

## IT 112: Introduction to Programming

Dr. Manish Khare


Dr. Bakul Gohel



Lecture 8,9

# Control Instructions in C

- As the name suggests the ‘Control Instructions’ enable us to specify the order in which the various instructions in a program are to be executed by the computer.
- In other words the control instructions determine the ‘flow of control’ in a program.
- There are four types of control instructions in C.
  - Sequence Control Instruction
  - Selection or Decision Control Instruction
  - Repetition or Loop Control Instruction
  - Case Control Instruction

- 
- The Sequence control instruction ensures that the instructions are executed in the same order in which they appear in the program.
  - Decision and Case control instructions allow the computer to take a decision as to which instruction is to be executed next.
  - The Loop control instruction helps computer to execute a group of statements repeatedly

# Decisions! Decisions

- By default the instructions in a program are executed sequentially.
- In serious programming situations, seldom do we want the instructions to be executed sequentially.
- We want a set of instructions to be executed in one situation, and an entirely different set of instructions to be executed in another situation

# Decision Control Instructions

➤ **Decision control instruction** can be implemented in C using

- **if** statement
- **if-else** statement
- The conditional operators

➤ C uses the keyword **if** to implement the decision control instruction. The general form of **if** statement:

- *if ( this condition is true )*
  - *execute this statement ;*

# Decision Control Instructions

## ➤ Relational Operators

- Condition can be specified using C's 'relational' operators.
- Relational operators allow us to compare two values.

this expression	is true if
$x == y$	x is equal to y
$x != y$	x is not equal to y
$x < y$	x is less than y
$x > y$	x is greater than y
$x \leq y$	x is less than or equal to y
$x \geq y$	x is greater than or equal to y

# Relational Operators

*/\* Demonstration of if statement \*/*

```
main( )
```

```
{
```

```
    int num ;
```

```
    printf ( "Enter a number less than 10 " ) ;
```

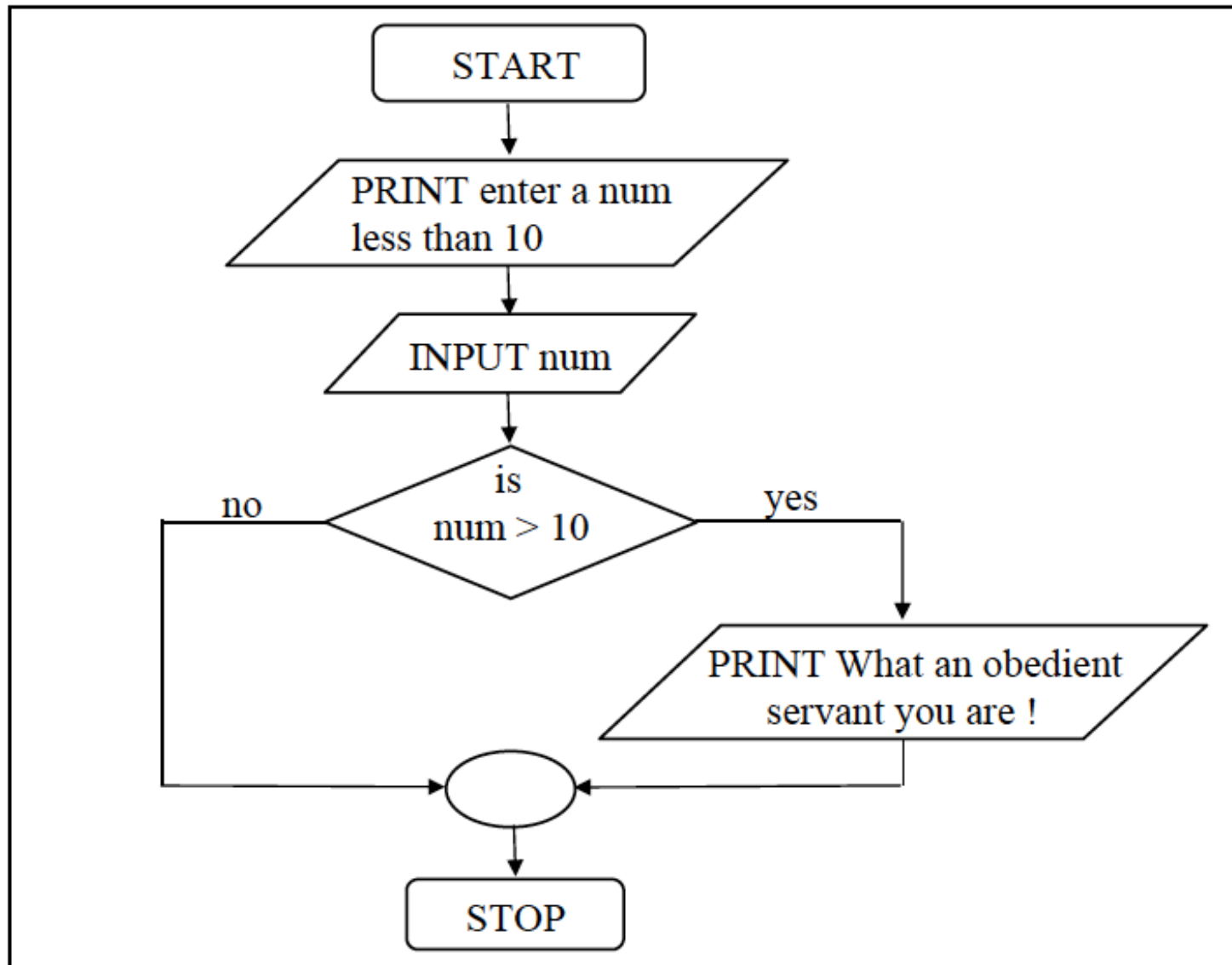
```
    scanf ( "%d", &num ) ;
```

```
    if ( num <= 10 )
```

```
        printf ( "What an obedient servant you are !" ) ;
```

```
}
```

# "If" Statement





## Example (IF)

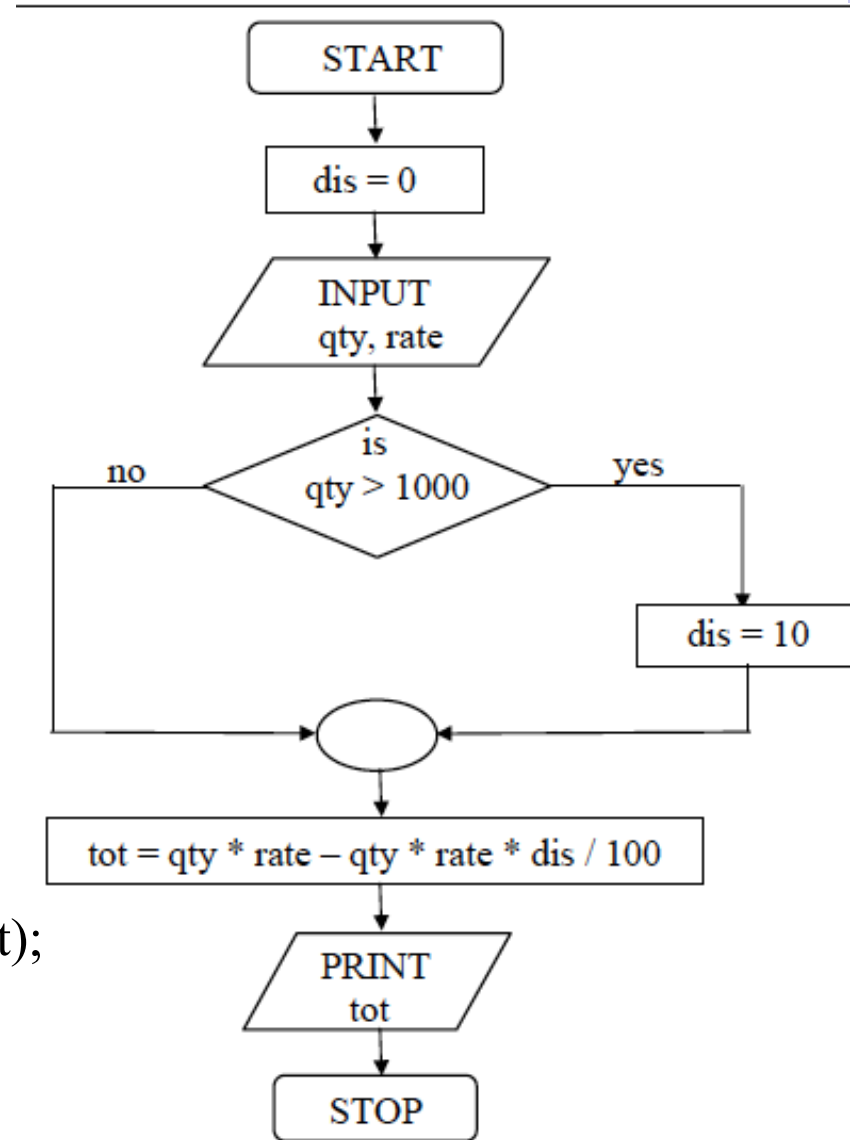
- While purchasing certain items, a discount of 10% is offered if the quantity purchased is more than 1000. If quantity and price per item are input through the keyboard, write a program to calculate the total expenses.

# Flowchart - Example

```
/* Calculation of total expenses */
main()
{
    int qty, dis = 0 ;
    float rate, tot ;

    printf ("Enter quantity and rate ") ;
    scanf ("%d %f", &qty, &rate) ;

    if ( qty > 1000 )
        dis = 10 ;
    tot=(qty*rate)-(qty*rate*dis/100) ;
    printf ("Total expenses = Rs. %f", tot);
}
```



# The Real Thing with “IF”



```
if (condition)  
    statement;
```

The expression can be any valid expression including a relational expression.

Can even use arithmetic expressions in the **if statement**.

```
if (expression)  
    statement;
```

# The Real Thing with "IF"

- `if ( 3 + 2 % 5 )`  
    `printf ( "This works" ) ;`
- `if ( a = 10 )`  
    `printf ( "Even this works" ) ;`
- `if ( -5 )`  
    `printf ( "Surprisingly even this works" ) ;`
- `if ( a==b==c )`  
    Result of `a==b` is compared with `c`

Note that in C a non-zero value is considered to be true, whereas a 0 is considered to be false.

# Multiple Statements within “IF”

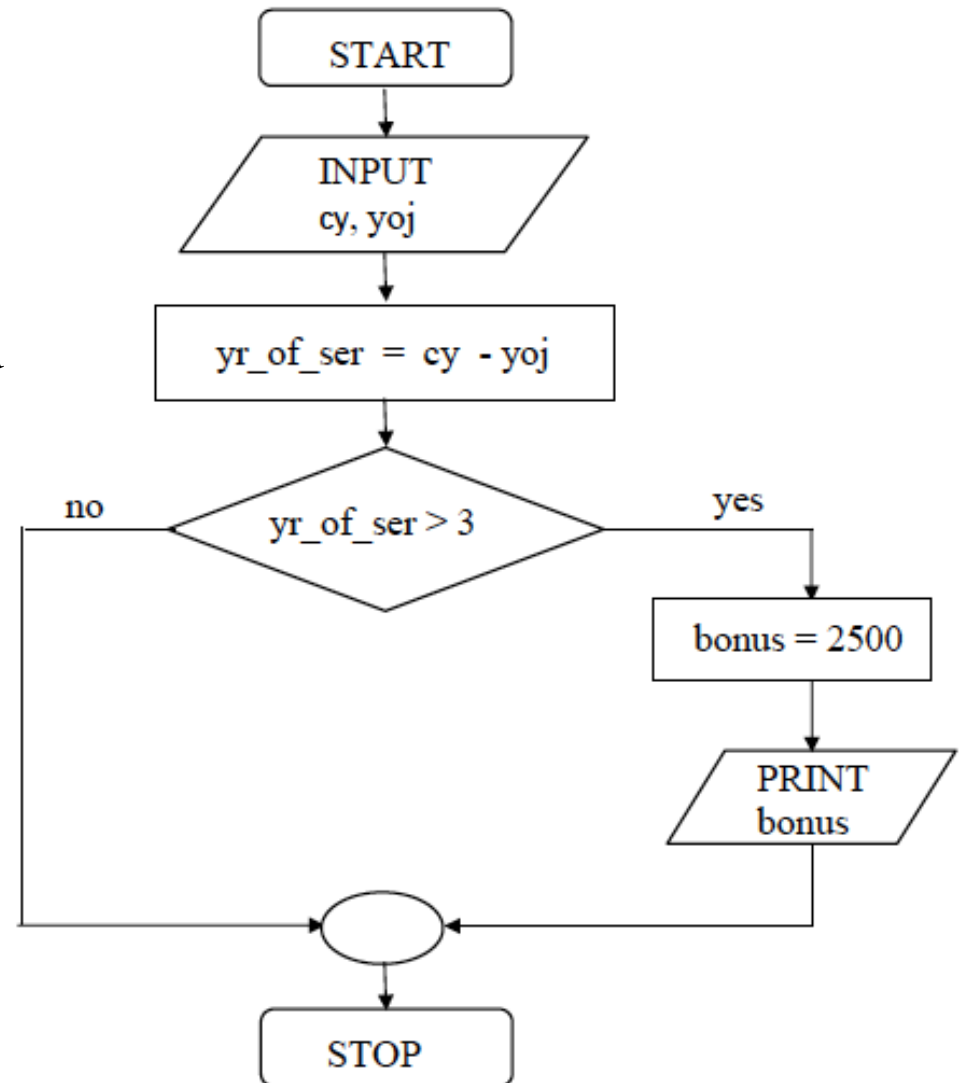
- In a “IF” statement multiple statements are to be executed then
  - They must be placed within a pair of braces.

## Example:

The current year and the year in which the employee joined the organization are entered through the keyboard. If the number of years for which the employee has served the organization is greater than 3 then a bonus of Rs. 2500/- is given to the employee. If the years of service are not greater than 3, then the program should do nothing.

# Multiple Statements within "IF" - Example

```
/* Calculation of bonus */  
main( )  
{  
    int bonus, cy, yoj, yr_of_ser ;  
    printf ("Enter current year and  
year of joining") ;  
    scanf ("%d %d", &cy, &yoj);  
    yr_of_ser = cy - yoj ;  
    if (yr_of_ser > 3)  
    {  
        bonus = 2500 ;  
        printf ("Bonus = Rs.  
%d", bonus);  
    }  
}
```



# The if-else Statement

- The **if statement** by itself will execute a single statement, or a group of statements, when the expression following if evaluates to true.
- It does nothing when the expression evaluates to false.
- Can we execute one group of statements if the expression evaluates to true and another group of statements if the expression evaluates to false?
- Of course! This is what is the purpose of the **else statement**

## Example (IF-ELSE)

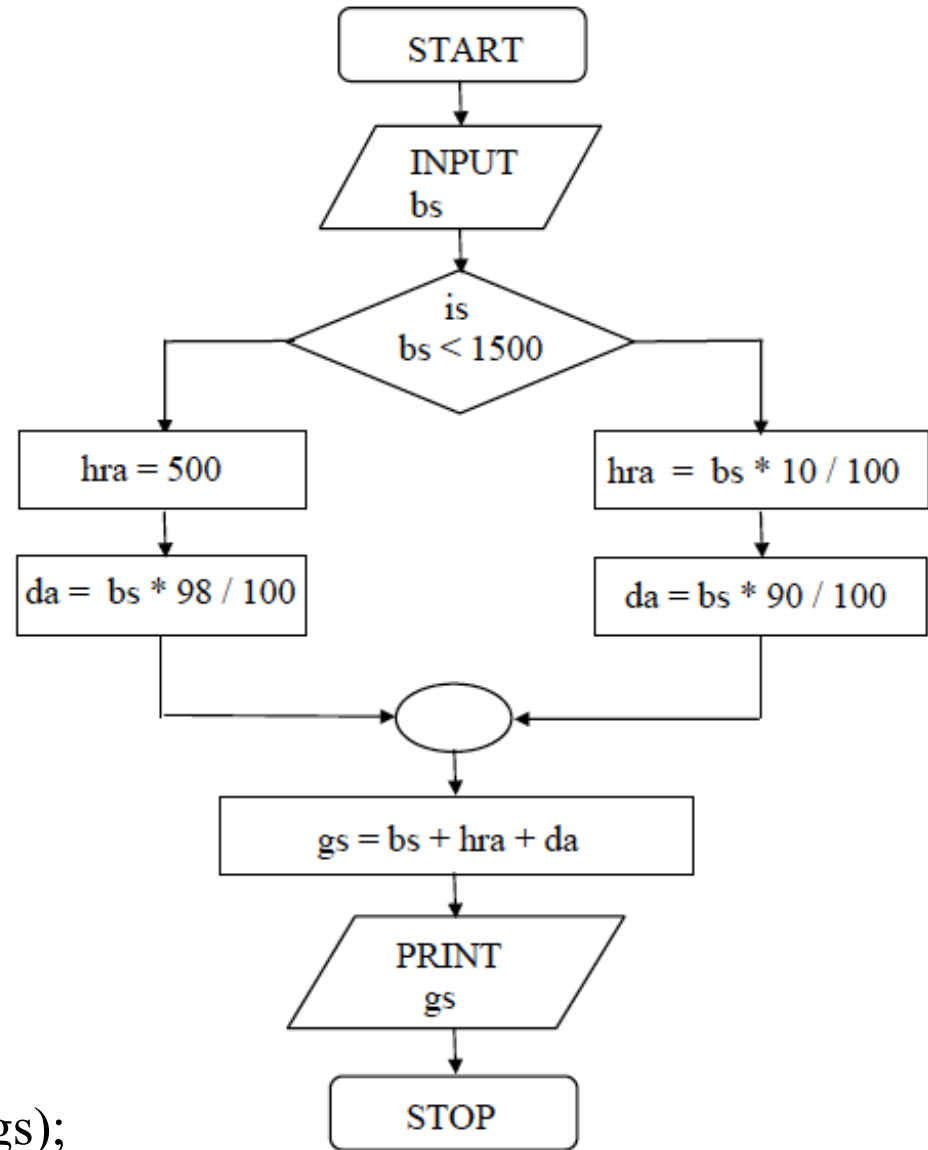
In a company an employee is paid as under:

If his basic salary is less than Rs. 1500, then HRA = 10% of basic salary and DA = 90% of basic salary. If his salary is either equal to or above Rs. 1500, then HRA = Rs. 500 and DA = 98% of basic salary. If the employee's salary is input through the keyboard write a program to find his gross salary.



# "IF-ELSE" - Example

```
/* Calculation of gross salary */  
main()  
{  
    float bs, gs, da, hra ;  
    printf ( "Enter basic salary " ) ;  
    scanf ( "%f", &bs ) ;  
    if ( bs < 1500 )  
    {  
        hra = bs * 10 / 100 ;  
        da = bs * 90 / 100 ;  
    }  
    else  
    {  
        hra = 500 ;  
        da = bs * 98 / 100 ;  
    }  
    gs = bs + hra + da ;  
    printf ("gross salary=Rs. %f", gs);  
}
```



# Points to Remember

- The group of statements after the **if** upto and not including the **else** is called an 'if block'. Similarly, the statements after the **else** form the 'else block'.
- Notice that the **else** is written exactly below the **if**. The statements in the if block and those in the else block have been indented to the right.
- Had there been only one statement to be executed in the if block and only one statement in the else block we could have dropped the pair of braces.
- As with the **if** statement, the default scope of **else** is also the statement immediately after the **else**. To override this default scope a pair of braces as shown in the above example must be used.

# Nested IF-ELSESES

- Write an entire **if-else** construct within either the body of the **if** statement or the body of an **else** statement.
  - This is called 'nesting' of ifs

# Nested IF-ELSESES (Example)

```
/* A quick demo of nested if-else */
main( )
{
    int i ;
    printf ( "Enter either 1 or 2 " ) ;
    scanf ( "%d", &i ) ;
    if ( i == 1 )
        printf ( "You would go to heaven !" ) ;
    else
    {
        if ( i == 2 )
            printf ( "Hell was created with you in mind" ) ;
        else
            printf ( "How about mother earth !" ) ;
    }
}
```

# Forms of *IF*

```
if ( condition )  
    do this ;
```

```
if ( condition )  
{  
    do this ;  
    and this ;  
}
```

```
if ( condition )  
    do this ;  
else  
    do this ;
```

```
if ( condition )  
{  
    do this ;  
    and this ;  
}  
else  
{  
    do this ;  
    and this ;  
}
```

# Forms of *IF*

```
if ( condition )
    do this ;
else
{
    if ( condition )
        do this ;
    else
    {
        do this ;
        and this ;
    }
}
```

```
if ( condition )
{
    if ( condition )
        do this ;
    else
    {
        do this ;
        and this ;
    }
}
else
    do this ;
```

# Word of Caution

Guess output of the program?

```
#include <stdio.h>
main()
{
    int i;
    printf("Enter Value of i");
    scanf("%d", &i);

    if (i=5)
        printf("You have entered 5\n");
    else
        printf("You have entered something other than 5\n");
}
```

# Word of Caution

Enter value of i 200

o/p: You have entered 5

Enter value of i 5

o/p: You have entered 5

Enter value of i 9999

o/p: You have entered 5

*Surprised?*



# Word of Caution: Another

Guess output of the program?

```
#include <stdio.h>
main()
{
    int i;
    printf("Enter Value of i");
    scanf("%d", &i);

    if (i==5);
        printf("You have entered 5\n");
}
```

*You have entered 5*

# Word of Caution: Another

```
#include <stdio.h>
main()
{
    int i;
    printf("Enter Value of i");
    scanf("%d", &i);

    if (i==5)
        ;
        printf("You have entered 5\n");
}
```

## *Note:*

- if the condition evaluates to true, the ; (null statement, which does nothing in execution gets executed.
- If the condition fails, printf() gets executed.
- In any case, printf() gets executed – fail or pass



## ➤ More complex Decision Making

- If a get good marks in my final year and if my GRE and TOFEL scores are good and if I get good recommendations or of I do not get a job with good prospects and if my family conditions permit me, then I would think of doing MS in US.
- How can such complex decision making be implemented in C?

# Use of Logical Operators

➤ C allows usage of three logical operators, namely, `&&`, `||` and `!`.

■ These are to be read as:

- 'AND'                      `&&`
- 'OR'                        `||`
- 'NOT'                      `!`

➤ **Remember:** Don't use the single symbol `|` and `&`. These have a different meaning. They are bitwise operators.

# Use of operators && and | |

➤ Example:

The marks obtained by a student in 5 different subjects are input through the keyboard. The student gets a division as per the following rules:

- Percentage above or equal to 60 - First division
- Percentage between 50 and 59 - Second division
- Percentage between 40 and 49 - Third division
- Percentage less than 40 - Fail

Write a program to calculate the division obtained by the student.

# Solution Method 1

*/\* Method – I \*/*

main( )

{

int m1, m2, m3, m4, m5, per ;

printf ( "Enter marks in five subjects " ) ;

scanf ( "%d %d %d %d %d", &m1, &m2, &m3, &m4,  
&m5 ) ;

per = ( m1 + m2 + m3 + m4 + m5 ) / 5 ;

# Solution Method 1

```
if ( per >= 60 )
    printf ( "First division " );
else
{
    if ( per >= 50 )
        printf ( "Second division" );
    else
    {
        if ( per >= 40 )
            printf ( "Third division" );
        else
            printf ( "Fail" );
    }
}
} /* main */
```

# Use of 'Logical operators'

*/\* Method – II \*/*

main( )

{

int m1, m2, m3, m4, m5, per ;

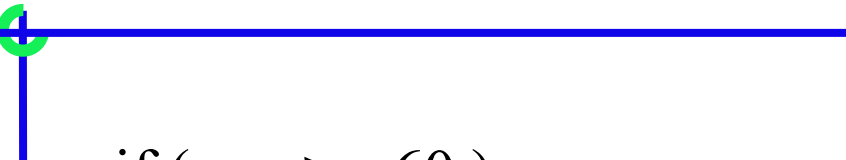
printf ( "Enter marks in five subjects " ) ;

scanf ( "%d %d %d %d %d", &m1, &m2, &m3, &m4,  
&m5 ) ;

per = ( m1 + m2 + m3 + m4 + m5 ) / 5 ;



## Solution Method 2



```
if ( per >= 60 )  
    printf ( "First division" ) ;  
  
if ( ( per >= 50 ) && ( per < 60 ) )  
    printf ( "Second division" ) ;  
  
if ( ( per >= 40 ) && ( per < 50 ) )  
    printf ( "Third division" ) ;  
  
if ( per < 40 )  
    printf ( "Fail" ) ;  
} /* main */
```

# The *else if* clause

```
if ( per >= 60 )
    printf ( "First division " );
else
{
    if ( per >= 50 )
        printf ( "Second division" );
    else
    {
        if ( per >= 40 )
            printf("Third division");
        else
            printf ( "Fail" );
    }
}
```

```
/* else if ladder demo */
main( )
{
    int m1, m2, m3, m4, m5, per ;
    per = ( m1+ m2 + m3 + m4+ m5 ) /5;
    if ( per >= 60 )
        printf ( "First division" );
    else if ( per >= 50 )
        printf ("Second division");
    else if ( per >= 40 )
        printf ( "Third division" );
    else
        printf ( "fail" );
}
```

# The *else if* clause

```
if ( i == 2 )
    printf ( "With you..." );
else
{
    if ( j == 2 )
        printf ( "...All the time" );
}
```

```
if ( i == 2 )
    printf ( "With you..." );
else if ( j == 2 )
    printf ( "...All the time " );
```

# Example – Logical Operators

Example:

A company insures its drivers in the following cases:

- If the driver is married.
- If the driver is unmarried, male & above 30 years of age.
- If the driver is unmarried, female & above 25 years of age.

In all other cases the driver is not insured. If the marital status, sex and age of the driver are the inputs, write a program to determine whether the driver is to be insured or not.

<https://ideone.com/sh7j2i>

# Example – Logical Operators

```
/* Insurance of driver - using logical operators */
main( )
{
    char sex, ms ;
    int age ;
    printf ( "Enter age, sex, marital status " ) ;
    scanf ( "%d %c %c" &age, &sex, &ms ) ;

    if ( ( ms == 'M') || ( ms == 'U' && sex == 'M' && age
> 30 ) || ( ms == 'U' && sex == 'F' && age > 25 ) )
        printf ( "Driver is insured" ) ;

    else
        printf ( "Driver is not insured" ) ;

}
```

# Example – Logical Operators

<https://ideone.com/BH7NhK>

Write a program to calculate the salary as per the following table:

Gender	Years of Service	Qualifications	Salary
Male	$\geq 10$	Post-Graduate	15000
	$\geq 10$	Graduate	10000
	$< 10$	Post-Graduate	10000
	$< 10$	Graduate	7000
Female	$\geq 10$	Post-Graduate	12000
	$\geq 10$	Graduate	9000
	$< 10$	Post-Graduate	10000
	$< 10$	Graduate	6000

# The ! Operator

➤ This operator reverses the result of the expression it operates on.

- if the expression evaluates to a non-zero value, then applying ! operator to it results into a 0. Vice versa
- if the expression evaluates to zero then on applying ! operator to it makes it 1, a non-zero value.

- $!(y < 10)$

- “not y less than 10”

- $y \geq 10$ .

*if (!flag) is same as if ( flag == 0 )*

# Hierarchy of Operators

Operators	Type
!	Logical NOT
* / %	Arithmetic and modulus
+ -	Arithmetic
< > <= >=	Relational
== !=	Relational
&&	Logical AND
	Logical OR
=	Assignment



# Summarize Logical Operators

Operands		Results			
x	y	!x	!y	x && y	x    y
0	0	1	1	0	0
0	non-zero	1	0	0	1
non-zero	0	0	1	0	1
non-zero	non-zero	0	0	1	1

# The Conditional Operators

➤ Also known as ternary operator

➤ A short form of if-then-else

■ *expression 1 ? expression 2 : expression 3*

■ “if expression 1 is true (that is, if its value is non-zero), then the value returned will be expression 2, otherwise the value returned will be expression 3”.

```
int x, y ;  
scanf ( "%d", &x ) ;  
y = ( x > 5 ? 3 : 4 ) ;
```

```
if ( x > 5 )  
    y = 3 ;  
else  
    y = 4 ;
```

# More on Conditional Operators

```
char a ;  
int y ;  
scanf ( "%c", &a ) ;  
y = ( a >= 65 && a <= 90 ? 1 : 0 ) ;
```

1 would be assigned to y if `a >= 65 && a <= 90` evaluates to true, otherwise 0 would be assigned.

- It's not necessary that the conditional operators should be used only in arithmetic statements.

```
int i ;  
scanf ( "%d", &i ) ;  
( i == 1 ? printf ( "Amit" ) : printf ( "All and sundry" ) ) ;
```

# More on Conditional Operators

The conditional operators can be nested as shown below.

```
int big, a, b, c ;  
big = ( a > b ? ( a > c ? 3: 4 ) : ( b > c ? 6: 8 ) ) ;
```

Check the following statement:

```
a > b ? g = a : g = b ;
```

- Error: 'Lvalue Required'.
- Can be removed by adding parenthesis
  - *a > b ? g = a : ( g = b ) ;*
- In absence of parentheses the compiler believes that **b** is being assigned to the result of the expression to the left of second =. Hence it reports an error.

# Exercise

➤ What will be output of the following program

```
# include <stdio.h>
```

```
main( )
```

```
{
```

```
    int a = 300, b, c ;
```

```
    if ( a >= 400 )
```

```
        b = 300 ;
```

```
    c = 200 ;
```

```
    printf ( "\n%d %d", b, c ) ;
```

```
}
```

# Exercise

➤ What will be output of the following program

```
# include <stdio.h>
```

```
main( )
```

```
{
```

```
    int a = 500, b, c ;
```

```
    if ( a >= 400 )
```

```
        b = 300 ;
```

```
    c = 200 ;
```

```
    printf ( "\n%d %d", b, c ) ;
```

```
}
```

# Exercise

➤ What will be output of the following program

```
# include <stdio.h>
```

```
main( )
```

```
{
```

```
    int x =3;
```

```
    float y = 3.0;
```

```
    if ( x == y )
```

```
        printf ("x and y are equal\n") ;
```

```
    else
```

```
        printf("x and y are not equal\n");
```

```
}
```

# Exercise

➤ What will be output of the following program

```
# include <stdio.h>
```

```
main( )
```

```
{
```

```
    int x =3, y, z;
```

```
    y = x = 10;
```

```
    z = x < 10;
```

```
    printf (“x = %d, y = %d, z = %d\n”, x, y, z) ;
```

```
}
```



# Exercise

➤ What will be output of the following program

```
# include <stdio.h>
```

```
main( )
```

```
{
```

```
    int x = 10, y = 20, z = 5, i;
```

```
    i = x < y < z;
```

```
    printf("%d\n", i);
```

```
}
```

# Exercise

➤ What will be output of the following program

```
#include <stdio.h>
```

```
int X=40;
```

```
main()
```

```
{
```

```
    int X=20;
```

```
    printf("%d\n", X);
```

```
}
```

# Exercise

➤ What will be output of the following program

```
#include <stdio.h>
```

```
main( )
```

```
{
```

```
    int i = 65 ;
```

```
    char j = 'A' ;
```

```
    if ( i == j )
```

```
        printf ( "C is WOW" ) ;
```

```
    else
```

```
        printf( "C is a headache" ) ;
```

```
}
```

# Exercise

➤ What will be output of the following program

```
#include <stdio.h>
```

```
main( )
```

```
{
```

```
    float a = 12.25, b = 12.52 ;
```

```
    if ( a = b )
```

```
        printf ( "\na and b are equal" ) ;
```

```
}
```

# Exercise

➤ What will be output of the following program

```
#include <stdio.h>
```

```
main( )
```

```
{
```

```
    int j = 10, k = 12 ;
```

```
    if ( k >= j )
```

```
    {
```

```
        {
```

```
            k = j ;
```

```
            j = k ;
```

```
        }
```

```
    }
```

```
}
```

# Exercise

➤ What will be output of the following program

```
#include <stdio.h>
```

```
main( )
```

```
{
```

```
    if ( 'X' < 'x' )
```

```
        printf ( "\nascii value of X is smaller than  
that of x" ) ;
```

```
}
```

# Exercise

➤ What will be output of the following program

```
#include <stdio.h>
```

```
main( )
```

```
{
```

```
    int x = 10 ;
```

```
    if ( x >= 2 ) then
```

```
        printf ( "\n%d", x ) ;
```

```
}
```

# Exercise

➤ What will be output of the following program

```
#include <stdio.h>
```

```
main( )
```

```
{
```

```
    int x = 10, y = 15 ;
```

```
    if ( x % 2 = y % 3 )
```

```
        printf ( "\nCarpathians" ) ;
```

```
}
```



# Exercise

➤ What will be output of the following program

```
#include <stdio.h>

main( )
{
    int x = 30 , y = 40 ;
    if ( x == y )
        printf( "x is equal to y" ) ;
    elseif ( x > y )
        printf( "x is greater than y" ) ;
    elseif ( x < y )
        printf( "x is less than y" ) ;
}
```

## Exercise

- Any integer is input through the keyword.  
Write a program to find out whether it is an odd or even number.

<https://ideone.com/FMV8bS>

# Exercise: Guess the Output

```
main( )
{
    int i = 4, z = 12 ;

    if ( i = 5 || z > 50 )
        printf ( "\nDean of students affairs" ) ;
    else
        printf ( "\nDosa" ) ;
}
```

# Exercise: Guess the Output

```
main( )  
{  
    int i = 4, j = -1, k = 0, w, x, y, z ;  
  
    w = i || j || k ;  
    x = i && j && k ;  
    y = i || j && k ;  
    z = i && j || k ;  
  
    printf ( "\nw = %d x = %d y = %d z = %d",  
w, x, y, z ) ;  
}
```

# Exercise: Guess the Output

```
main( )  
{  
    int x = 20 , y = 40 , z = 45 ;  
    if ( x > y && x > z )  
        printf( "x is big" ) ;  
    else if ( y > x && y > z )  
        printf( "y is big" ) ;  
    else if ( z > x && z > y )  
        printf( "z is big" ) ;  
}
```


# Exercise: Guess the Output

```
main( )
{
    char spy = 'a', password = 'z' ;
    if ( spy == 'a' or password == 'z' )
        printf ( "\nAll the birds are safe in the
nest" ) ;
}
```

# Exercise: Guess the Output

```
main( )
{
    int x = 2;
    if ( x == 2 && x != 0 ) ;
    {
        printf ( "\nHi" ) ;
        printf( "\nHello" ) ;
    }
    else
        printf( "Bye" ) ;
}
```

# Exercise: Guess the Output



```
main( )
{
    int i = 10, j ;
    i >= 5 ? j = 10 : j = 15 ;
    printf ( "\n%d %d", i, j ) ;
}
```



# Exercise: Guess the Output

```
main( )
{
    int a = 5 , b = 6 ;
    ( a == b ? printf( "%d",a) ) ;
}
```