EXERCISE 4

Content:

1. C if-else
2. C switch
3. if-else vs switch

I. C if-else

The if-else statement in C is used to perform the operations based on some specific condition. The operations specified in if block are executed if and only if the given condition is true.

There are the following variants of if statement in C language.

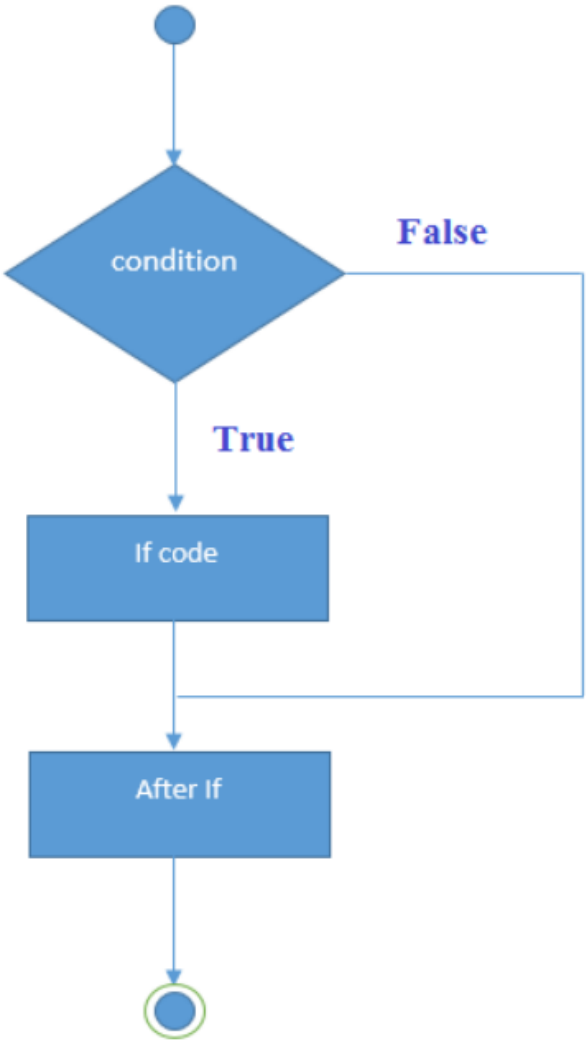
- If statement
- If-else statement
- If else-if ladder
- Nested if

1. If Statement

The if statement is used to check some given condition and perform some operations depending upon the correctness of that condition. It is mostly used in the scenario where we need to perform the different operations for the different conditions. The syntax of the if statement is given below.

```
if(expression){
//code to be executed
}
```

Flowchart of if statement in C:



LAB: Let's see a simple example of C language if statement.

C (Gcc 6.3)

```
1  #include<stdio.h>
2  int main(){
3  int number=0;
4  printf("Enter a number:");
5  scanf("%d",&number);
6  if(number%2==0){
7  printf("%d is even number",number);
8  }
9  return 0;
10 }
```

Result/Output:

Custom Input

```
40
50
```

Status Successfully executed Date 2022-02-04 08:59:38 Time 0.008621 sec Mem 5.372 kB

Input

```
40
50
```

Output

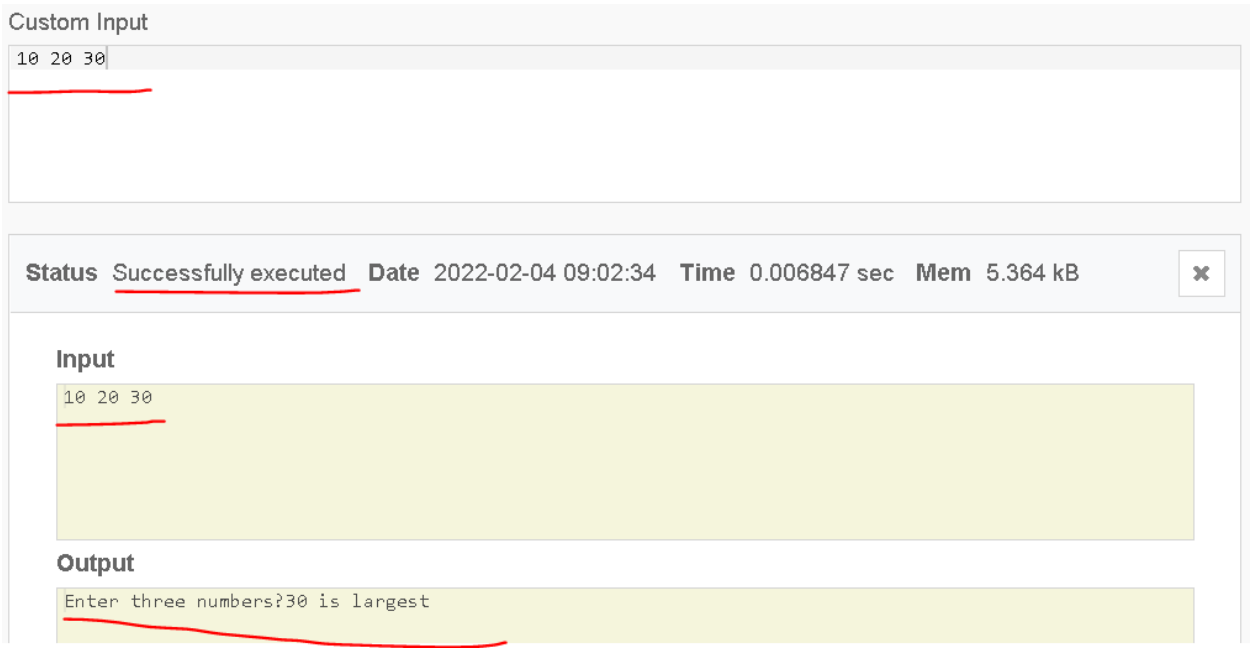
```
Enter a number:40 is even number
```

LAB: Program to find the largest number of the three.

C (Gcc 6.3)

```c
#include <stdio.h>
int main()
{
    int a, b, c;
     printf("Enter three numbers?");
    scanf("%d %d %d",&a,&b,&c);
    if(a>b && a>c)
    {
        printf("%d is largest",a);
    }
    if(b>a  && b > c)
    {
        printf("%d is largest",b);
    }
    if(c>a && c>b)
    {
        printf("%d is largest",c);
    }
    if(a == b && a == c)
    {
        printf("All are equal");
    }
}
```

Result/Output:

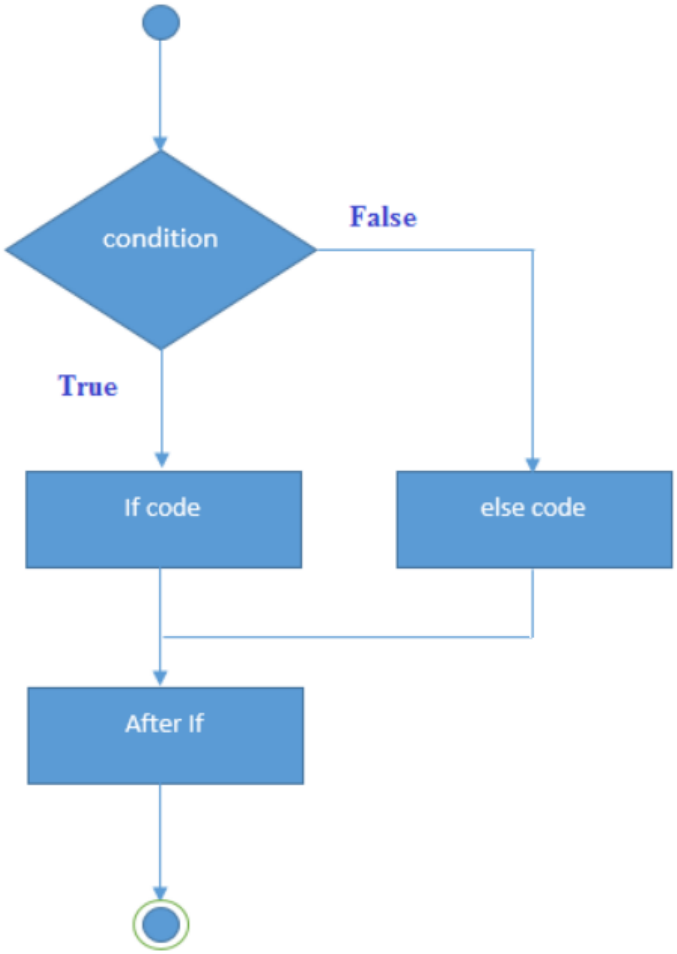


2. If-else Statement

The if-else statement is used to perform two operations for a single condition. The if-else statement is an extension to the if statement using which, we can perform two different operations, i.e., one is for the correctness of that condition, and the other is for the incorrectness of the condition. Here, we must notice that if and else block cannot be executed simiulteneously. Using if-else statement is always preferable since it always invokes an otherwise case with every if condition. The syntax of the if-else statement is given below.

```
if(expression){
//code to be executed if condition is true
}else{
//code to be executed if condition is false
}
```

Flowchart of the if-else statement in C:

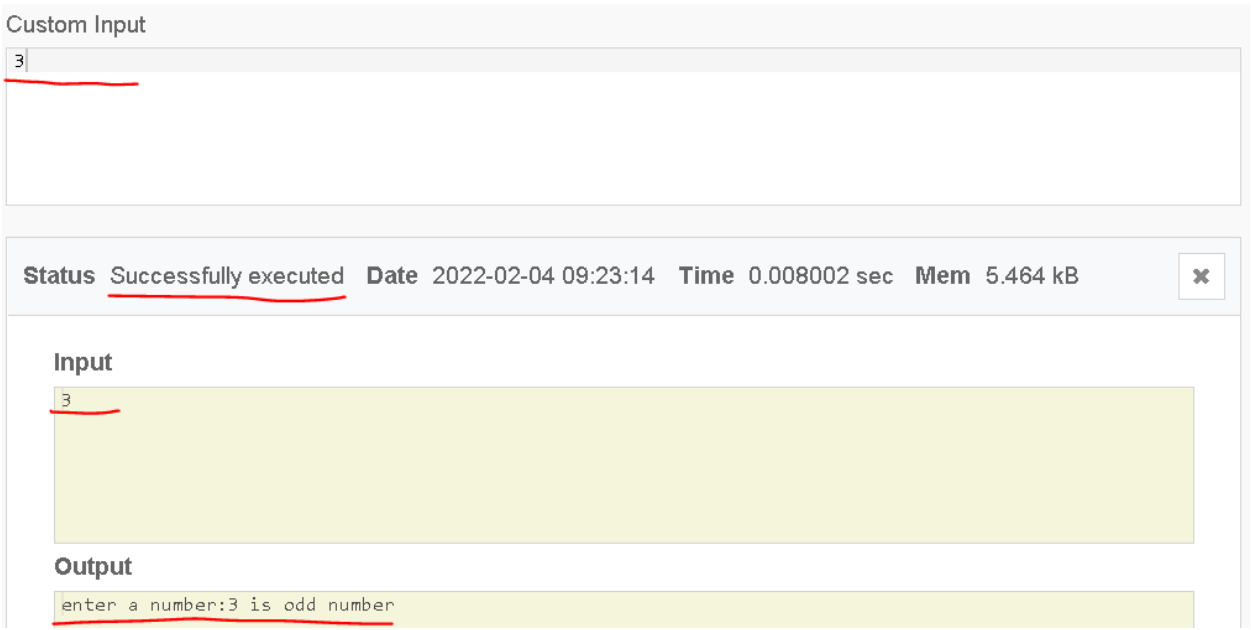


LAB: Let's see the simple example to check whether a number is even or odd using if-else statement in C language.

C (Gcc 6.3)

```c
#include<stdio.h>
int main(){
int number=0;
printf("enter a number:");
scanf("%d",&number);
if(number%2==0){
printf("%d is even number",number);
}
else{
printf("%d is odd number",number);
}
return 0;
}
```

Result/Output:

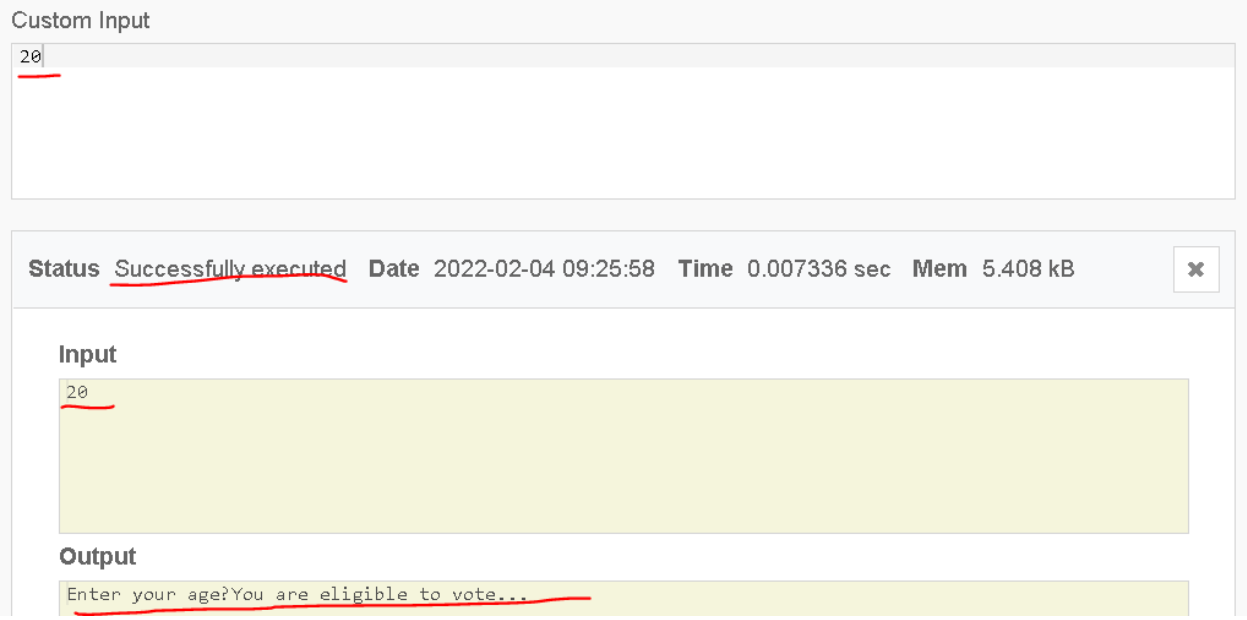


LAB: Program to check whether a person is eligible to vote or not.

C (Gcc 6.3)

```c
#include <stdio.h>
int main()
{
    int age;
    printf("Enter your age?");
    scanf("%d",&age);
    if(age>=18)
    {
        printf("You are eligible to vote...");
    }
    else
    {
        printf("Sorry ... you can't vote");
    }
}
```

Result/Output:



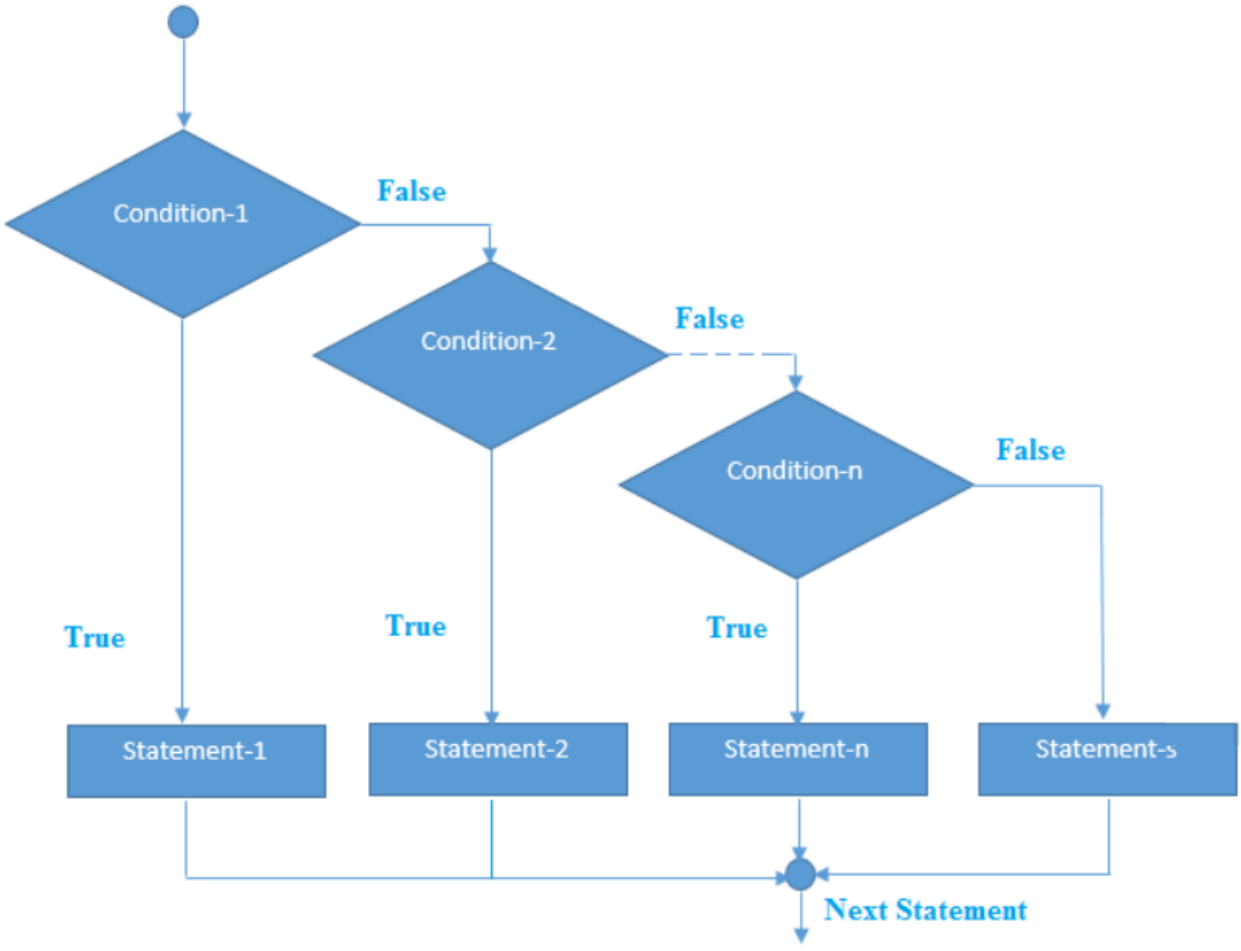
3. If else-if ladder Statement

The if-else-if ladder statement is an extension to the if-else statement. It is used in the scenario where there are multiple cases to be performed for different conditions. In if-else-if ladder statement, if a condition is true then the statements defined in the if block will be executed, otherwise if some other condition is true then the statements defined in the else-if block will be executed, at the last if none of the condition is true then the statements defined in the else block will be executed. There are multiple else-if blocks possible. It is similar to the switch case statement where the default is executed instead of else block if none of the cases is matched.

```
if(condition1){
//code to be executed if condition1 is true
}else if(condition2){
//code to be executed if condition2 is true
}
else if(condition3){
//code to be executed if condition3 is true
}
...
```

```
else{
//code to be executed if all the conditions are false
}
```

Flowchart of else-if ladder statement in C:



LAB: An example of an if-else-if statement.

Step 1: Type the following code in CodeChef IDE:

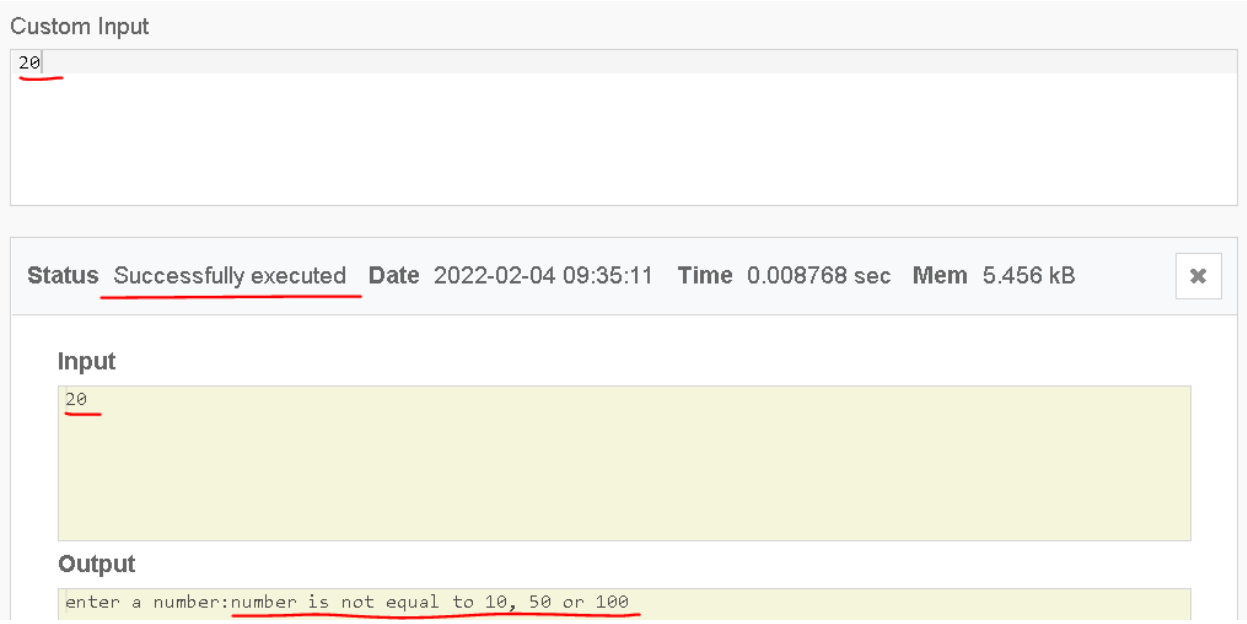
C (Gcc 6.3)

```c
#include<stdio.h>
int main(){
int number=0;
printf("enter a number:");
scanf("%d",&number);
if(number==10){
printf("number is equals to 10");
}
else if(number==50){
printf("number is equal to 50");
}
else if(number==100){
printf("number is equal to 100");
}
else{
printf("number is not equal to 10, 50 or 100");
}
return 0;
}
```

Step 2: Put a number in the **Custom input** field.

Step 3: Execute the program by Run button.

Step 4: Check the Result/Output:



LAB: Create a program that calculate the grade of the student according to the specified marks.

C (Gcc 6.3)

```c
#include <stdio.h>
int main()
{
    int marks;
    printf("Enter your marks?");
    scanf("%d",&marks);
    if(marks > 85 && marks <= 100)
    {
        printf("Congrats ! you scored grade A ...");
    }
    else if (marks > 60 && marks <= 85)
    {
        printf("You scored grade B + ...");
    }
    else if (marks > 40 && marks <= 60)
    {
        printf("You scored grade B ...");
    }
    else if (marks > 30 && marks <= 40)
    {
        printf("You scored grade C ...");
    }
    else
    {
        printf("Sorry you are fail ...");
    }
}
```

Result/Output:

Custom Input

```
88
```

Status Successfully executed **Date** 2022-02-04 09:40:19 **Time** 0.006146 sec **Mem** 5.392 kB

Input

```
88
```

Output

```
Enter your marks?Congrats ! you scored grade A ...
```

II. C switch

The switch statement in C is an alternate to if-else-if ladder statement which allows us to execute multiple operations for the different possibles values of a single variable called switch variable. Here, We can define various statements in the multiple cases for the different values of a single variable.

The syntax of switch statement in c language is given below:

```
switch(expression){
case value1:
 //code to be executed;
 break;  //optional
case value2:
 //code to be executed;
 break;  //optional
......

default:
 code to be executed if all cases are not matched;
}
```

Rules for switch statement in C language:

1) The switch expression must be of an integer or character type.

2) The case value must be an integer or character constant.

3) The case value can be used only inside the switch statement.

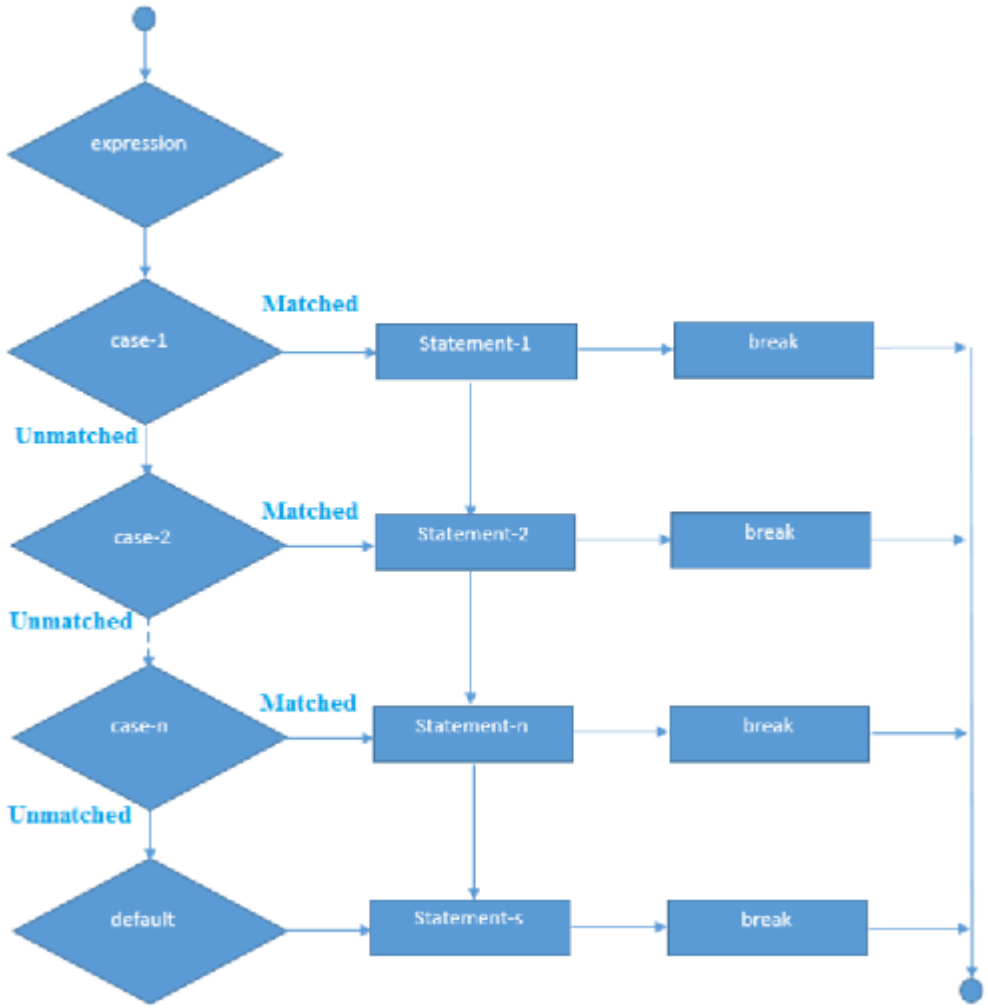
4) The break statement in switch case is not must. It is optional. If there is no break statement found in the case, all the cases will be executed present after the matched case. It is known as fall through the state of C switch statement.

Let's try to understand it by the examples. We are assuming that there are following variables.

```
int x,y,z;
char a,b;
float f;
```

Valid Switch	Invalid Switch	Valid Case	Invalid Case
switch(x)	switch(f)	case 3;	case 2.5;
switch(x>y)	switch(x+2.5)	case 'a';	case x;
switch(a+b-2)		case 1+2;	case x+2;
switch(func(x,y))		case 'x'>'y';	case 1,2,3;

Flowchart of switch statement in C:



Functioning of switch case statement

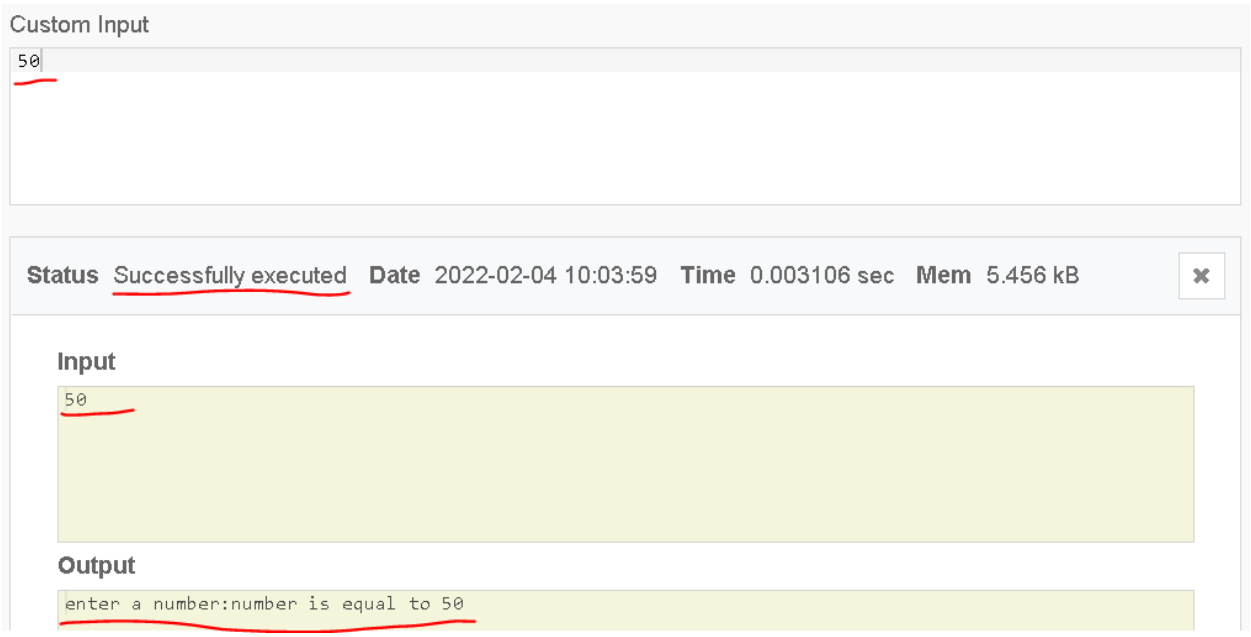
First, the integer expression specified in the switch statement is evaluated. This value is then matched one by one with the constant values given in the different cases. If a match is found, then all the statements specified in that case are executed along with the all the cases present after that case including the default statement. No two cases can have similar values. If the matched case contains a break statement, then all the cases present after that will be skipped, and the control comes out of the switch. Otherwise, all the cases following the matched case will be executed.

LAB: Let's see a simple example of c language switch statement.

C (Gcc 6.3)

```c
#include<stdio.h>
int main(){
int number=0;
printf("enter a number:");
scanf("%d",&number);
switch(number){
case 10:
printf("number is equals to 10");
break;
case 50:
printf("number is equal to 50");
break;
case 100:
printf("number is equal to 100");
break;
default:
printf("number is not equal to 10, 50 or 100");
}
return 0;
}
```

Result/Output:



C Switch statement is fall-through

In C language, the switch statement is fall through; it means if you don't use a break statement in the switch case, all the cases after the matching case will be executed.

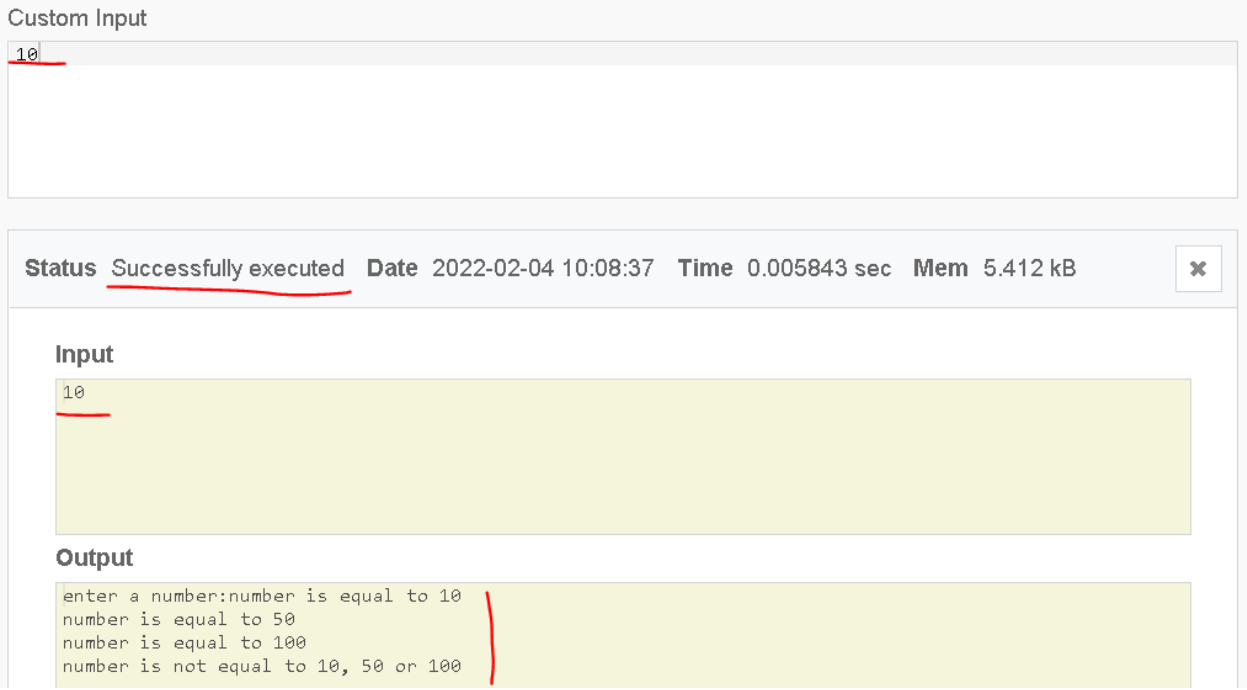
Let's try to understand the fall through state of switch statement by the example given below.

LAB: Type the following code in CodeChef IDE and execute the program:

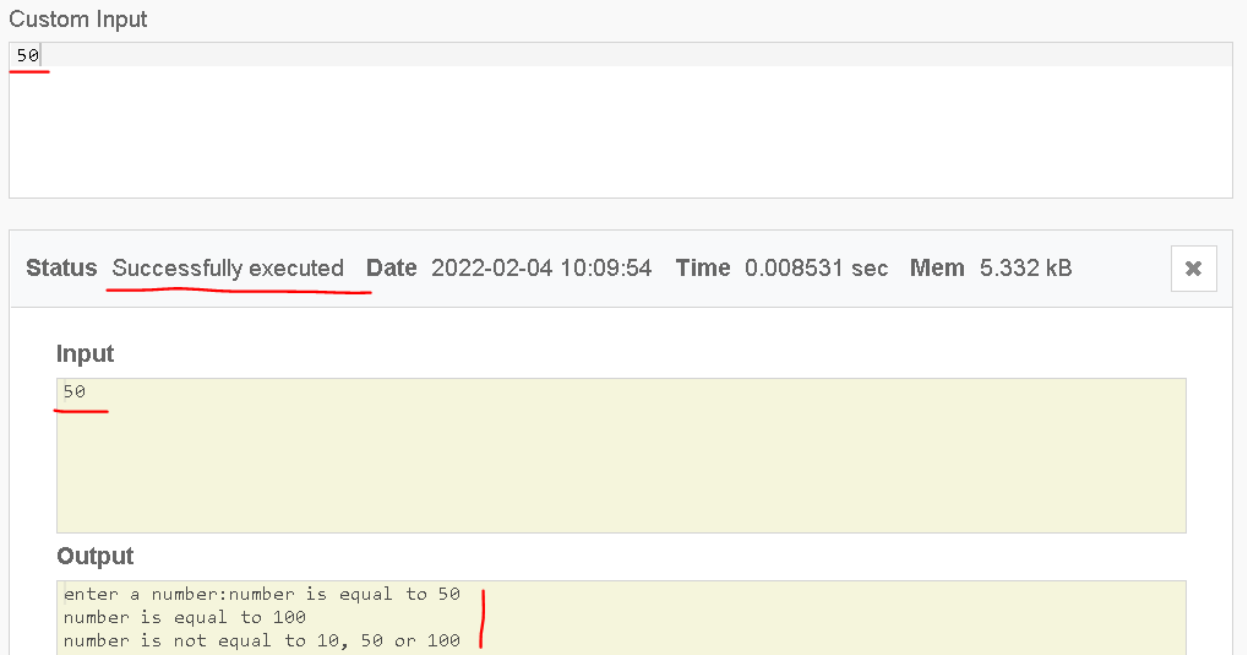
C (Gcc 6.3)

```c
#include<stdio.h>
int main(){
int number=0;

printf("enter a number:");
scanf("%d",&number);

switch(number){
case 10:
printf("number is equal to 10\n");
case 50:
printf("number is equal to 50\n");
case 100:
printf("number is equal to 100\n");
default:
printf("number is not equal to 10, 50 or 100");
}
return 0;
}
```

Result/Output if you enter number 10:



Result/Output if you enter number 50:



Nested switch case statement

We can use as many switch statement as we want inside a switch statement. Such type of statements is called nested switch case statements. Consider the following example.

C (Gcc 6.3)

```c
#include <stdio.h>
int main () {

   int i = 10;
   int j = 20;

   switch(i) {

      case 10:
         printf("the value of i evaluated in outer switch: %d\n",i);
      case 20:
         switch(j) {
            case 20:
               printf("The value of j evaluated in nested switch: %d\n",j);
         }
   }

   printf("Exact value of i is : %d\n", i );
   printf("Exact value of j is : %d\n", j );

   return 0;
}
```

Result/Output:

Status Successfully executed Date 2022-02-04 10:11:48 Time 0.005793 sec Mem 5.364 kB

Output

```
the value of i evaluated in outer switch: 10
The value of j evaluated in nested switch: 20
Exact value of i is : 10
Exact value of j is : 20
```

III. if-else vs switch

What is an if-else statement?

An if-else statement in C programming is a conditional statement that executes a different set of statements based on the condition that is true or false. The 'if' block will be executed only when the specified condition is true, and if the specified condition is false, then the else block will be executed.

What is a switch statement?

A switch statement is a conditional statement used in C programming to check the value of a variable and compare it with all the cases. If the value is matched with any case, then its corresponding statements will be executed.

Each case has some name or number known as the identifier. The value entered by the user will be compared with all the cases until the case is found. If the value entered by the user is not matched with any case, then the default statement will be executed.

Similarity b/w if-else and switch

Both the if-else and switch are the decision-making statements. Here, decision-making statements mean that the output of the expression will decide which statements are to be executed.

Differences b/w if-else and switch statement

The following are the differences between if-else and switch statement are:

- **Definition**

if-else

Based on the result of the expression in the 'if-else' statement, the block of statements will be executed. If the condition is true, then the 'if' block will be executed otherwise 'else' block will execute.

Switch statement

The switch statement contains multiple cases or choices. The user will decide the case, which is to execute.

- **Expression**

If-else

It can contain a single expression or multiple expressions for multiple choices. In this, an expression is evaluated based on the range of values or conditions. It checks both equality and logical expressions.

Switch

It contains only a single expression, and this expression is either a single integer object or a string object. It checks only equality expression.

- **Evaluation**

If-else

An if-else statement can evaluate almost all the types of data such as integer, floating-point, character, pointer, or Boolean.

Switch

A switch statement can evaluate either an integer or a character.

- **Sequence of Execution**

If-else

In the case of 'if-else' statement, either the 'if' block or the 'else' block will be executed based on the condition.

Switch

In the case of the 'switch' statement, one case after another will be executed until the break keyword is not found, or the default statement is executed.

- **Default Execution**

If-else

If the condition is not true within the 'if' statement, then by default, the else block statements will be executed.

Switch

If the expression specified within the switch statement is not matched with any of the cases, then the default statement, if defined, will be executed.

- **Values**

If-else

Values are based on the condition specified inside the 'if' statement. The value will decide either the 'if' or 'else' block is to be executed.

Switch

In this case, value is decided by the user. Based on the choice of the user, the case will be executed.

- **Use**

If-else

It evaluates a condition to be true or false.

Switch

A switch statement compares the value of the variable with multiple cases. If the value is matched with any of the cases, then the block of statements associated with this case will be executed.

- **Editing**

If-else

Editing in 'if-else' statement is not easy as if we remove the 'else' statement, then it will create the havoc.

Switch

Editing in switch statement is easier as compared to the 'if-else' statement. If we remove any of the cases from the switch, then it will not interrupt the execution of other cases. Therefore, we can say that the switch statement is easy to modify and maintain.

- **Speed**

If-else

If the choices are multiple, then the speed of the execution of 'if-else' statements is slow.

Switch

The case constants in the switch statement create a jump table at the compile time. This jump table chooses the path of the execution based on the value of the expression. If we have a multiple choice, then the execution of the switch statement will be much faster than the equivalent logic of 'if-else' statement.

Let's summarize the above differences in a table:

	If-else	switch
Definition	Depending on the condition in the 'if' statement, 'if' and 'else' blocks are executed.	The user will decide which statement is to be executed.
Expression	It contains either logical or equality expression.	It contains a single expression which can be either a character or integer variable.

Evaluation	It evaluates all types of data, such as integer, floating-point, character or Boolean.	It evaluates either an integer, or character.
Sequence of execution	First, the condition is checked. If the condition is true then 'if' block is executed otherwise 'else' block	It executes one case after another till the break keyword is not found, or the default statement is executed.
Default execution	If the condition is not true, then by default, else block will be executed.	If the value does not match with any case, then by default, default statement is executed.
Editing	Editing is not easy in the 'if-else' statement.	Cases in a switch statement are easy to maintain and modify. Therefore, we can say that the removal or editing of any case will not interrupt the execution of other cases.
Speed	If there are multiple choices implemented through 'if-else', then the speed of the execution will be slow.	If we have multiple choices then the switch statement is the best option as the speed of the execution will be much higher than 'if-else'.