C if else Statement

The if statement in C language is used to perform operation on the basis of condition. By using if-else statement, you can perform operation either condition is true or false.

There are many ways to use if statement in C language:

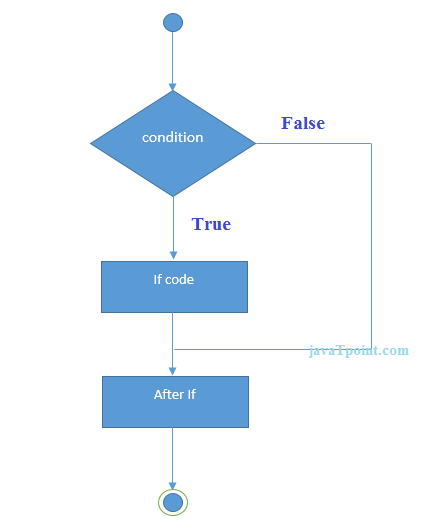
* If statement
* If-else statement
* If else-if ladder
* Nested if

If Statement

The single if statement in C language is used to execute the code if condition is true. The syntax of if statement is given below:

1. **if**(expression){
2. //code to be executed
3. }

**Flowchart of if statement in C**



Let's see a simple example of c language if statement.

1. #include<stdio.h>
2. #include<conio.h>
3. **void** main(){
4. **int** number=0;
5. clrscr();
7. printf("enter a number:");
8. scanf("%d",&number);
10. **if**(number%2==0){
11. printf("%d is even number",number);
12. }
14. getch();
15. }

**Output**

enter a number:4

4 is even number

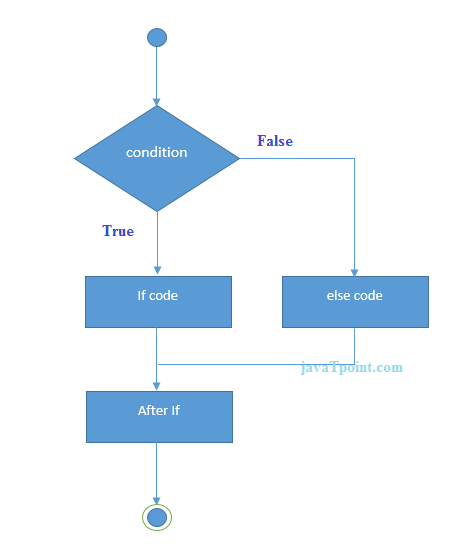
enter a number:5

If-else Statement

The if-else statement in C language is used to execute the code if condition is true or false. The syntax of if-else statement is given below:

1. **if**(expression){
2. //code to be executed if condition is true
3. }**else**{
4. //code to be executed if condition is false
5. }

**Flowchart of if-else statement in C**



Let's see the simple example of even and odd number using if-else statement in C language.

1. #include<stdio.h>
2. #include<conio.h>
3. **void** main(){
4. **int** number=0;
5. clrscr();
7. printf("enter a number:");
8. scanf("%d",&number);
10. **if**(number%2==0){
11. printf("%d is even number",number);
12. }
13. **else**{
14. printf("%d is odd number",number);
15. }
16. getch();
17. }

**Output**

enter a number:4

4 is even number

enter a number:5

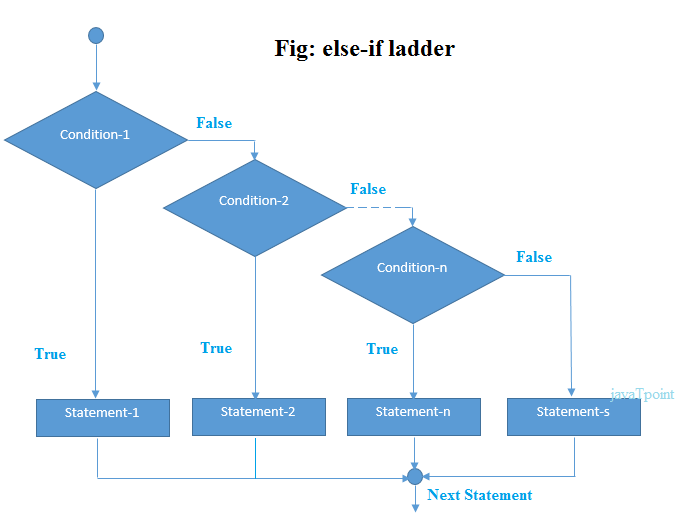
5 is odd number

If else-if ladder Statement

The if else-if statement is used to execute one code from multiple conditions. The syntax of if else-if statement is given below:

1. **if**(condition1){
2. //code to be executed if condition1 is true
3. }**else** **if**(condition2){
4. //code to be executed if condition2 is true
5. }
6. **else** **if**(condition3){
7. //code to be executed if condition3 is true
8. }
9. ...
10. **else**{
11. //code to be executed if all the conditions are false
12. }

**Flowchart of else-if ladder statement in C**



The example of if-else-if statement in C language is given below.

1. #include<stdio.h>
2. #include<conio.h>
3. **void** main(){
4. **int** number=0;
5. clrscr();
7. printf("enter a number:");
8. scanf("%d",&number);
10. **if**(number==10){
11. printf("number is equals to 10");
12. }
13. **else** **if**(number==50){
14. printf("number is equal to 50");
15. }
16. **else** **if**(number==100){
17. printf("number is equal to 100");
18. }
19. **else**{
20. printf("number is not equal to 10, 50 or 100");
21. }
22. getch();
23. }

**Output**

enter a number:4

number is not equal to 10, 50 or 100

enter a number:50

number is equal to 50

# C Switch Statement

The switch statement in C language is used *to execute the code from multiple conditions*. It is like if else-if ladder statement.

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

1. **switch**(expression){
2. **case** value1:
3. //code to be executed;
4. **break**;  //optional
5. **case** value2:
6. //code to be executed;
7. **break**;  //optional
8. ......
10. **default**:
11. code to be executed **if** all cases are not matched;
12. }

### **Rules for switch statement in C language**

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

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

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

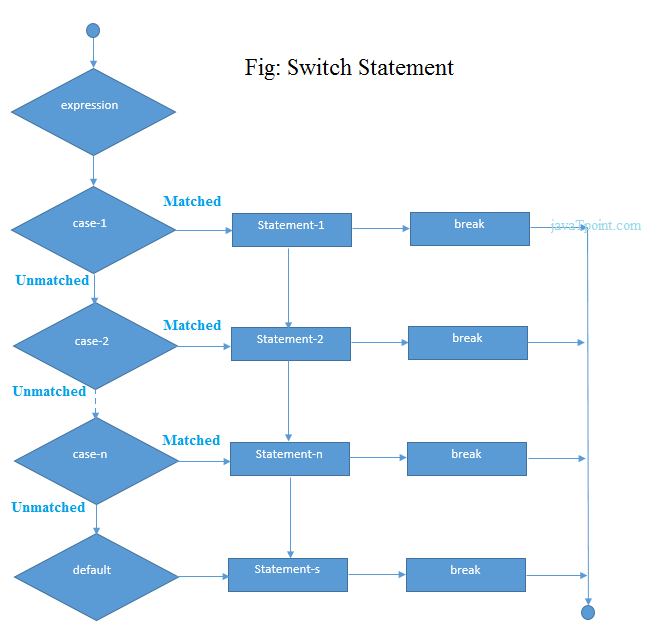
4) The *break statement* in switch case is not must. It is optional. If there is no break statement found in switch case, all the cases will be executed after matching the case value. It is known as *fall through* state of C switch statement.

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

1. **int** x,y,z;
2. **char** a,b;
3. **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



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

1. #include<stdio.h>
2. #include<conio.h>
3. **void** main(){
4. **int** number=0;
5. clrscr();
7. printf("enter a number:");
8. scanf("%d",&number);
10. **switch**(number){
11. **case** 10:
12. printf("number is equals to 10");
13. **break**;
14. **case** 50:
15. printf("number is equal to 50");
16. **break**;
17. **case** 100:
18. printf("number is equal to 100");
19. **break**;
20. **default**:
21. printf("number is not equal to 10, 50 or 100");
22. }
23. getch();
24. }

#### Output

enter a number:4

number is not equal to 10, 50 or 100

enter a number:50

number is equal to 50

#### C Switch statement is fall-through

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

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

1. #include<stdio.h>
2. #include<conio.h>
3. **void** main(){
4. **int** number=0;
5. clrscr();
7. printf("enter a number:");
8. scanf("%d",&number);
10. **switch**(number){
11. **case** 10:
12. printf("number is equals to 10\n");
13. **case** 50:
14. printf("number is equal to 50\n");
15. **case** 100:
16. printf("number is equal to 100\n");
17. **default**:
18. printf("number is not equal to 10, 50 or 100");
19. }
20. getch();
21. }

#### Output

enter a number:10

number is equals to 10

number is equals to 50

number is equals to 100

number is not equal to 10, 50 or 100

enter a number:50

number is equal to 50

number is equals to 100

number is not equal to 10, 50 or 100