Control Statements

C if else Statement

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.

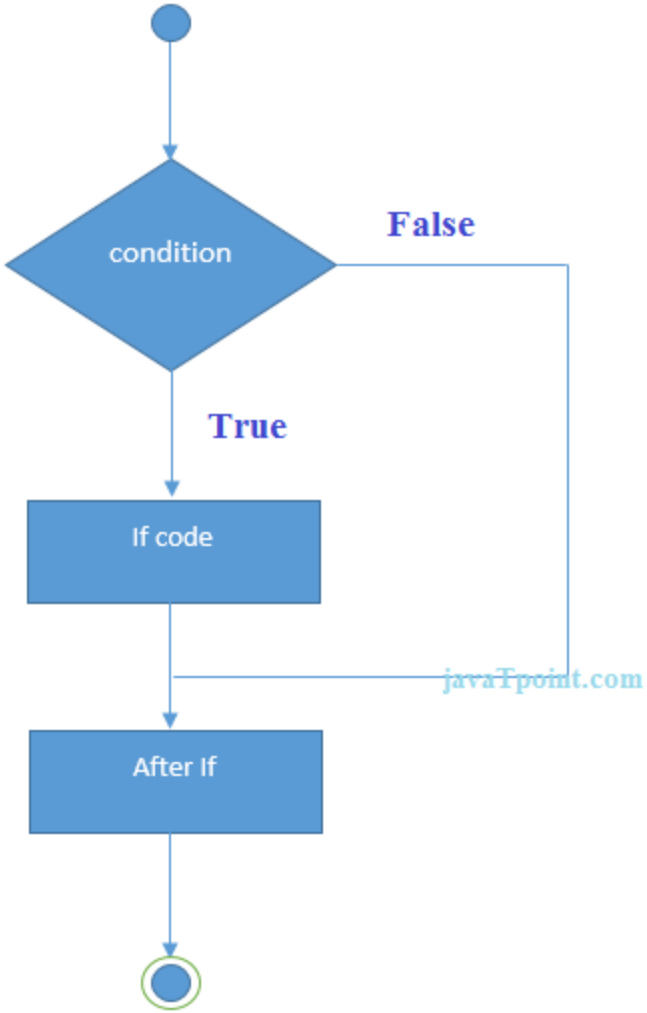
If statement

If-else statement

If else-if ladder

Nested if

Flowchart of if statement in C



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

```
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.}
```

Output

```
Enter a number:4 4 is even number
```

If-else Statement

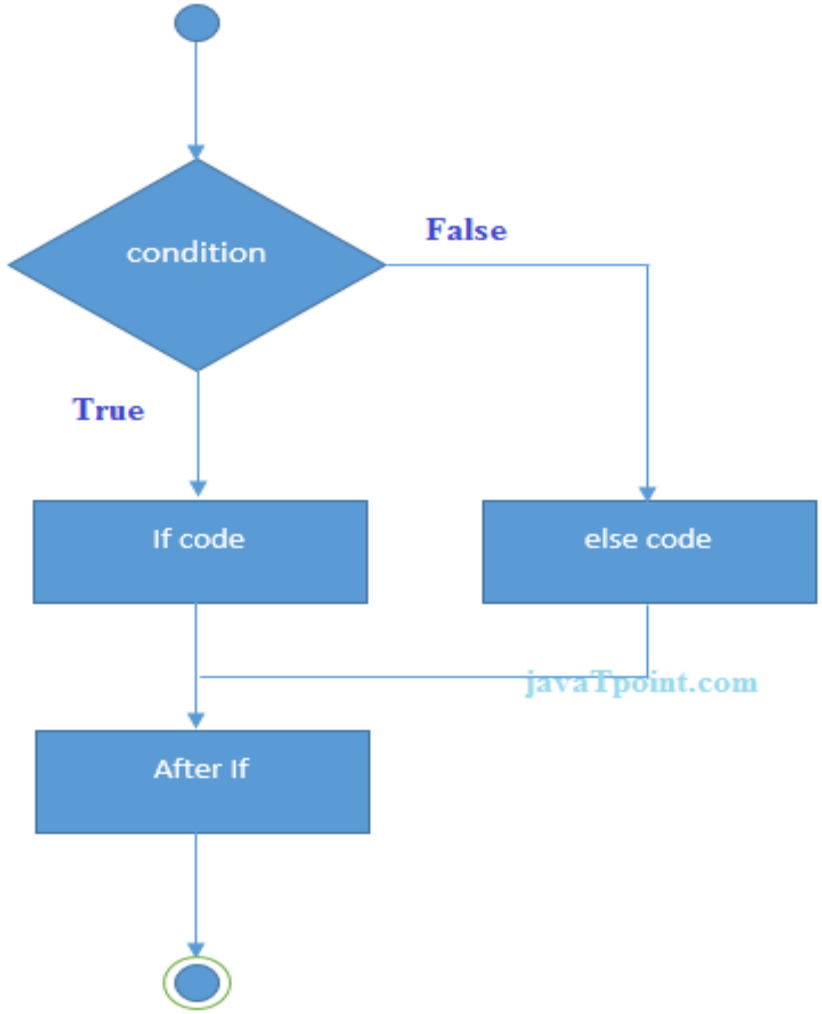
The if-else statement is used to perform two operations for a single condition. The if-else statement is an extension to the if statement using which, we can perform two different operations, i.e., one is for the correctness of that condition, and the other is for the incorrectness of the condition.

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



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

```
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.else{
10.printf("%d is odd number",number);
11.}
12.return 0;
13.}
```

Output

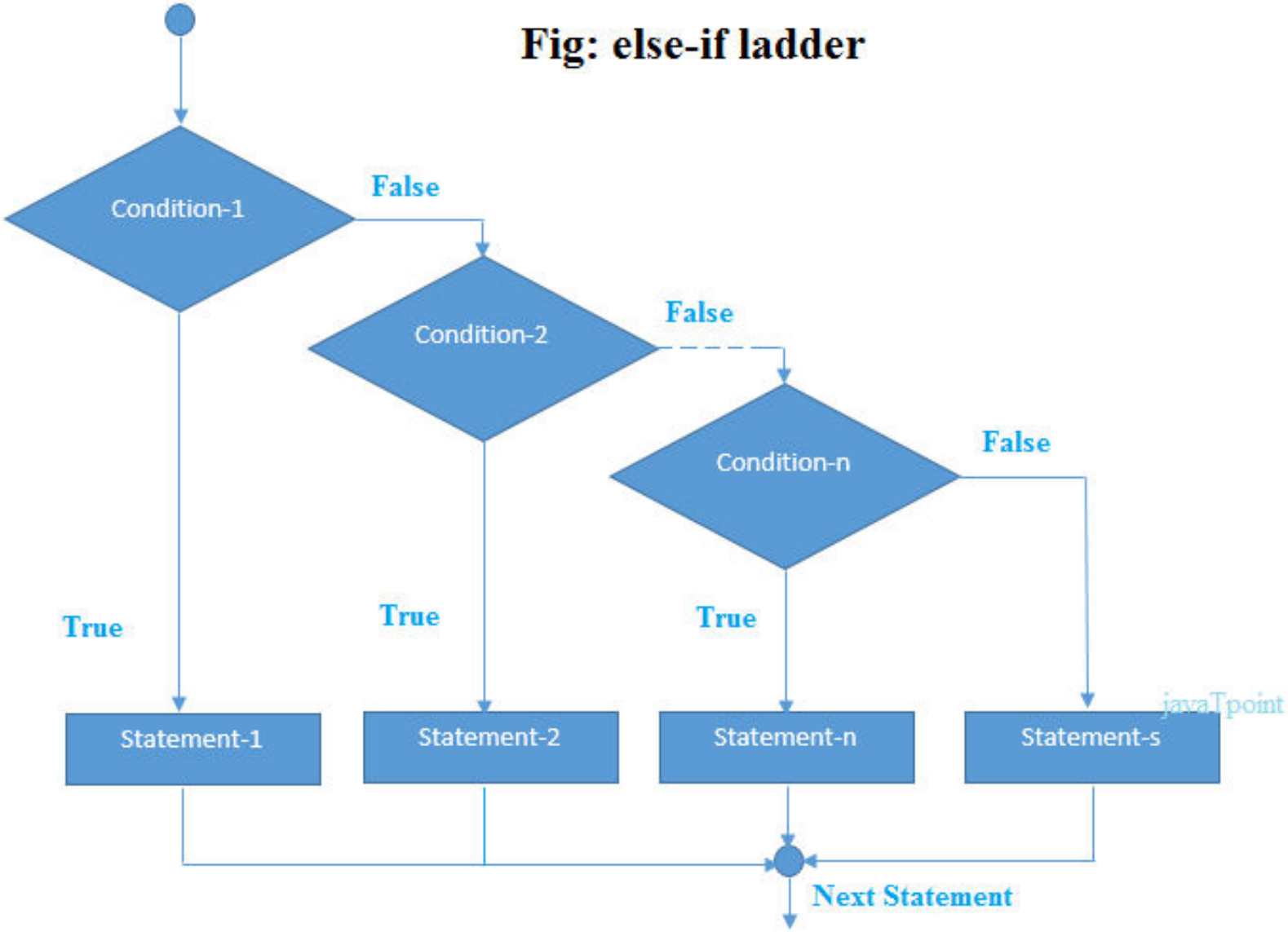
enter a number:4 4 is even number
enter a number:5 5 is odd number

If else-if ladder Statement

The if-else-if 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
}
```

Fig: else-if ladder



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

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

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 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

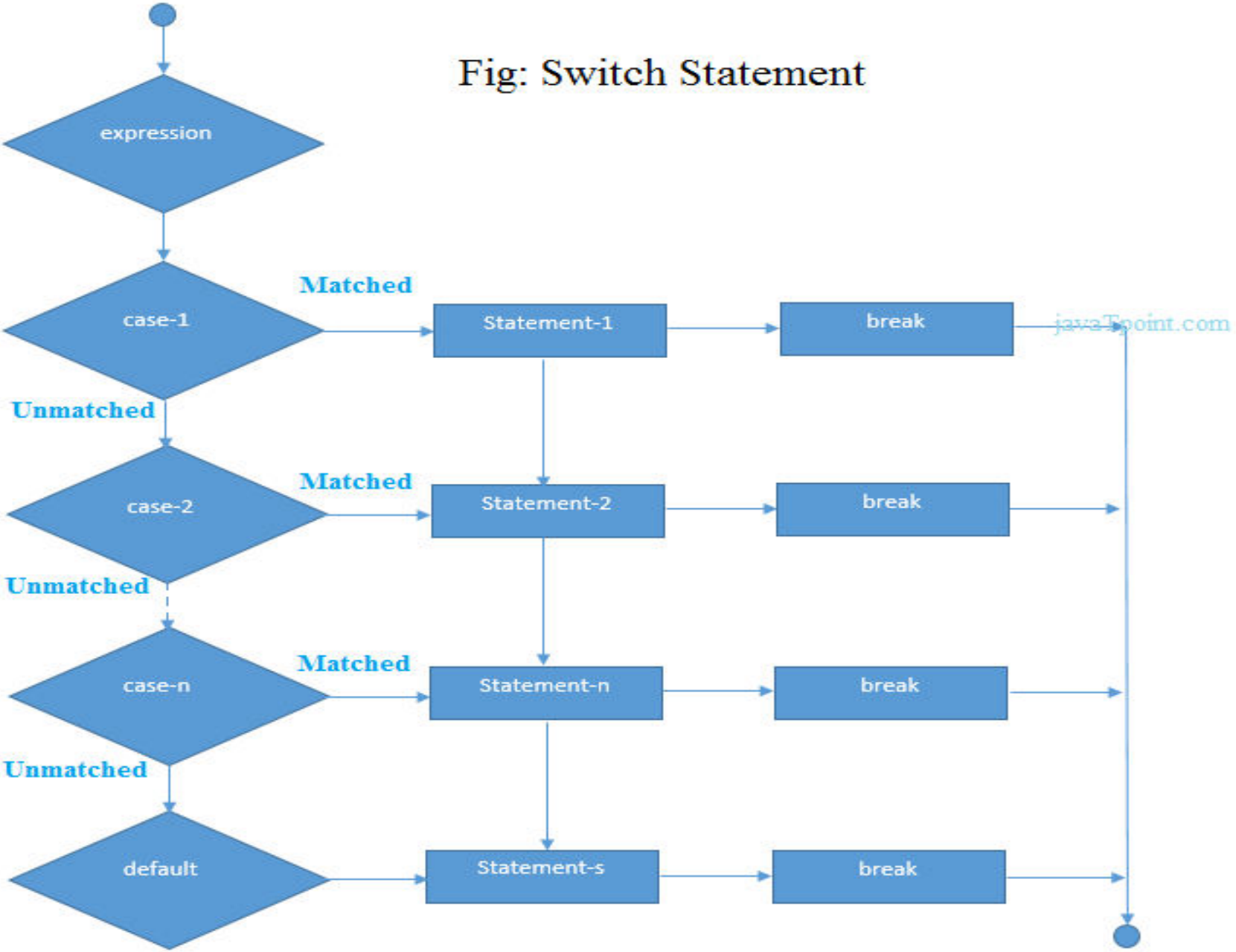
1) The *switch expression* must be of an integer or character type.

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.

Fig: Switch Statement



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

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

Output

enter a number:4 number is not equal to 10, 50 or 100

enter a number:50 number is equal to 50