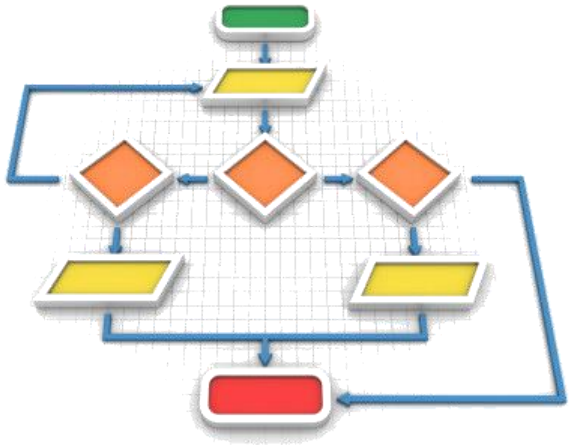
Decision Making, Branching & Switch

L8



Learning objectives

To learn and appreciate the following concepts

- The if Statement
- The if-else Statement

Learning Outcomes

- At the end of session student will be able to learn and understand
  - The if Statement
  - The if-else Statement

Control Structures

- **A control structure refers to the order of executing the program statements.**
- The following three approaches can be chosen depending on the problem statement:

✓**Sequential (Serial)**

- In a **Sequential approach**, all the statements are executed in the same order as it is written.

✓**Selectional (Decision Making and Branching)**

- In a **Selectional approach**, based on some conditions, different set of statements are executed.

✓**Iterational (Repetition)**

- In an **Iterational approach** certain statements are executed repeatedly.

DECISION MAKING AND BRANCHING

C decision making and branching statements are:

1. if statement
2. switch statement

Different forms of **if** statement

1. Simple **if** statement.

2. **if...else** statement.

3. Nested **if...else** statement.

4. **else if** ladder.

Simple if Statement

General form of the simplest if statement:

```
if (test Expression)
   {
    statement-block;
   }
    next_statement;
```

If expression is true (non-zero), executes statement.
It gives you the choice of executing statement or skipping it.

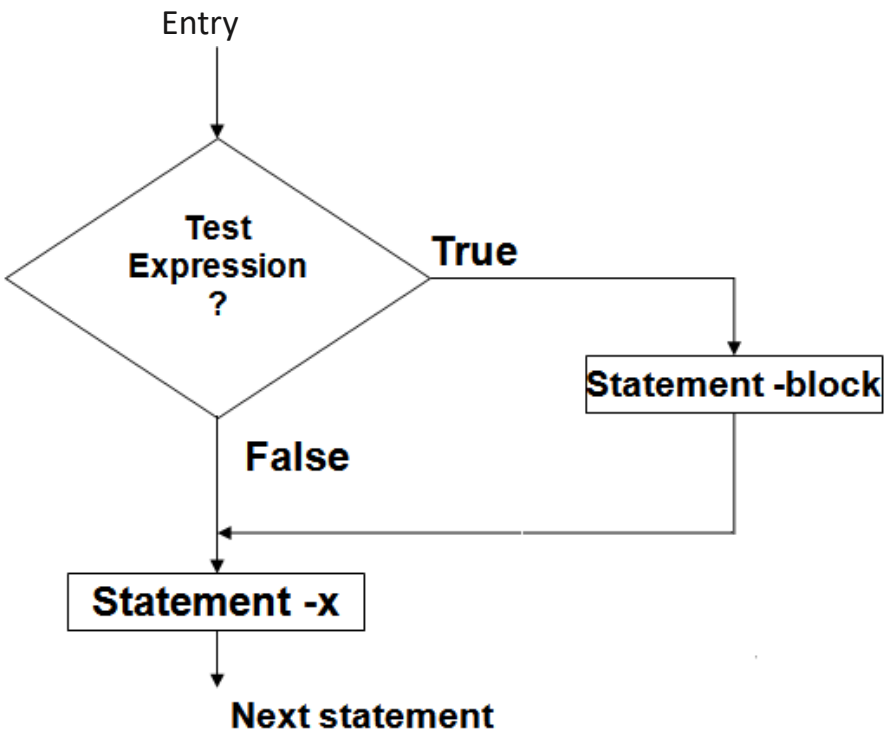
if Statement- explanation

- (*test Expression*) is first evaluated.
- If TRUE (non-zero), the 'if' statement block is executed.
- If FALSE (zero) the next statement following the if statement block is executed.
- So, during the execution, based on some condition, some code will not be executed (skipped).

For example:

```
bonus = 0;
if (hours > 70)
bonus = 10000;
salary= salary + bonus;
```

Flow chart of simple if



Find out whether a number is even or odd.

```
#include <stdio.h>
int main()
{
  int  x;
  printf("input an integer\n");
  scanf("%d",&x);
  if ((x % 2) == 0)
  {
        printf("It is an even number\n");
  }
  if ((x%2) == 1)
  {
        printf("It is an odd number\n");
  }
   return 0;
}
```

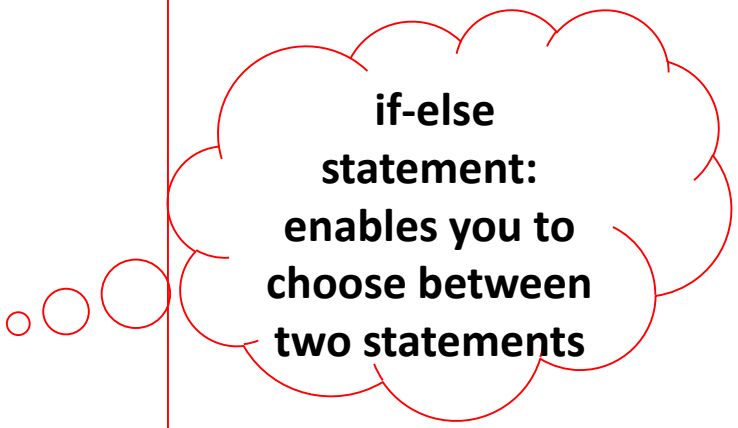
Example - `if`

```
// Program to calculate the absolute value of an integer

int main ()
{
      int number;
      printf("Type in your number: ");
      scanf("%d",&number);
      if ( number < 0 )
            number = -number;
      printf("The absolute value is");
      printf("%d",number);
      return 0;
}
```

The `if-else` statement

```
if (test expression )
{
    statement _block1
}
else
{
    statement _block2
}
Next_statement
```



if-else statement

Explanation:

1.First ,the (test expression) is evaluated.

2.If it evaluates to non-zero (TRUE), statement_1 is executed, otherwise, if it evaluates to zero (FALSE), statement_2 is executed.

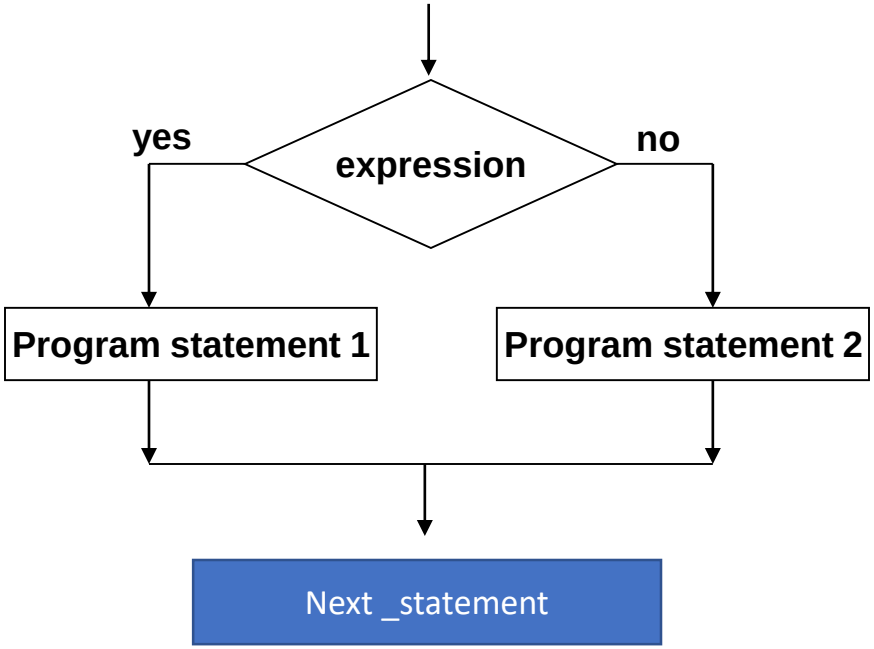
3.They are mutually exclusive, meaning, either statement_1 is executed or statement_2, but not both.

4.If the statements_ 1 and statements_ 2 take the form of block , they must be put in curly braces.

Example:

```
if(job_code ==  1)
        rate = 7.00;
else
        rate = 10.00;
prinf("%d",rate);
```

The `if-else` statement



Find out whether a number is even or odd.

```
#include <stdio.h>
int main()
{
  int  x;
  printf("Input an integer\n");
  scanf("%d",&x);
  if ((x % 2) == 0)
  {
        printf("It is an even number\n");
  }
  else
  {
        printf("It is an odd number\n");}
   return 0;
  }
```

WAP to find largest of 2 numbers

```
#include<stdio.h>
int main()
{
        int a, b;
        printf("Enter 2 numbers\n");
        scanf("%d %d",&a,&b);
         if(a > b)
                        printf("Large is %d\t",a);
         else
                        printf("Large is %d\t",b);

        return 0;
}
```

Attention on `if-else` syntax !

```
if ( expression )
        program
statement 1
else
        program
statement 2
```

In C, the ; is part (end) of a statement !

```
if ( remainder == 0 )
        printf("The number is even.\n");
else
        printf("The number is odd.\n");
```

Syntactically OK (void statement on if) but a semantic error !

```
if ( x == 0 );
        printf("The number is zero.\n");
```

Example: determine if a year is a leap year

```
#include<stdio.h>
int main()
{
   int year;
   printf(“Enter the year“);
   scanf(“%d”,&year);
   if(year%4 == 0)
   {
      if( year%100 == 0)
      {
            if ( year%400 == 0)
            printf("%d is a leap year",year);
         else
            printf("%d is not a leap year",year);
     }  else printf("%d is a leap year",year);
   } else    printf("%d is not a leap year",year);
   return 0;
}
```

A leap year is exactly divisible by 4 except for century years (years ending with 00). The century year is a leap year only if it is perfectly divisible by 400.

Testing for character ranges

```
#include<stdio.h>
int main()
{
   char ch;
   printf(“enter a character\n”);
    scanf(“%c”,&ch);
   if (ch >= 'a' && ch <= 'z')
                printf(“lowercase char\n“);
   if (ch >= ‘A' && ch <= ‘Z')
                printf(“uppercase char\n“);
   if (ch >= ‘0' && ch <= ‘9')
                printf(“digit char\n“);
   else
                printf(“ special char\n”);
return 0;
}
```

Output:

```
enter a
character:
 C
uppercase char
special char

enter a
character:
 j
lowercase char
special char

enter a
character:
 5
digit char
```

Testing for ranges

```
if (x >= 5 && x <= 10)
        printf("in range");
```

```
if (5 <= x <= 10)
        printf("in range");
```

Testing for ranges

YES

```
if (x >= 5 && x <= 10)
                printf("in range");
```

NO !

```
if (5 <= x <= 10)
                printf("in range");
```

Syntactically correct, but semantically an error !!!

Because the order of evaluation for the <= operator is left-to-right, the test expression is interpreted as follows:

`(5<= x) <= 10`

The subexpression `5 <= x` either has the value 1 (for true) or 0 (for false). Either value is less than 10, so the whole expression is always true, regardless of `x` !

Poll Question

Go to chat box/posts for the link to the Poll question

Submit your solution in next 2 minutes

Click the result button to view your score