Control Statements

In C programs, statements are executed sequentially in the order in which they appear in the program. But sometimes we may want to use a condition for executing only a part of program. Also many situations arise where we may want to execute some statements several times.

Control statements enable us to specify the order in which the various instructions in the program are to be executed. They define how the control is transferred to other parts of the program.

The different types of control statements in C .

Selection Statements

If….else

Switch

Iterative statements

While

for

do…… while

Jump statements

Break

Continue

goto

Compound statements

A compound statement (also called a "block") typically appears as the body of another statement, such as the if statement, for statement, while statement, etc

Prasant Dash
7008191907

A Compound statement consists of several individual statements enclosed within a pair of braces { } control statements. Expression statements, a compound statement does not end with a semicolon.

```
{
pi=3.14;
area=pi*radius*radius;
}
```

Selection Statement/Conditional Statements/Decision Making Statements

These are used to evaluate one or more conditions and make the decision whether to execute a set of statements or not. These decision-making statements in programming languages decide the direction of the flow of program execution.

C‟‟ language supports two conditional statements.

1: if
2: switch.

if Statement: The if Statement may be implemented in different forms.

1: simple if statement.

2: if –else statement

3: nested if-else statement.

4: else if ladder.

if statement.

It is used to decide whether a certain statement or block of statements will be executed or not

Prasant Dash
7008191907

The body of an if statement is executed if the value of the expression is nonzero. Or if statement is used to execute the code if condition is true. If the expression/condition is evaluated to false (0), a statement inside the body of if is skipped from execution.

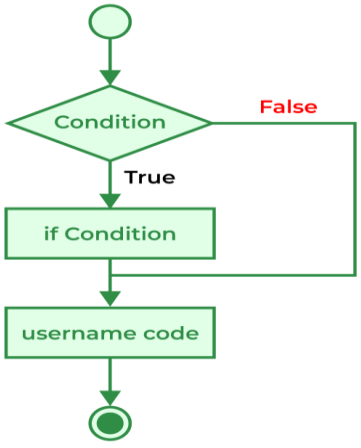
Syntax:

```
if(condition/expression)
{
true statement;
}
    statement-x;
```

If the condition/expression is true, then the true statement will be executed otherwise the true statement block will be skipped and the execution will jump to the statement-x. The „true statement" may be a single statement or group of statement.

If there is only one statement in the if block, then the braces are optional. But if there is more than one statement the braces are compulsory

Flowchart of if Statement



Prasant Dash
7008191907

Experiments #1

```
#include<stdio.h>
 void main()
 {
 int a=10,b=7;

if(a>b)
 printf("i am  true 1\n ");
 printf("\n i am  true 2");
 }
```

Output:

i am true 1

i am true 2

Hear the condition is true so print statements onwards

Experiments #2

```
#include<stdio.h>
 void main()
 {
 int a=10,b=7;

if(a<=b)
```


Prasant Dash
7008191907

```
printf("i am  true 1\n ");
printf("\n i am  true 2");
}
```

Output:

```
i am  true 2
```

Hear the condition is false, nearest statement not print, otherwise next statements will be print.

Experiments #3

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

if(-5)
 printf("i am  true 1\n ");
 printf("\n i am  true 2");
 }
```

Output:

```
i am  true 1
i am  true 2
```

Hear the condition is true so print statements onwards

Experiments #4

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


Prasant Dash
7008191907

```
 int a,b;
printf(“enter 2 number:”);
scanf(“%d,%d”,&a,&b);
if(a>=b)
{
 printf("i am  true A\n ");
 printf("\n i am  B 2");
}
}
```

Output: As per a and b values, if it is true if block will execute, if false no result

If-else statement:

The if-else statement is an extension of the simple if statement. The general form is. The if...else statement executes some code if the test expression is true (nonzero) and some other code if the test expression is false (0).

Syntax :

```
if (condition)
    {
        true statement;
    }
   else
   {
        false statement;
   }
  statement-x;
```

If the condition is true, then the true statement and statement-x will be

Prasant Dash
7008191907

executed and if the condition is false, then the false statement and statement-x is executed.

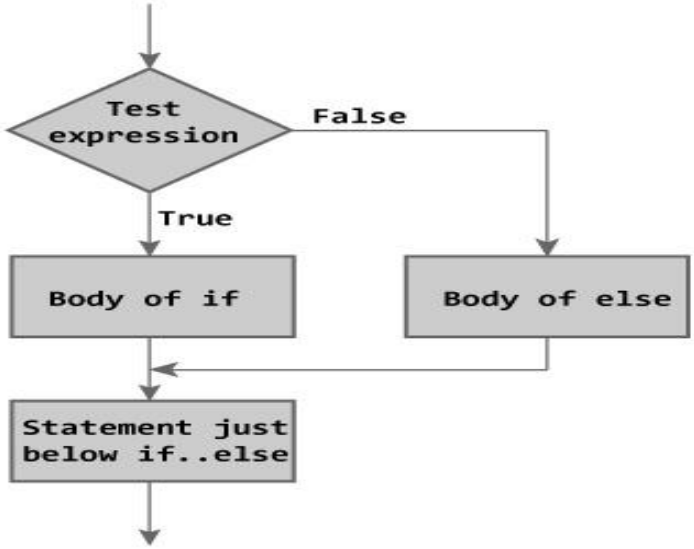


Figure: Flowchart of if...else Statement

Experiments #1 W.A.P Enter a number check even or odd

```
#include<stdio.h>
 void main()
 {
 int no;
 printf("enter Number:");
 scanf("%d",&no);
 if(no%2==0)
 printf("%d is  even number",no);
 else
  printf("%d is odd number",no);
  }
```

Output:

Prasant Dash
7008191907

Enter Number 37
37 is odd number

Experiments #2 W.A.P Enter two number find greater number

```
#include<stdio.h>
 void main()
 {
 int a,b;
 printf("enter two Number:");
 scanf("%d%d",&a,&b);
 if(a>b)
 printf("A Greater number is:%d",a );
 else
 printf(" B Greater number is:%d",b );
  }
```

Output: enter two Number 100 240
B Greater number is240

Nested if-else statement

When a series of decisions are involved, we may have to use more than on if-else statement in nested form. If –else statements can also be nested inside another if block or else block or both.

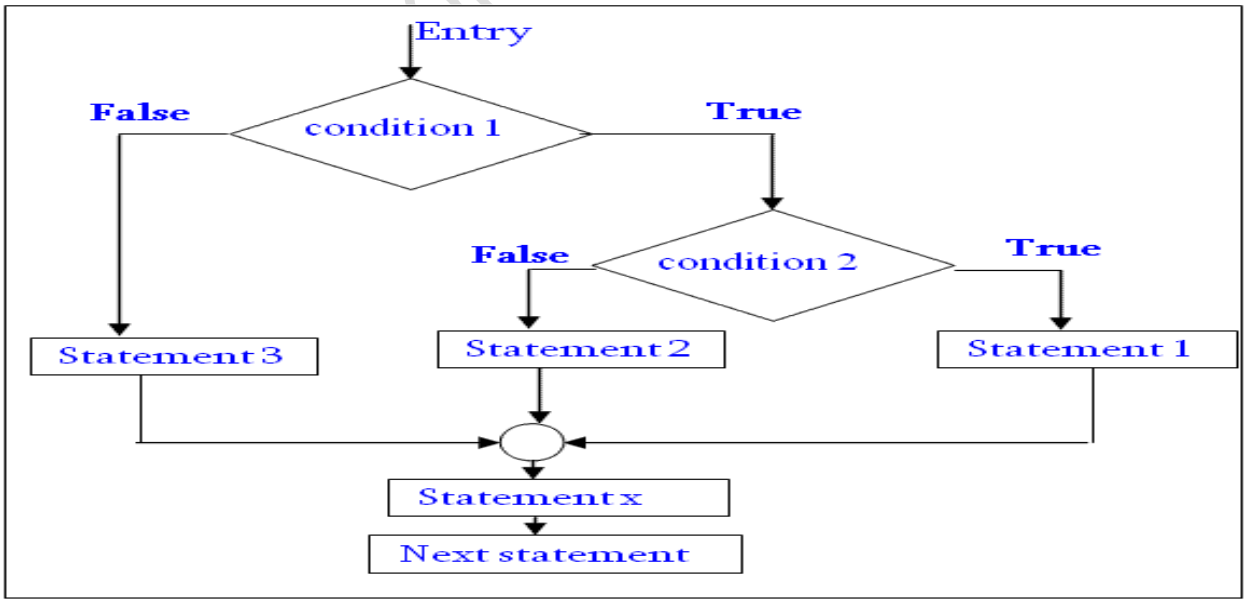
Syntax:

```
if(condition-1)
{
if (condition-2)
{
statement-1;
```


Prasant Dash
7008191907

```
}
else
{
statement-2;
}
}
else
{
        statement-3;
        }

        statement-x;
```

- ❖ if the condition 1 is true then check condition 2 if the condition 2 is true then execute statement1 and statement x.
- ❖ if the condition 1 is false execute statement 3 and statement x
- ❖ if the condition 1 is true and condtion2 is false statement 2 and statement x will be execute



Prasant Dash
7008191907

Experiment #1 W.A.P Enter three number print which is greater number

```
#include<stdio.h>
void main()
{
int a,b,c;
printf("Enter three Number:");
scanf("%d%d%d",&a,&b,&c);

if(a>b)
{
if(a>c)
{
printf("a is greater");
}
else
{
        printf("c is greater");

        }
}
                else
                {
                if(b>c)
                printf("b is greater");
                        else
                        printf("c is greater");
                        }
                }
```

Output Enter three Number: 100 200 300

C is greater

Prasant Dash
7008191907

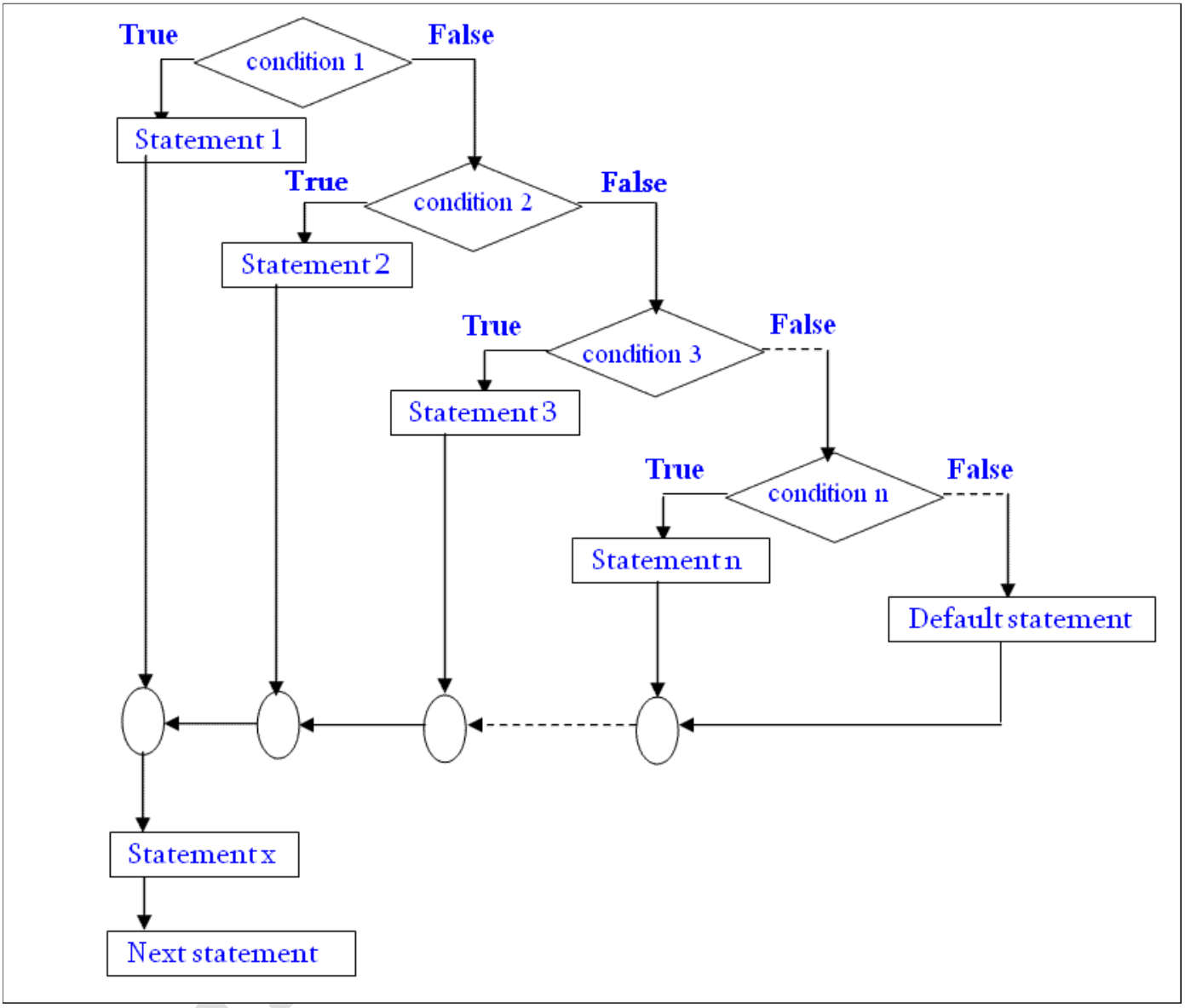
Else if ladder

The if else-if statement is used to execute one code from multiple conditions**.**

Syntax :

```
if( condition-1)
{
statement-1;
}
else if(condition-2)
{
statement-2;
}
else if(condition-3)
{
statement-3;
}
else if(condition-n)
{
statement-n;
}
else
{
default-statement;
}
        statement-x;
```


Prasant Dash
7008191907



Experiment #01

W.A.P enter current month electric units and previous units find the units for used current month ,if units 0 to 50 per units 3rupees,51-200 (4.8),201-400(5.8) above 401 per units 6.2 print current due of consumer.

Prasant Dash
7008191907


```
#include <stdio.h>

int main()
{

        int cu,pu,ru;
        float price;
        printf("Enter current units and previous units \n");
        scanf("%d%d",&cu,&pu);
        ru=cu-pu;
        printf("current units for this month is:%d\n",ru);

        if(ru>=0 && ru<=50)
                price=ru*3;
        else if(ru>=51 && ru<=200)
                price=50*3+(ru-50)*4.80;
else if(ru>=201 && ru<=400)
                price=50*3+150*4.80+(ru-200)*5.80;
else
price=50*3+150*4.80+200*5.8+(ru-400)*6.2;


printf("your current energy bill is %.2f \n",price);

        return 0;
}
```


Prasant Dash
7008191907

Points to Remember

1. In *if* statement, a single statement can be included without enclosing it into curly braces { }
2. int a = 5;
3. if(a > 4)
4. printf("success");

No curly braces are required in the above case, but if we have more than one statement inside *if* condition, then we must enclose them inside curly braces.

5. == must be used for comparison in the expression of *if* condition, if you use = the expression will always return true, because it performs assignment not comparison.

6. Other than **0(zero)**, all other values are considered as true.

7. if(27)

8. printf("hello");

In above example, hello will be printed.

Switch statement:

When there are several options and we have to choose only one option from the available ones, we can use switch statement. Depending on the selected option, a particular task can be performed. A task represents one or more statements.

Prasant Dash
7008191907

Syntax

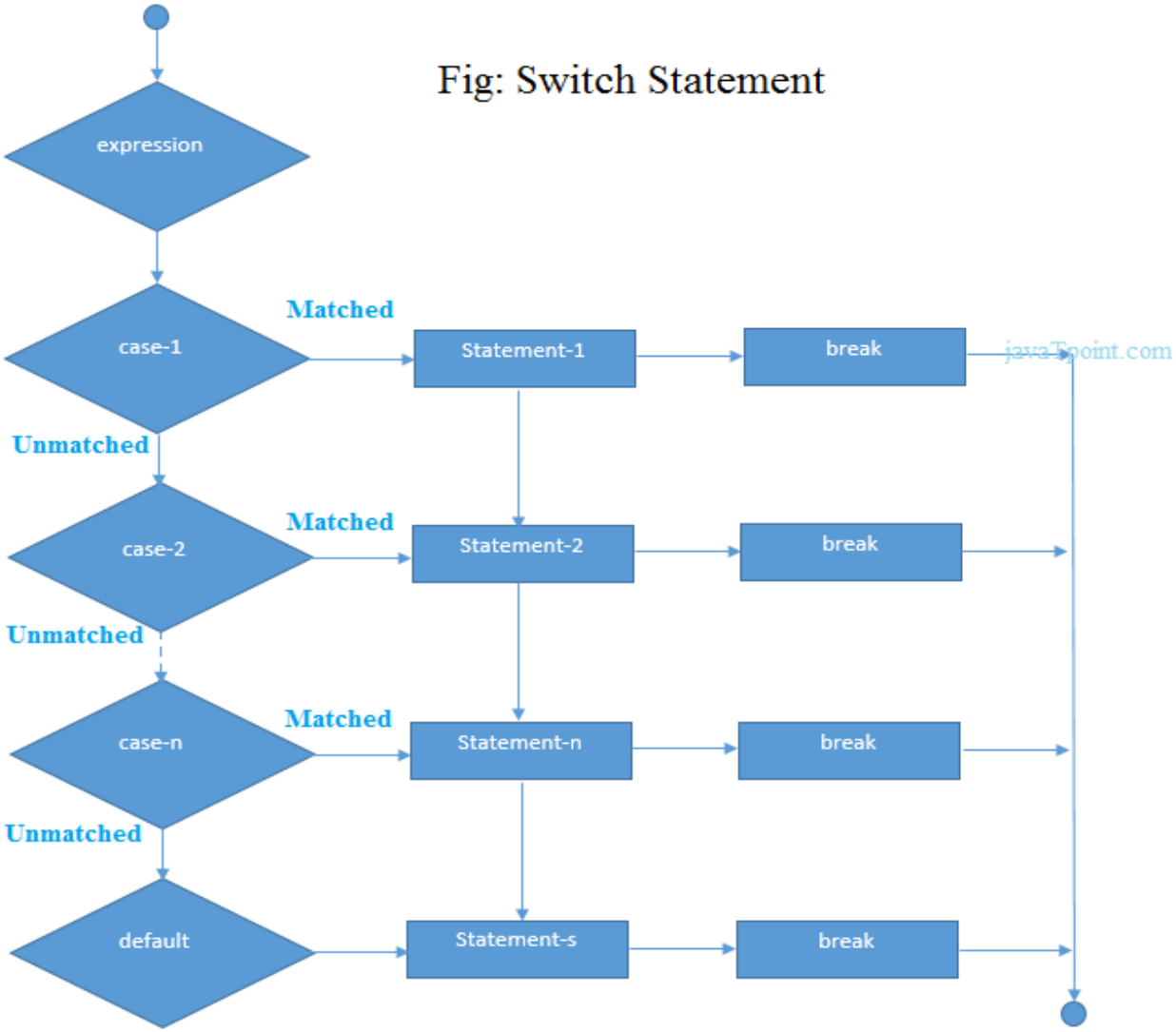
```
switch(expression)
{
case value1:
statement_1;
        break;
case value2:
 statement_2;
        break;
.
.

case value_n:
statement_n;
         break;

default:
default_statement;
}
```

Rules of the switch case statement

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 and it is unique
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.
5. The case keyword must terminate with colon (:).
6. No real numbers are used in an expression.

Prasant Dash
7008191907

Fig: Switch Statement



Experiment #01

```
#include<stdio.h>
void main()
{
int no;
printf("enter number");
scanf("%d",&no);
```


Prasant Dash
7008191907

```
switch(no)
{
case 1:
printf("i am one \n");

case 2:
printf("i am Two \n");

case 3:
printf("i am three \n");

default:
printf("i am default  \n");
}
}
```

When enter no is match onwards is print

Experiments #02

```
#include<stdio.h>
void main()
{
int no;
printf("enter number");
scanf("%d",&no);
switch(no)
{

default:
printf("i am default  \n");
```


Prasant Dash
7008191907

```
case 1:
printf("i am one \n");

case 2:
printf("i am Two \n");

case 3:
printf("i am three \n");

}
}
```

If out of case number then print default onwards or if case number then print matches number onwards

Experiments #03

```
// C program to print the day using switch
#include <stdio.h>

// Driver Code
int main()
{
        int day ;
        printf("Enter day \n");
   scanf("%d",&day);
   printf("The day with number %d is ", day);
   switch (day) {
     case 1:
       printf("Monday");
```


Prasant Dash
7008191907

```
    break;
case 2:
    printf("Tuesday");
    break;
case 3:
    printf("Wednesday");
    break;
case 4:
    printf("Thursday");
    break;
case 5:
    printf("Thursday");
    break;
case 6:
      printf("Thursday");
    break;
case 7:
    printf("Thursday");
    break;
default:
    printf("Invalid Input");
    break;
  }
return 0;
}
```


Prasant Dash
7008191907

Experiments #04

```
#include <stdio.h>

// Driver Code
int main()
{
char oper;
int result,a,b;
printf("enter 2 number:");
scanf("%d%d",&a,&b);
        printf("Enter operater \n");
        fflush(stdin);
        scanf("%c",&oper);
        switch (oper)
        {
                case '+':
                result=a+b;
                        printf("add=%d",result);
                        break;
                case '-':
                                result=a-b;
                        printf("sub=%d",result);
                        break;

                case '*':
                                result=a*b;
                        printf("mul=%d",result);
                        break;
                case '/':
```


Prasant Dash
7008191907

```
                result=a/b;
                printf("div=%d",result);
                break;

        case '%':
                result=a%b;
                printf("mod=%d",result);
                break;

        default:
                printf("Invalid operator");
                break;
    }
        return 0;
}
```

Output

```
enter 2 number:24 3
Enter operater
+
add=27
```


Prasant Dash
7008191907