

CL-1002
Programming
Fundamentals

LAB - 05
Decision Structure (If else, switch
case, ternary operator)

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES
Fall 2023

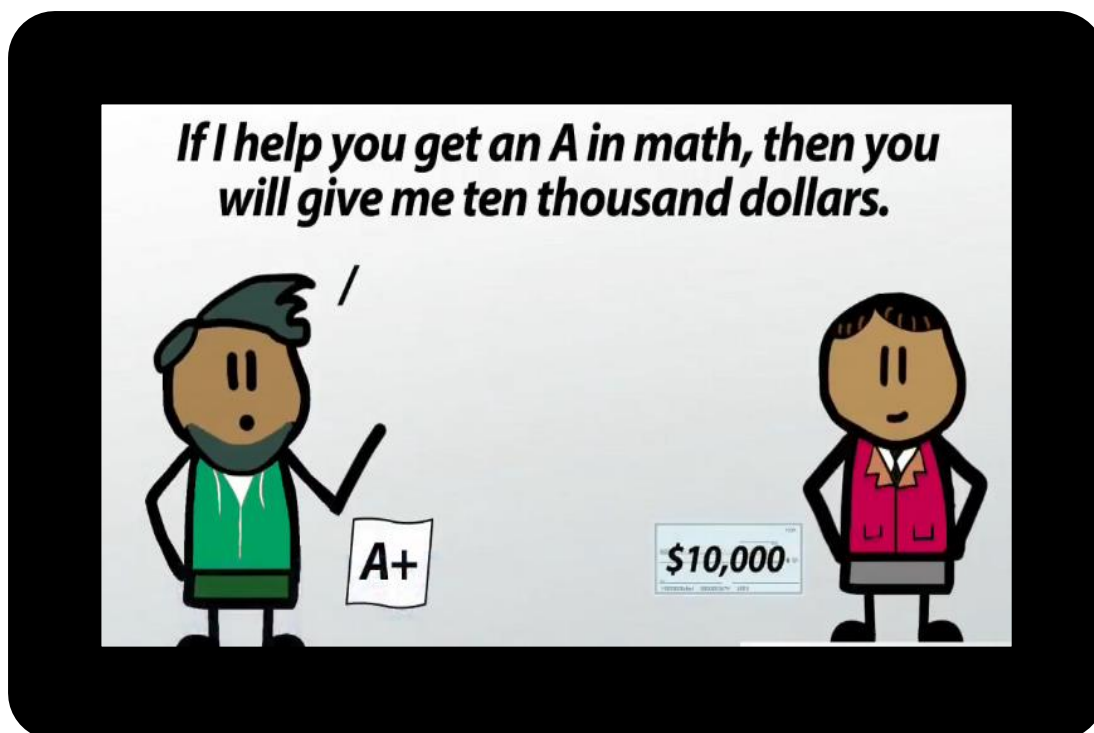
NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

Learning Objectives

- Introduction to conditional statements
- If structure
- If –else structure
- If-else-if structure
- Ternary Operator
- Switch statements

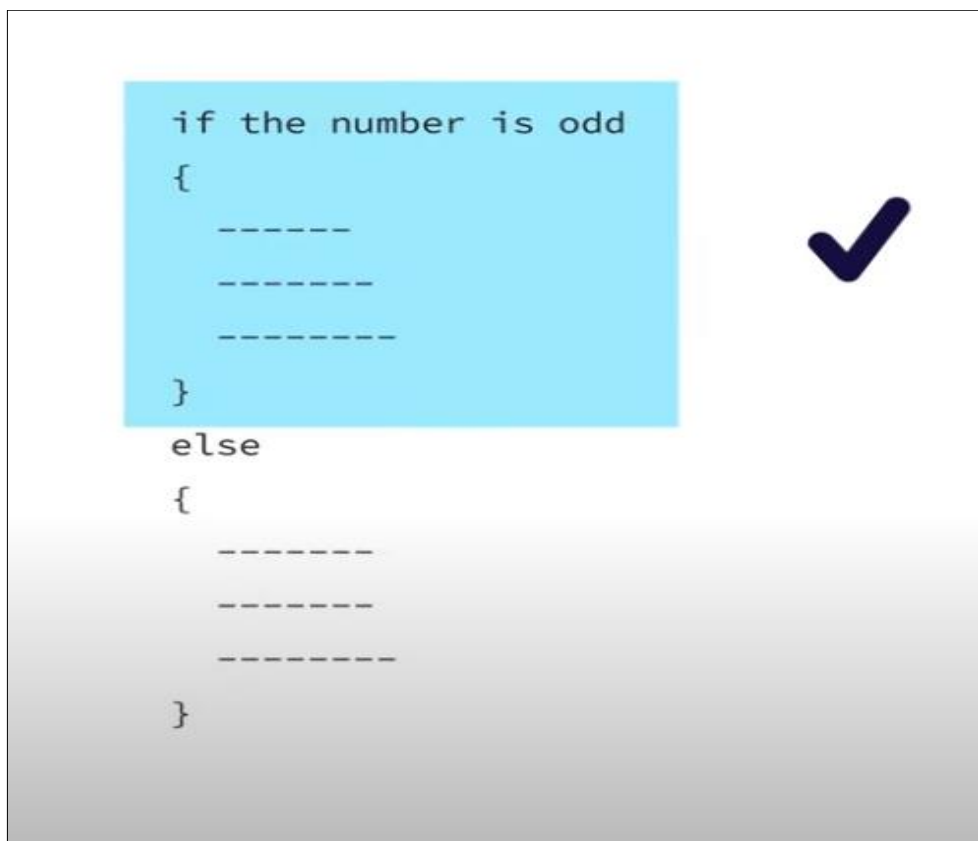
Conditional Statements

In C programming there are decision making statements. We need these kinds of statements because while programming we often need to make a lot of decisions. Like shown below in the pictures.

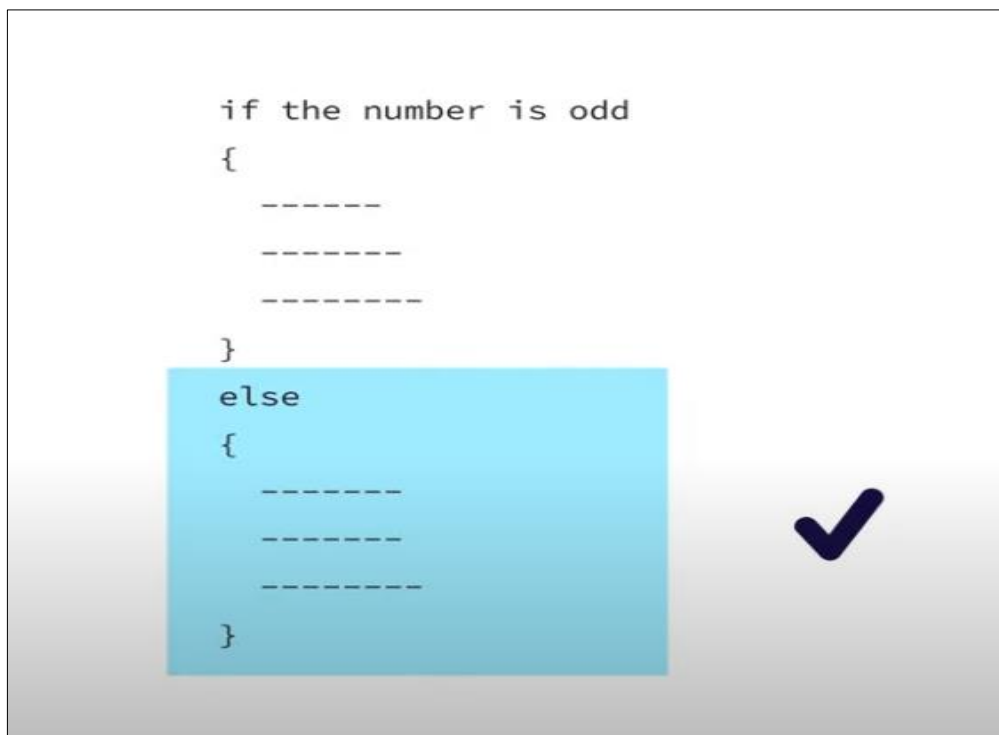




In some cases, we need to execute this block of code.



Otherwise, we want to execute this block.



In C, an if/else statement specifies that one block of code should be executed if a condition is true, and another block should be executed if that condition is false.

To write meaningful if/else statements, it is important to know operators which allow us to compare two expressions and produce a Boolean outcome.

In C, however, there are no distinct values for true or false, instead, false is 0, and anything which is non-zero is true. We will refer to true and false because they make more sense conceptually; the distinction should not make a practical difference in most cases.

In 'C' programming conditional statements are possible with the help of the following two constructs:

1. If statement
2. If-else statement

It is also called branching as a program decides which statement to execute based on the result of the evaluated condition.

If statement

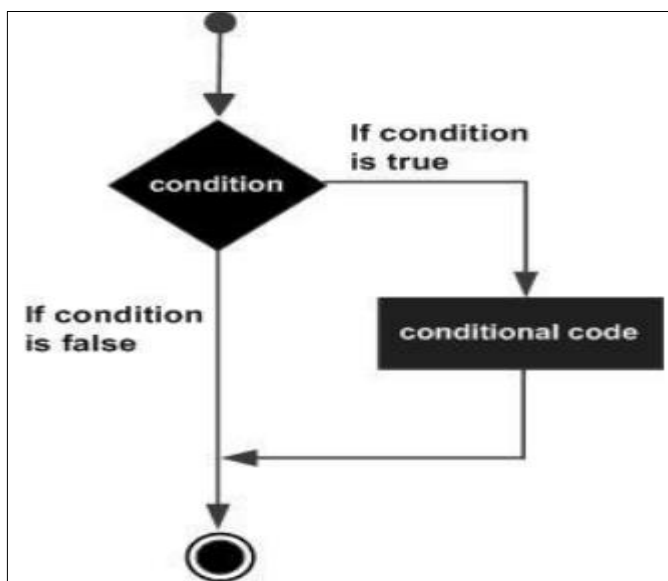
An if statement consists of a conditional expression followed by one or more statements.

If the conditional expression evaluates to true, then the block of code inside the if statement will be executed. If the conditional expression evaluates to be false, then the first set of code after the end of the if statement (after the closing curly brace) will be executed.

Syntax: The syntax of an if statement in C programming language is:

```
If (condition)
{
    //statements;
}
```

Flowchart:



Example: Checking if the number input by user is 0 or not. If it's 0 then print Zero else print non-zero

```
#include <stdio.h>
int main() {
    int num;
    printf ("Enter any number\n");
    scanf("%d", &num);
    if (num==0)
        printf("Zero");
    else
        printf("Non-zero");
    return 0;
}
```

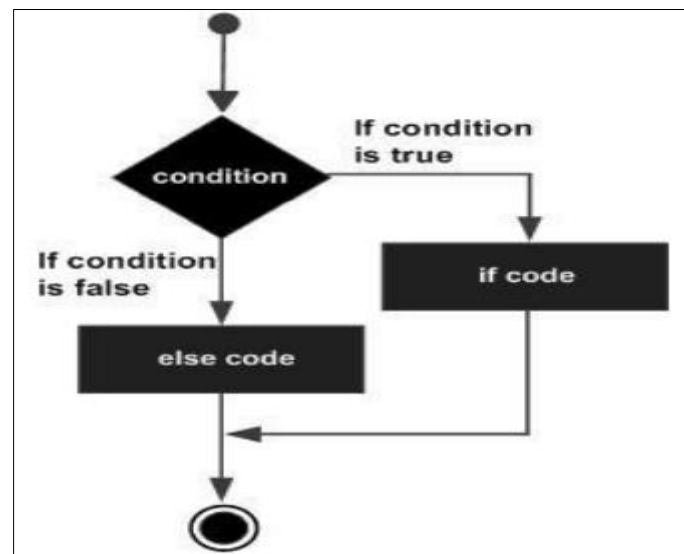
If else statement

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

Syntax: The syntax of an if...else statement in C programming language is:

```
If (condition)
{
    //statements;
}
else
{
    //statements;
}
```

Flowchart:



Example:

Problem

In FAST University 80% attendance is required for students to appear in the examination otherwise you won't be able to sit in an exam.

C Code

```
#include <stdio.h>
int main() {
    int attendance;
    printf ("Enter Attendance of your semeseter:\n");
    scanf ("%d", &attendance);
    if (attendance>80)
        printf ("You are eligible for the Examination");
    else
        printf ("Sorry! You are not for the Examination");
}
```

If else if statement

An **if** statement can be followed by an optional **else if...else** statement, which is very useful to test various conditions using single **if...else if** statement.

When using **if**, **else if**, **else** statements there are few points to keep in mind:

- An **if** can have zero or one else's and it must come after any **else if's**.
- An **if** can have zero to many **else if's** and they must come before the **else**.
- Once an **else if** succeeds, none of the remaining **else if's** or **else's** will be tested.

Syntax: The syntax of an if...else statement in C programming language is:

```
If (condition)
{
    //statements;
}
else if (condition)
{
    //statements;
}
else if (condition)
{
    //statements;
}
else
{
    //statements;
}
```


Example:

Problem FAST University wants to assign a grade to every PhD student according to the obtained marks. The grading criteria for PhD students is given below.

Ph.D.	Equivalent %
A+	90 & above
A	86
A-	82
B+	78
B	74
F	70
	66
	62
	58
	54
	50

Write a program to assign the grade to each student according to his marks.

```
#include<stdio.h>
void main()
{
    int marks;
    printf("Enter your marks ");
    scanf("%d",&marks);
    if(marks<0 || marks>100) {
        printf("Wrong Entry");
    }
    else if(marks>=90) {
        printf("Grade A+");
    }
    else if(marks>=86 && marks<90){
        printf("Grade A");
    }
    else if(marks>=82 && marks<86){
        printf("Grade A-");
    }
    else if(marks>=78 && marks<82){
        printf("Grade B+");
    }
    else if(marks>=74 && marks<78){
        printf("Grade B");
    }
    else{
        printf("Grade ");
    }
}
```

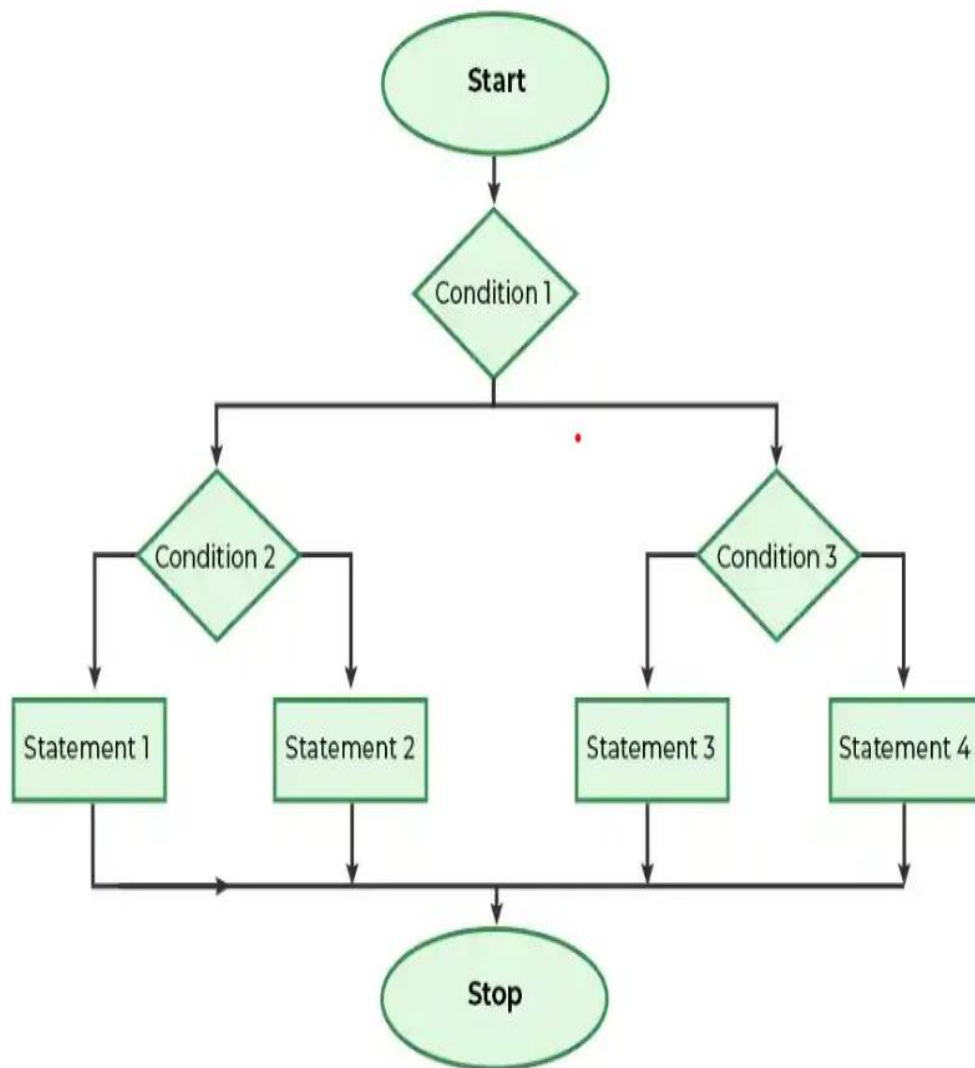
Nested If-else Statement

A nested if in C is an if statement that is the target of another if statement. Nested if statements mean an if statement inside another if statement.

Syntax: The syntax of a Nested if...else statement in C programming language is:

```
//check if the first condition holds
if (condition 1) {
    //if the second condition holds
    if (condition 2) {do something}
    //if the second condition does not hold
    else {do something else}
}
// if the first condition does not hold
else {
    //if the third condition holds
    if (condition 3) {do something}
    //if the third condition does not hold
    else {do something else}
}
```

Flowchart



Example:

Problem: write c program to analyze if the number is even or odd, and then if it is even, whether it is divisible by 4 or not, and if it is odd, whether it is divisible by 3 or not will be:

```
#include <stdio.h>
int main() {
    int n;
    printf("Enter The Number ");
    scanf("%d", &n);
    if (n % 2 == 0)
    {
        printf("Even ");
        if (n % 4 == 0) {
            printf("and divisible by 4");
        } else {
            printf("and not divisible by 4");
        }
    }
    else
    {
        printf("Odd ");
        if (n % 3 == 0) {
            printf("and divisible by 3");
        } else {
            printf("and not divisible by 3");
        }
    }
}
```

Ternary Operator

We use the ternary operator in C to run one code when the condition is true and another code when the condition is false.

Syntax.

The `testCondition` is a boolean expression that results in either **true** or **false**.

If the condition is

- `true` - **expression1** (before the colon) is executed
- `false` - **expression2** (after the colon) is executed

The ternary operator takes 3 operands (`condition, expression1 and expression2`).

Hence, the name **ternary operator**.

Example

Write a C program that takes a user's age as input and uses a ternary operator to print 'You can vote' if the age is 18 or older, and 'You cannot vote' otherwise.

```
#include <stdio.h>

int main() {
    int age;
    printf("Enter your age: ");
    scanf("%d", &age);

    // ternary operator to find if a person can vote or not
    (age >= 18) ? printf("You can vote") : printf("You cannot vote");

    return 0;
}
```

Switch Statement

Another way that programs can make decisions is to use switch/case. The syntax of switch/case is shown in the figure below.

Syntax:

```
switch (selection expression) {  
    case 1:  
        //statement  
        break;  
    case 2:  
        //statement  
        break;  
    default:  
        //statement  
}
```

Here, when the execution arrow reaches the switch statement, the selection expression—in parenthesis after the keyword switch—is evaluated to a value.

This value is then used to determine which case to enter. The execution arrow then jumps to the corresponding case—the one whose label (the constant immediately after the keyword case) matches the selection expression's value. If no label matches, then the execution arrow jumps to the default case if there is one, and to the closing curly brace of the switch if not.



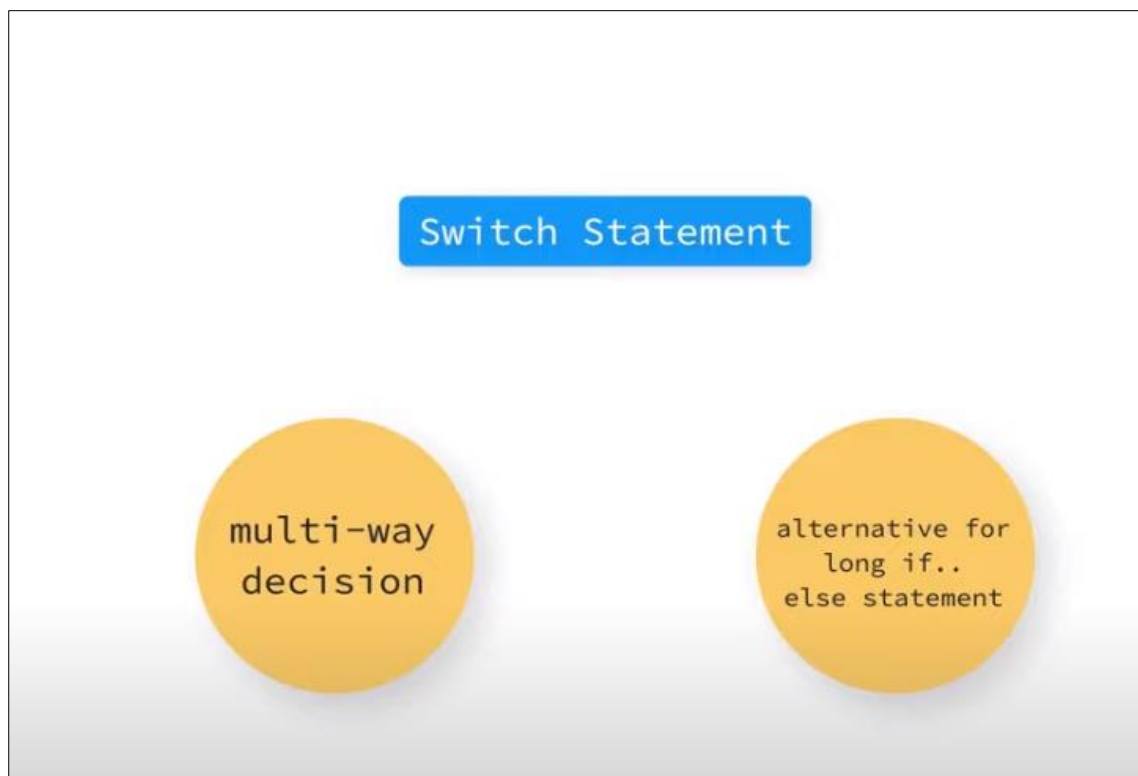
The following rules apply to a **switch** statement:

- The **expression** used in a **switch** statement must have an integral or enumerated type, or be of a class type in which the class has a single conversion function to an integral or enumerated type.
- You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon.

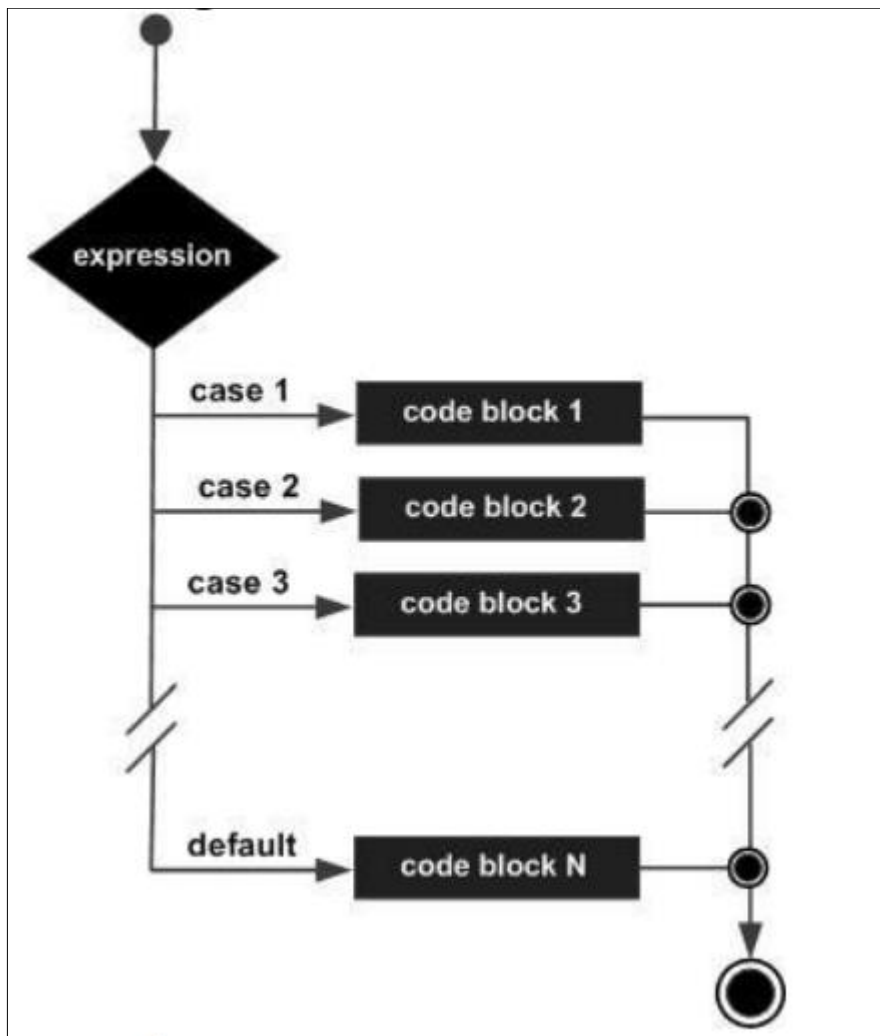
Switch statement is better than if else statement.

A switch statement is **usually more efficient than a set of nested ifs**

Switch statement acts as a substitute for a long **if-else-if** ladder that is used to test a list of cases.



Flowchart



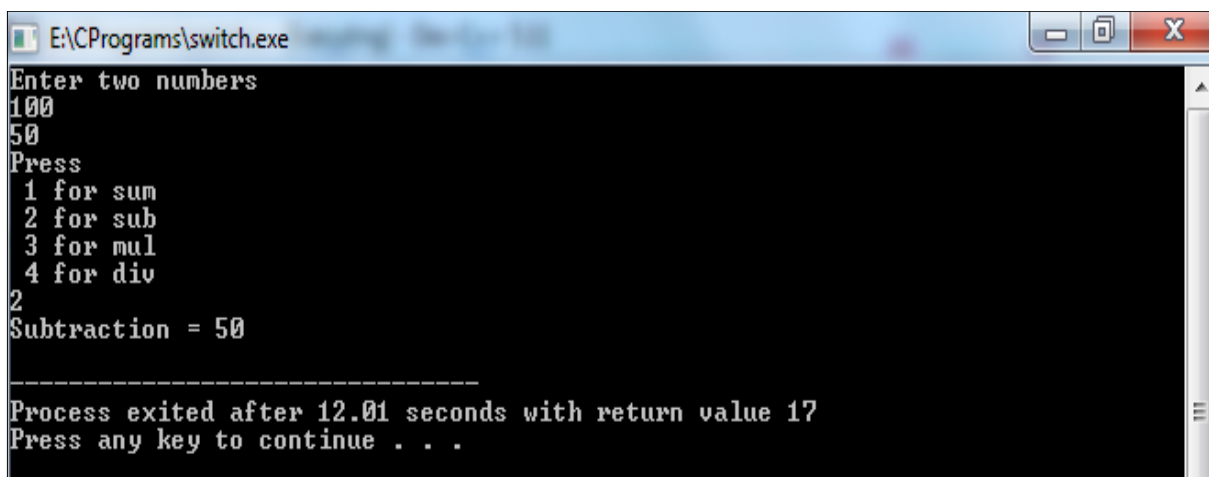
Example: Calculator using numbers to choose operators, if 1 is pressed then addition is performed, if 2 is pressed then subtraction if 3 is pressed then multiplication, if 4 is pressed then division otherwise print invalid choice.

```
#include <stdio.h>
int main() {
    int num1, num2, choice;
    printf ("Enter two numbers\n");
    scanf ("%d%d", &num1, &num2);

    printf("Press \n 1 for sum \n 2 for sub \n 3 for mul \n 4 for div\n");
    scanf ("%d", &choice);

    switch (choice) {
        case 1:
            printf("Sum = %d\n", num1 + num2);
            break;
        case 2:
            printf("Subtraction = %d\n", num1 - num2);
            break;
        case 3:
            printf("Multiplication = %d\n", num1 * num2);
            break;
        case 4:
            printf("Division = %d\n", num1 / num2);
            break;
        default:
            printf("Enter valid choice\n");
    }
}
```

Output:



```
E:\CPrograms\switch.exe
Enter two numbers
100
50
Press
1 for sum
2 for sub
3 for mul
4 for div
2
Subtraction = 50

-----
Process exited after 12.01 seconds with return value 17
Press any key to continue . . .
```

Nested-Switch Statement:

Nested-Switch statements refers to Switch statements inside of another Switch Statements.

Syntax

Syntax

The syntax for a **nested switch** statement is as follows -

```
switch(ch1) {  
  
    case 'A':  
        printf("This A is part of outer switch");  
  
        switch(ch2) {  
            case 'A':  
                printf("This A is part of inner switch");  
                break;  
            case 'B': /* case code */  
            }  
  
            break;  
            case 'B': /* case code */  
        }  
}
```

Example

C program that simulates a weekly schedule for a person. The program should take inputs for the day (1-7) and activity (1 for working from home, 2 for the office, 3 for the gym) and display the corresponding activity for that day. Use switch-case statements for the logic.

```
#include <stdio.h>
int main() {
    int day = 1; // Assuming it's Wednesday
    int activity = 3; // 1: Work from home, 2: Office, 3: Gym
    switch (day) {
        case 1:
            printf("Monday: ");
            switch (activity) {
                case 1:
                    printf("Working from home\n");
                    break;
                case 2:
                    printf("Office\n");
                    break;
                case 3:
                    printf("Gym in the evening\n");
                    break;
                default:
                    printf("Invalid input\n");
            }
            break;
        case 2:
            printf("Tuesday: ");
            switch (activity) {
                case 1:
                    printf("Working from home\n");
                    break;
                case 2:
                    printf("Office\n");
                    break;
                case 3:
                    printf("Gym in the evening\n");
                    break;
                default:
                    printf("Invalid input\n");
            }
            break;
        // Add cases for the remaining days here

        default:
            printf("Other day\n");
    }

    return 0;
}
```