchart:


test cond 1
F
T
Statement 1
test condition 2
F
T
statement 2
test condition n
F
T
statement n
else stat.
next
next statement

Example:

```
/*W A Menu Driven Program to perform the follwing task
1. Sunday
2. Monday
3. Tuesday
4. Wednesday
5. Thursday
6. Friday
7. Saturday   */
#include<stdio.h>
#include<conio.h>
```

```
void main()
{
        int c;
        clrscr();
        printf("\n1. Sunday");
        printf("\n2. Monday");
        printf("\n3. Tuesday");
        printf("\n4. Wednesday");
        printf("\n5. Thursday");
        printf("\n6. Friday");
        printf("\n7. Saturday");
        printf("\nEnter Your Choice:(1 to 7):");
        scanf("%d",&c);
        if(c==1)
                printf("Sunday");
        else if(c==2)
                printf("Monday");
        else if(c==3)
                printf("Tuesday");
        else
                printf("Invalid Chioce");
        getch();
}
```

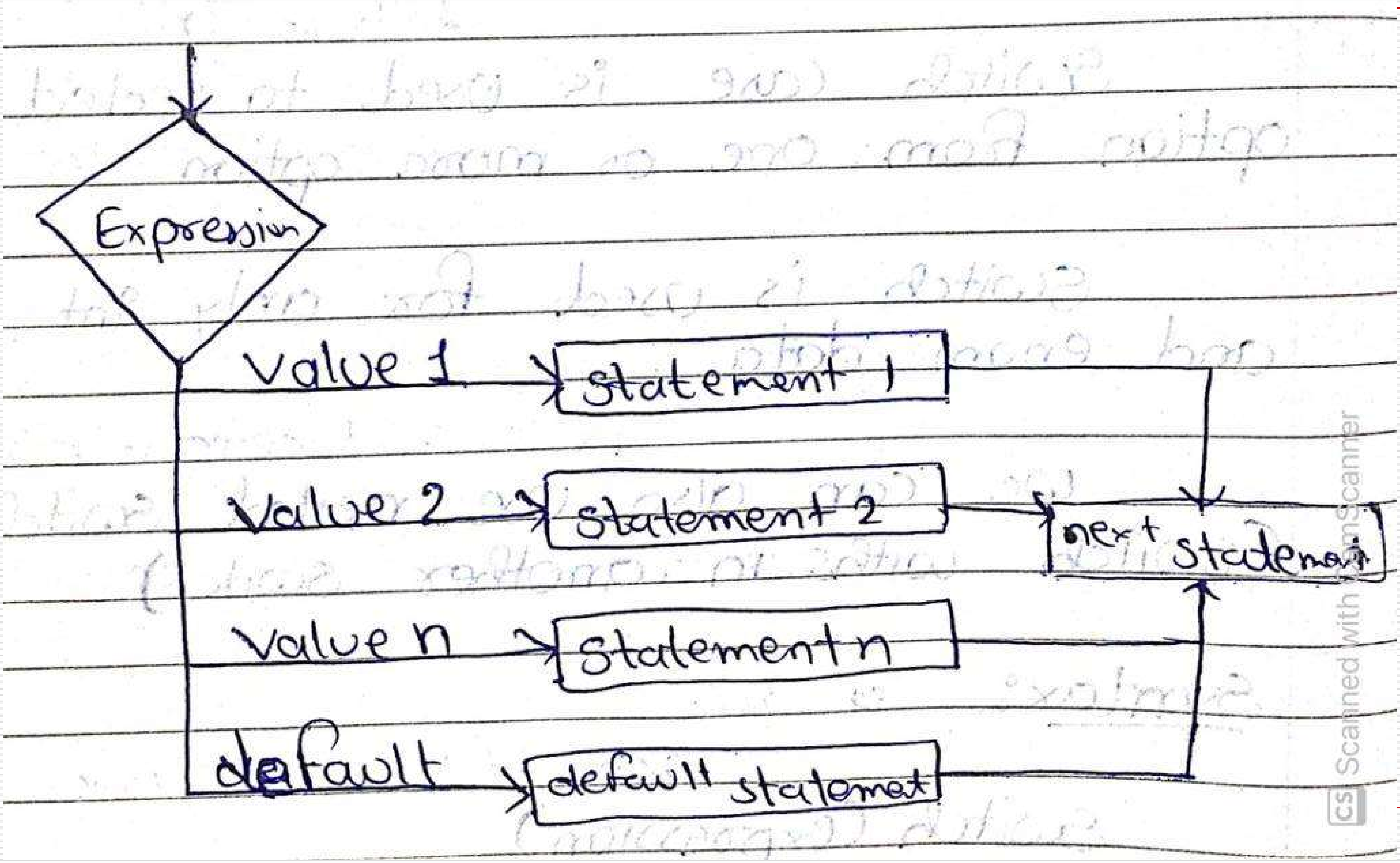
3.2 Switch Case or Selection Statement

- □ Switch case is used to select one option from one or more option.
- □ Switch is used for only int, char and enum data.
- □ We can also use nested switch (switch within switch).

Syntax:

```
switch(expression)
{
        case value1:
                Statement 1;
                break;
        case value2:
                Statement 2;
                break;
        …
        …
        …
        case valuen:
                Statement n;
                break;
        default:
                default Statement;
                break;
}
next- statement;
```

Flowchart:



Expression
value 1
statement 1
value 2
statement 2
value n
statement n
default
default statement
next statement

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    char c;
    clrscr();
    printf("Enter your choice:");
    scanf("%c",&c);
    switch(c)
    {
        case 'A':
                printf("Assignment");
                break;
        case 'B':
                printf("Quiz");
                break;
        case 'C':
                printf("Viva");
                break;
        default:
                printf("Invalid Choice");
                break;
    }
    getch();
}
```

Rules of Switch case.

- Case label should be unique.
- Case label must be ends with : (colon).
- Case label must have constant value or constant expression.
- Case label must be int, char or enum.
- Case label should not be floating point number.
- Default label is optional.

- break statement takes control out of loop.
- Two or more case may share one break operation.
- Nesting is also possible.
- Relational operations are not allowed in switch case.

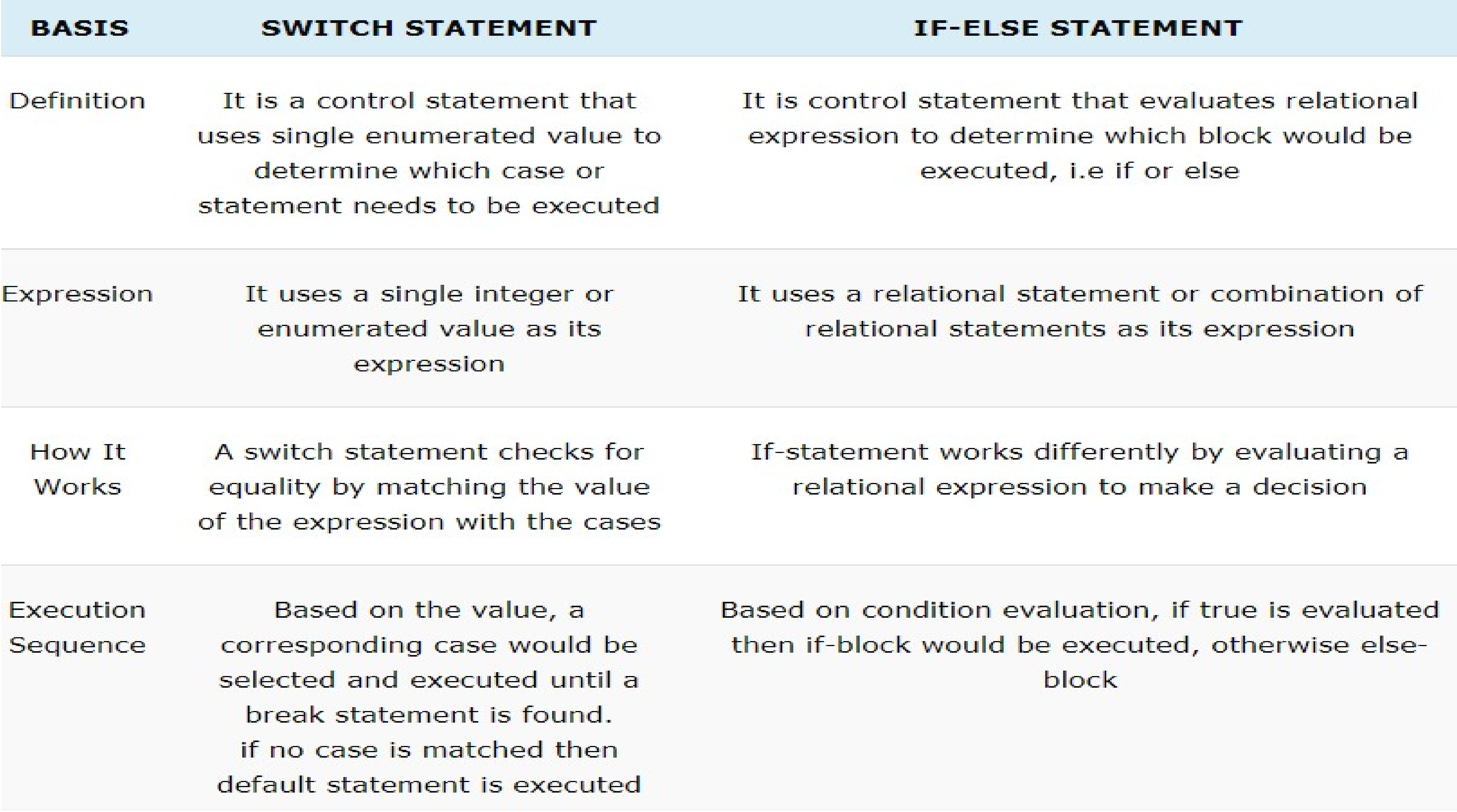
3.2.1 Use of break

- You can **use** the **break statement** to end processing of a particular labeled **statement** within the **switch statement**.
- It branches to the end of the **switch statement**. Without **break** , the program continues to the next labeled **statement**, executing the **statements** until a **break** or the end of the **statement** is reached.

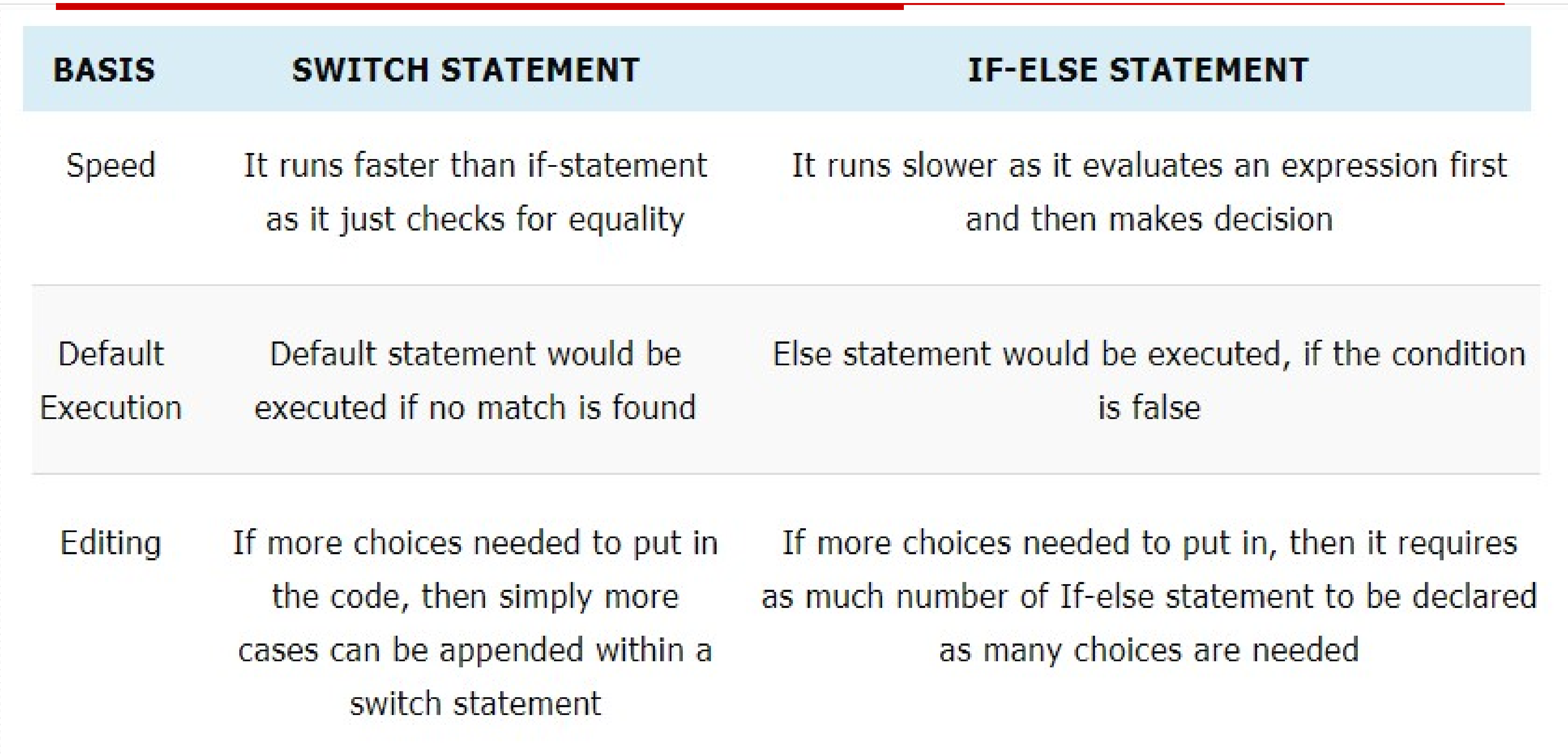
Use of default

- A **switch statement** can have an optional **default case**, which must appear at the end of the **switch**.
- The **default case** can be **used** for performing a task when none of the **cases** is true. No break is needed in the **default case**.

3.2.2 Difference between switch and if statements



BASIS	SWITCH STATEMENT	IF-ELSE STATEMENT
Definition	It is a control statement that uses single enumerated value to determine which case or statement needs to be executed	It is control statement that evaluates relational expression to determine which block would be executed, i.e if or else
Expression	It uses a single integer or enumerated value as its expression	It uses a relational statement or combination of relational statements as its expression
How It Works	A switch statement checks for equality by matching the value of the expression with the cases	If-statement works differently by evaluating a relational expression to make a decision
Execution Sequence	Based on the value, a corresponding case would be selected and executed until a break statement is found. if no case is matched then default statement is executed	Based on condition evaluation, if true is evaluated then if-block would be executed, otherwise else-block



BASIS	SWITCH STATEMENT	IF-ELSE STATEMENT
Speed	It runs faster than if-statement as it just checks for equality	It runs slower as it evaluates an expression first and then makes decision
Default Execution	Default statement would be executed if no match is found	Else statement would be executed, if the condition is false
Editing	If more choices needed to put in the code, then simply more cases can be appended within a switch statement	If more choices needed to put in, then it requires as much number of If-else statement to be declared as many choices are needed