CSE101-Lec#1

Programming Methodologies
C Program Development Environment
Algorithm and Pseudo code

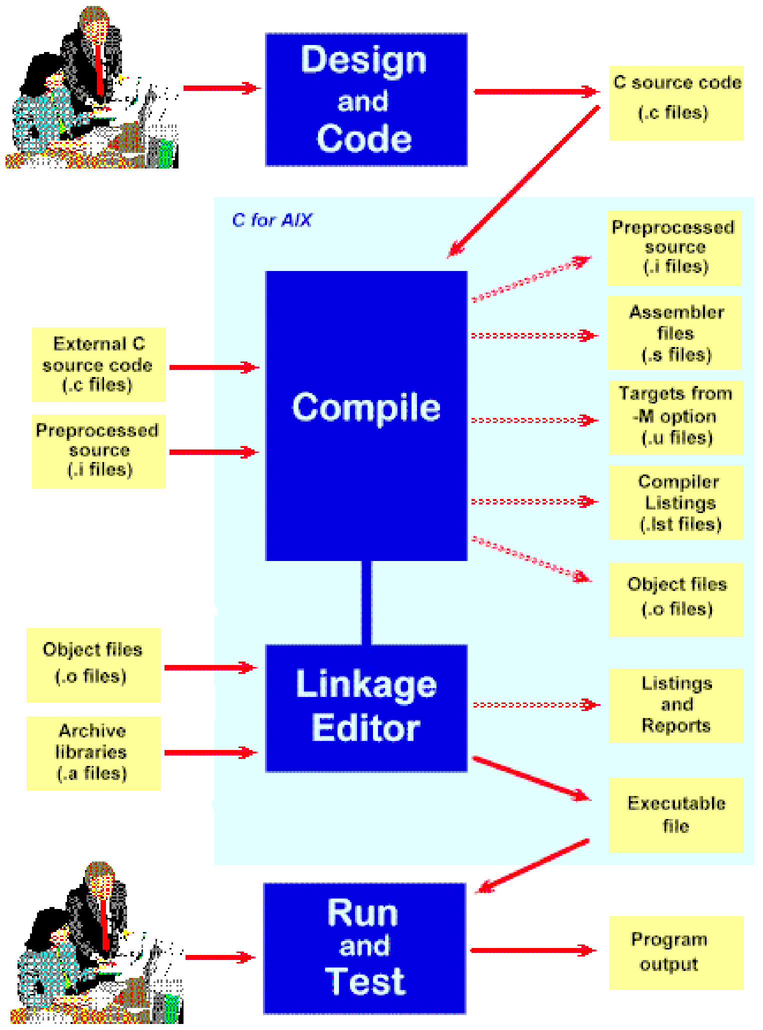
Outline

- Program Development Environment of C
- Structured programming
- Algorithms

C Program Development Environment

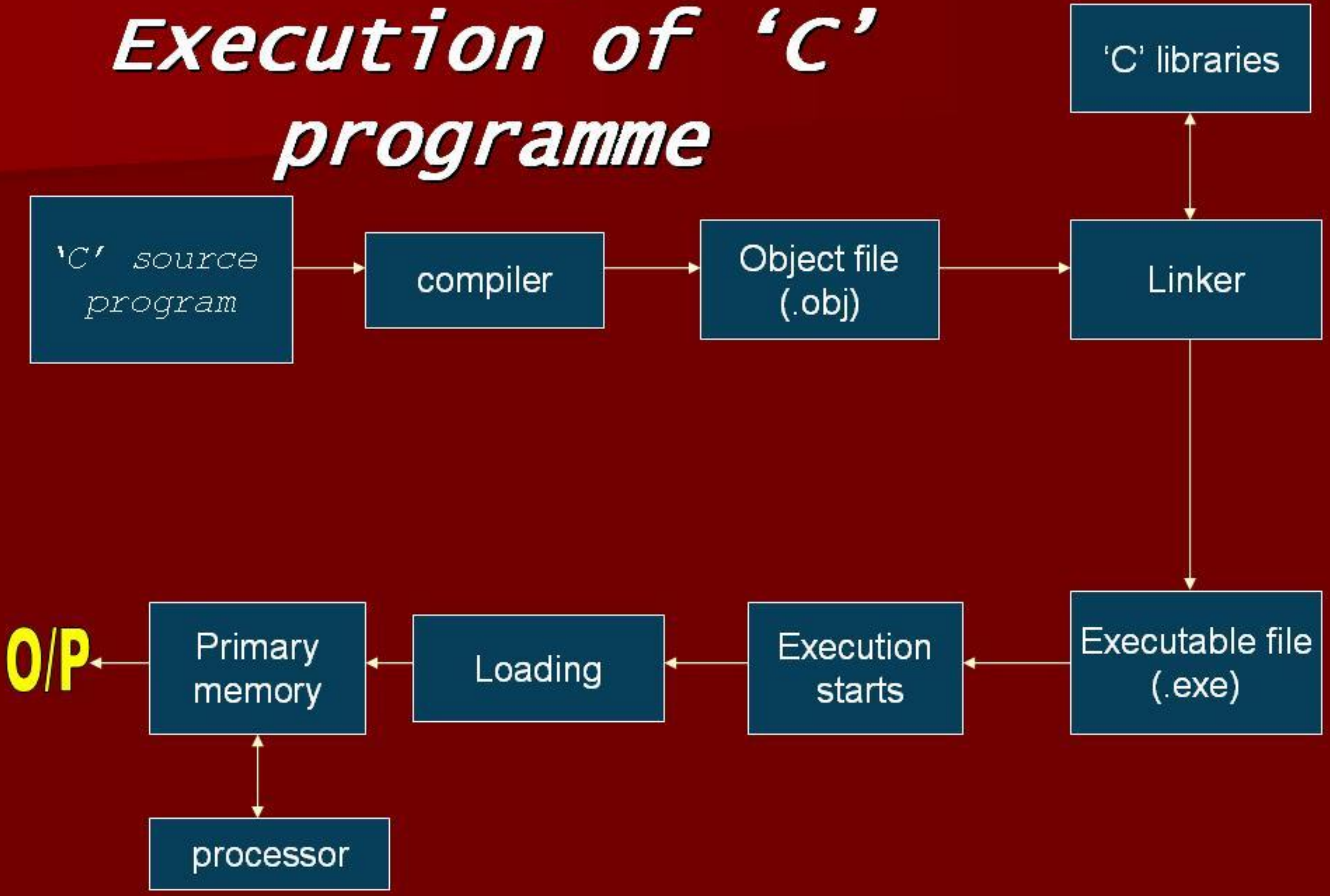
A C program must go through various phases such as:

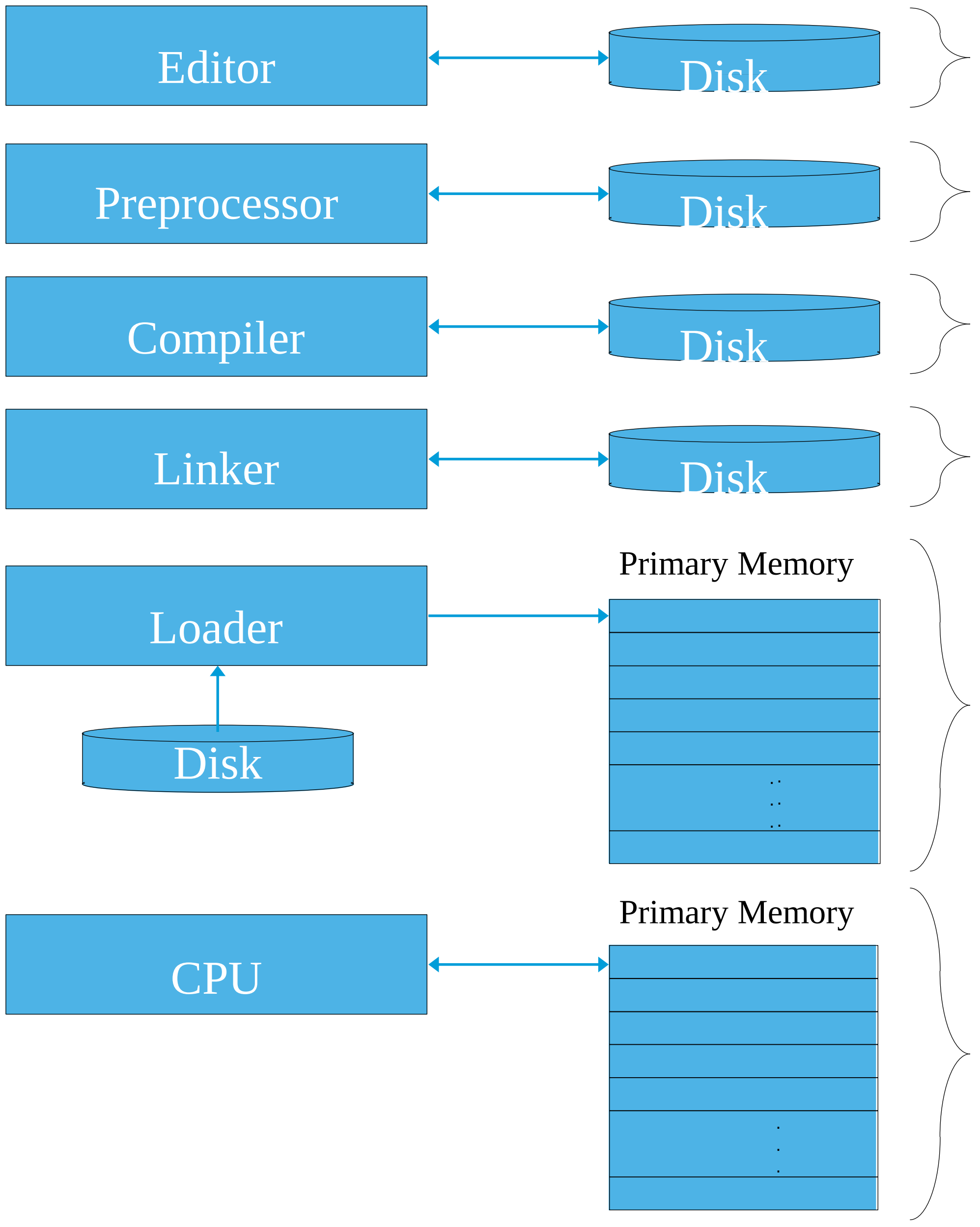
- Creating a program with editor
- Execution of preprocessor program
- Compilation
- Linking
- Loading
- Executing



Design and Code
C source code (.c files)
C for AIX
Preprocessed source (.i files)
Assembler files (.s files)
External C source code (.c files)
Compile
Targets from -M option (.u files)
Preprocessed source (.i files)
Compiler Listings (.lst files)
Object files (.o files)
Object files (.o files)
Linkage Editor
Listings and Reports
Archive libraries (.a files)
Executable file
Run and Test
Program output

Block Diagram of Execution of 'C' programme





Program is created in the editor and stored on disk.

Preprocessor program processes the code.

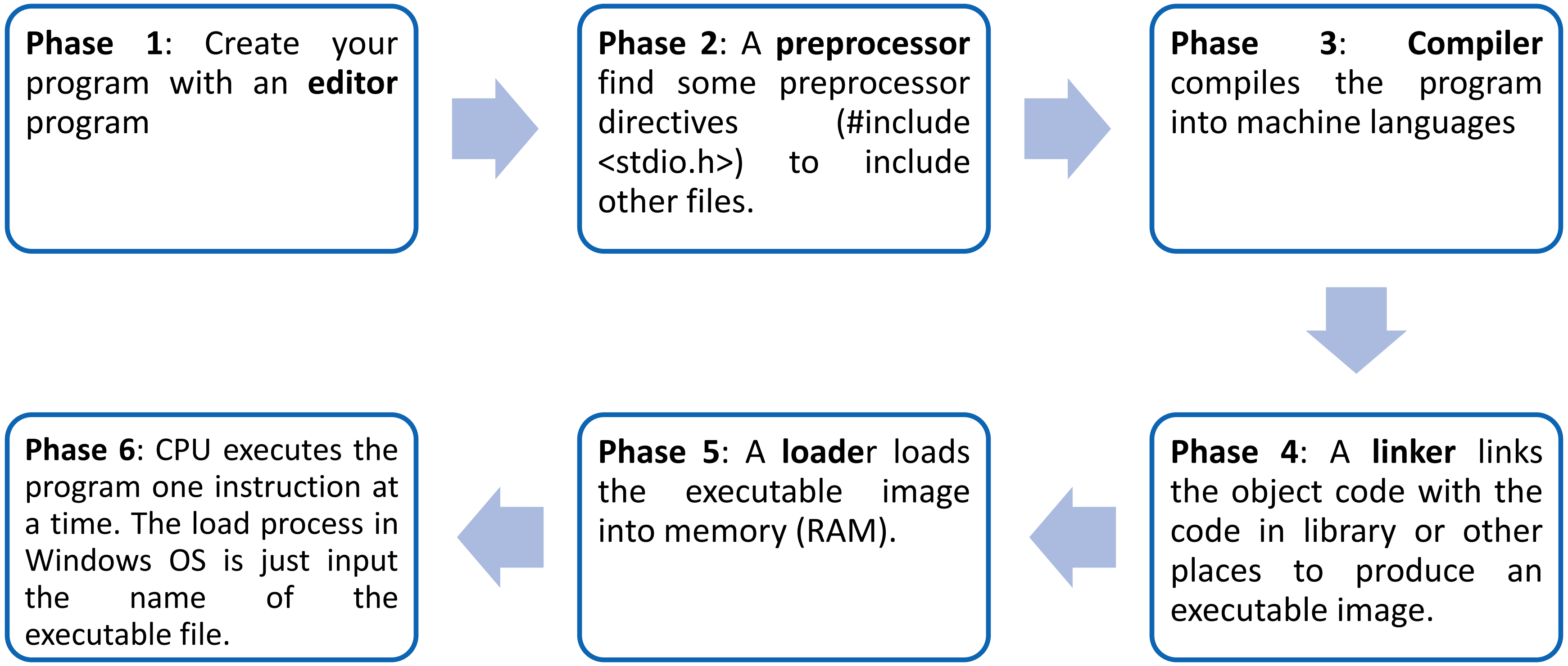
Compiler creates object code and stores it on disk.

Linker links the object code with the libraries,

Loader puts program in memory.

CPU takes each instruction and executes it, possibly storing new data values as the program executes.

C Program Development Environment(cont.)



Phase 1: Create your program with an **editor** program

Phase 2: A **preprocessor** find some preprocessor directives (#include <stdio.h>) to include other files.

Phase 3: **Compiler** compiles the program into machine languages

Phase 4: A **linker** links the object code with the code in library or other places to produce an executable image.

Phase 5: A **loader** loads the executable image into memory (RAM).

Phase 6: CPU executes the program one instruction at a time. The load process in Windows OS is just input the name of the executable file.

Program Development Tools

- Algorithm
- Flow chart
- Pseudo-code

Algorithm

- Algorithm is defined as " *the finite set of steps, which provide a chain of action for solving a problem*"
- It is *step by step solution* to given problem.
- Well organized, pre-arranged and defined textual computational module

Characteristics of good Algorithm

1. **Correctness** - terminates on ALL inputs (even invalid inputs!) and outputs the correct answer.
2. **Simplicity** - each step of the algorithm performs one logical step in solving the problem.
3. **Precision** - each step of the algorithm is unambiguous in meaning.
4. **Comprehensibility** - the algorithm is easy to read and understand.
5. **Abstraction** - presents the solution steps precisely and concisely without referring to low-level (program code) details.
6. **Efficient** - Gives results rapidly based on the problem size; does not waste any space or time.
7. **Easy to Implement** - relatively easy to translate into a programming language.

Steps to create an Algorithm

1. Identify the Inputs

- What data do I need?
- How will I get the data?
- In what format will the data be?

2. Identify the Outputs

- What outputs do I need to return to the user?
- What format should the outputs take?

Steps to create an Algorithm

3. Identify the Processes

- How can I manipulate data to produce meaningful results?
- Data vs. Information

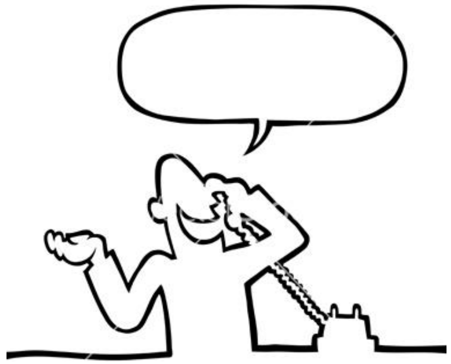
4. Break the Solution to steps

By breaking the solution to the steps we can easily understand the logic of program

Example of Algorithm

To establish a telephone communication

- Step 1: Dial a phone number
- Step 2: Phone rings at the called party
- Step 3: Caller waits for the response
- Step 4: Called party picks up the phone
- Step 5: Conversation begins between them
- Step 6: After the conversation, both disconnect the call



Pseudocode

- Pseudocode is similar to everyday English language, its convenient and user friendly.
- Informal language that helps programmers develop algorithms. Pseudocode is a "text-based" detail (algorithmic) design tool.
- It's *not a actual computer programming language.*
- Pseudocode consist only of action statements :
  - Each of steps will be written via operators and statements equivalent to some programming language instructions.
  - The only difference will be that the exact syntax of the programming language will not be followed.

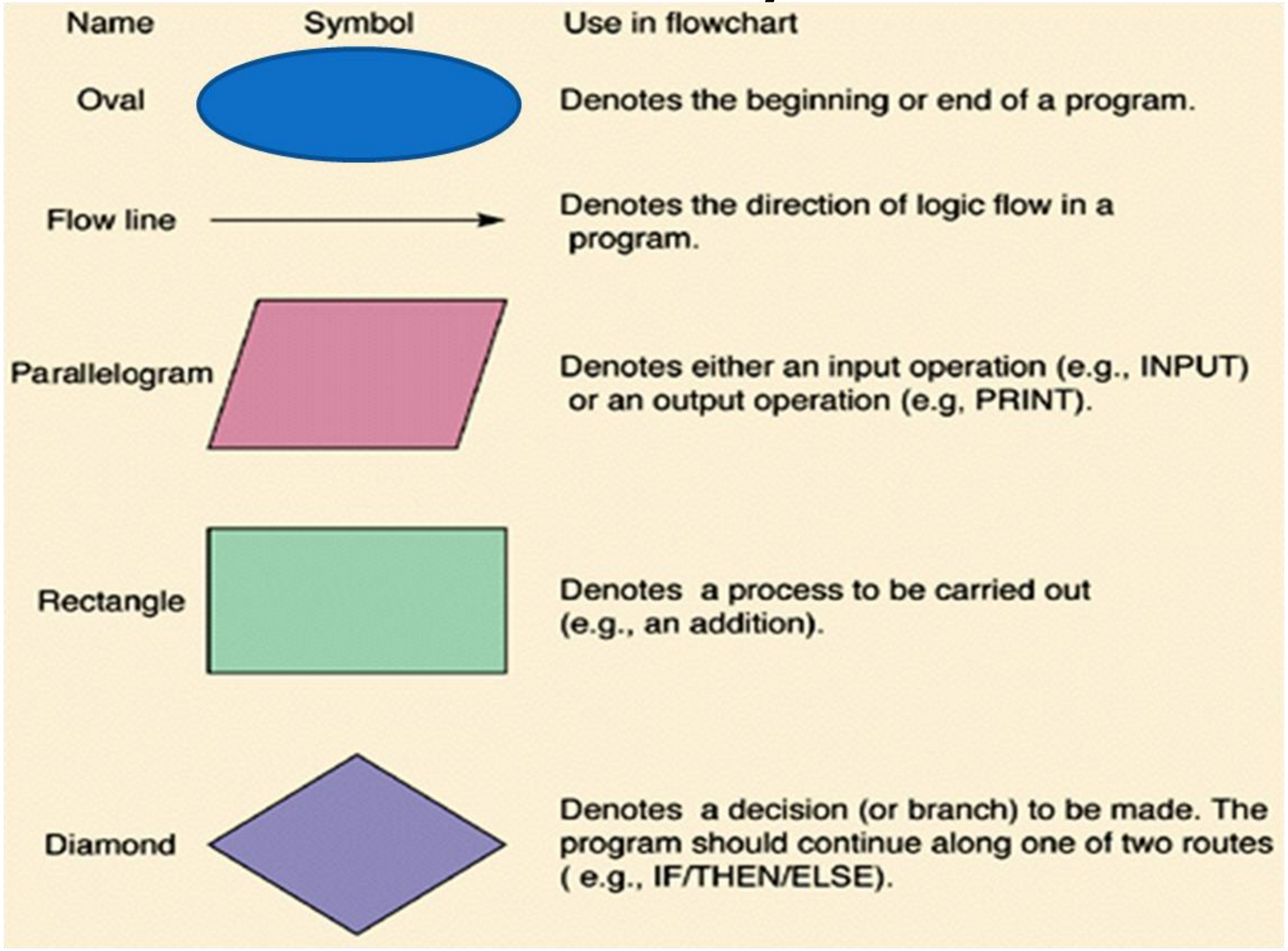
Pseudo code: Add 2 Numbers

- Set A = 4.
- Set B = 2.
- Calculate Sum = A + B.
- Print Sum.

Flow Chart

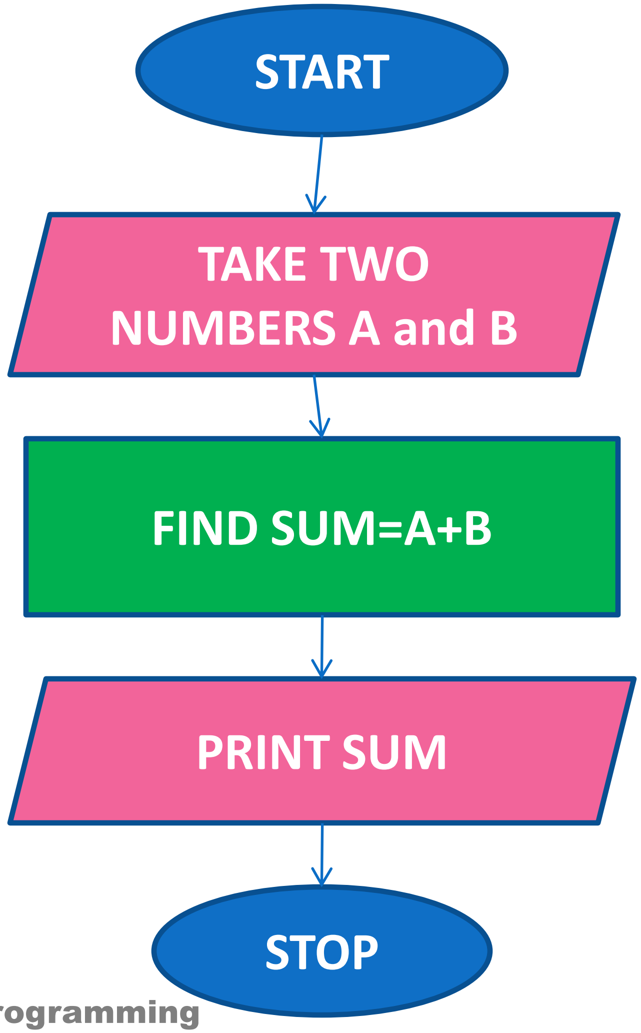
- Flow Chart is pictorial representation of an algorithm.
- Whatever we have done in algorithm we can represent it in picture.
- It is easy to understand.
- Shows the flow of the instruction

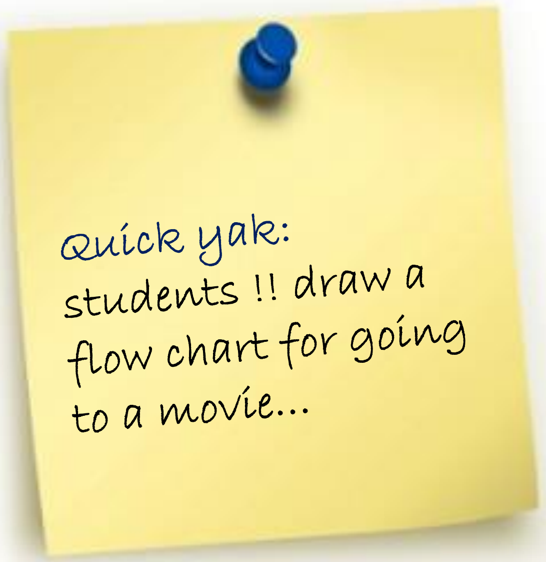
Flow Chart Symbols



Name	Symbol	Use in flowchart
Oval	⬭	Denotes the beginning or end of a program.
Flow line	→	Denotes the direction of logic flow in a program.
Parallelogram	▱	Denotes either an input operation (e.g., INPUT) or an output operation (e.g, PRINT).
Rectangle	▭	Denotes a process to be carried out (e.g., an addition).
Diamond	◇	Denotes a decision (or branch) to be made. The program should continue along one of two routes (e.g., IF/THEN/ELSE).

Flow Chart: Add 2 Numbers





Program in C: Add 2 Numbers

```c
#include<stdio.h>
void main()
{
 int a=4;
 int b= 2;
 int sum;
 sum=a+b;
 printf("Sum is: %d", sum);
}
```

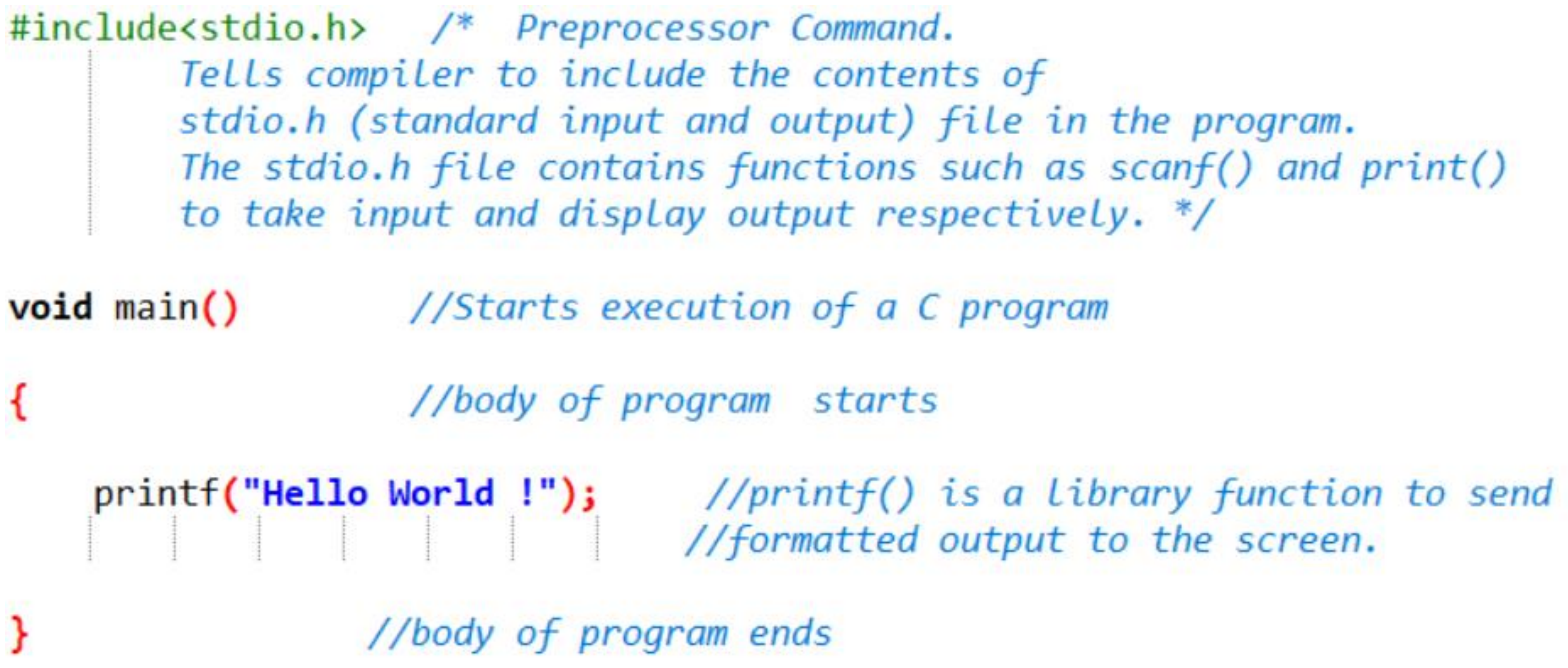
```
Sum is: 6
```

Program in C

```
#include<stdio.h>
#include<conio.h>
int main()
{
 int pricePerKg = 45;
 float quantityInKg = 7.5;
 float total_price;
 total_price = pricePerKg * quantityInKg;
 printf(“%f”, total_price );
 getch();
return 0;
}
```

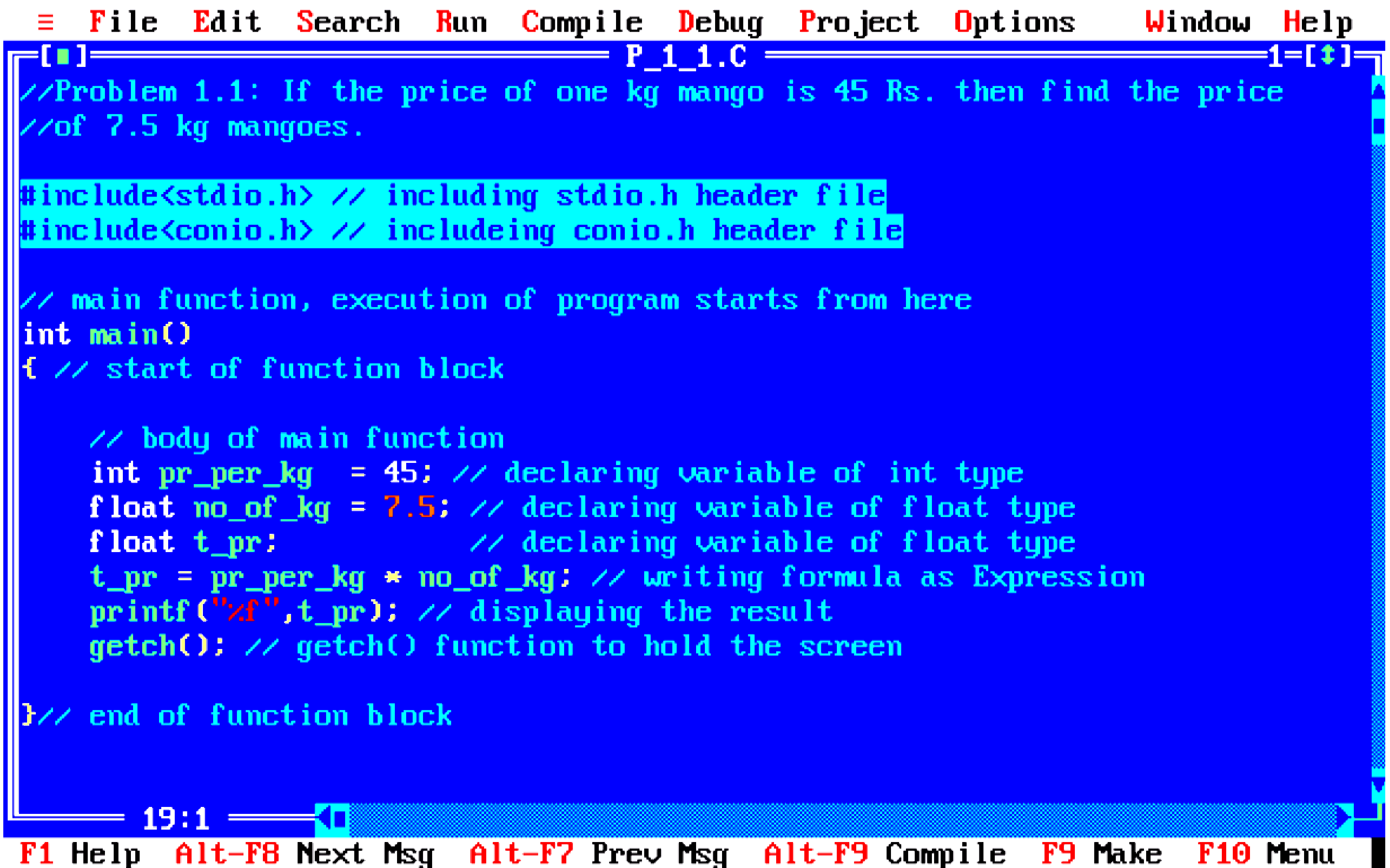
```
337.5
```

Example



```
#include<stdio.h>    /*  Preprocessor Command.
          Tells compiler to include the contents of
          stdio.h (standard input and output) file in the program.
          The stdio.h file contains functions such as scanf() and print()
          to take input and display output respectively. */

void main()         //Starts execution of a C program

{                   //body of program  starts

    printf("Hello World !");      //printf() is a library function to send
                                 //formatted output to the screen.

}                 //body of program ends
```

Structure of C program

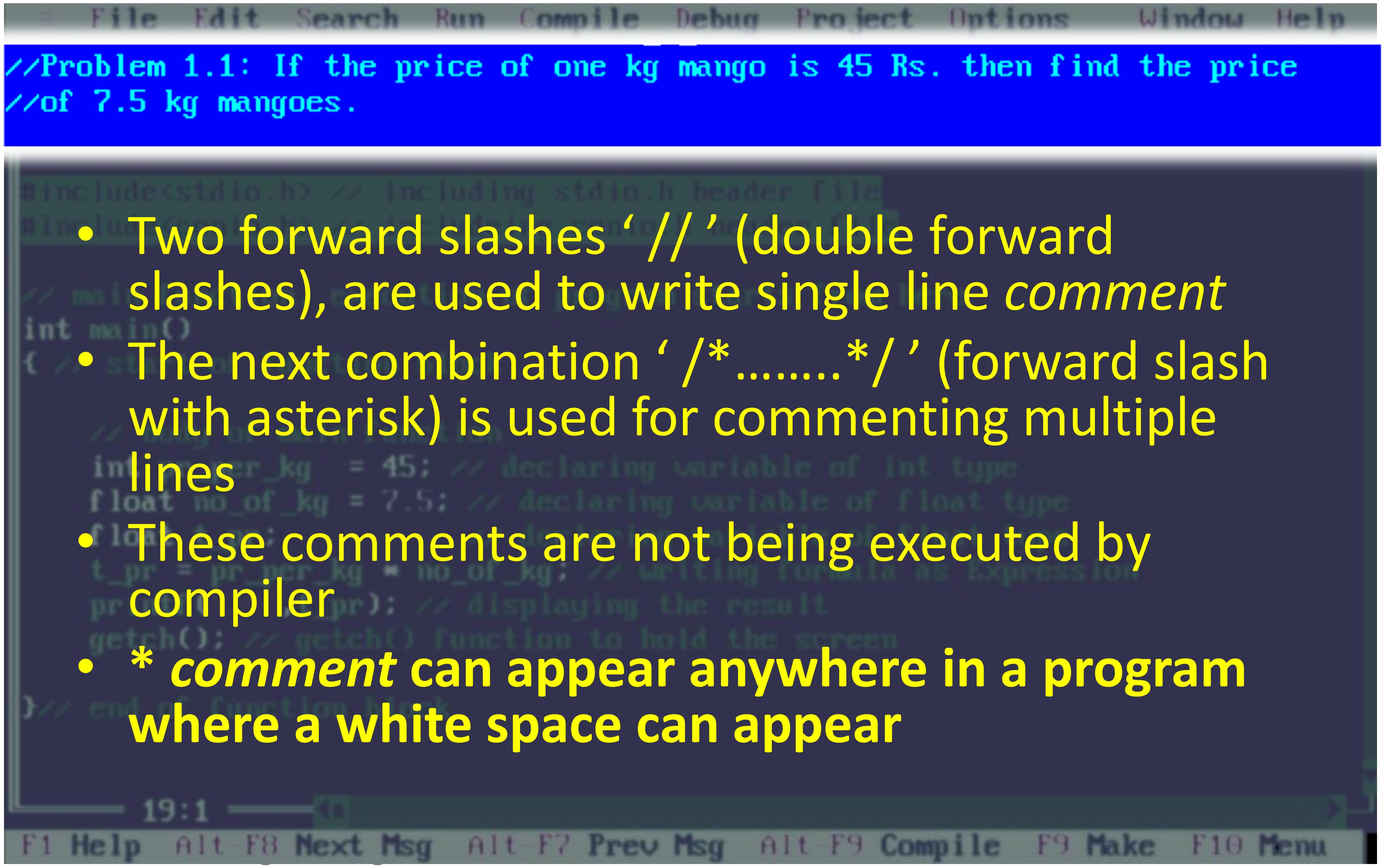

≡ File Edit Search Run Compile Debug Project Options Window Help
[■] P_1_1.C 1=[↕]
19:1
F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt-F9 Compile F9 Make F10 Menu

```
//Problem 1.1: If the price of one kg mango is 45 Rs. then find the price
//of 7.5 kg mangoes.

#include<stdio.h> // including stdio.h header file
#include<conio.h> // includeing conio.h header file

// main function, execution of program starts from here
int main()
{ // start of function block

    // body of main function
    int pr_per_kg  = 45; // declaring variable of int type
    float no_of_kg = 7.5; // declaring variable of float type
    float t_pr;           // declaring variable of float type
    t_pr = pr_per_kg * no_of_kg; // writing formula as Expression
    printf("%f",t_pr); // displaying the result
    getch(); // getch() function to hold the screen

}// end of function block
```

Explanation

- A simple C program consists of
  - *Comments (optional)*
    - //
    - /*….*/
  - *Including header files*
    - #include<header file name>
  - *Functions*
    - main function as special function
    - Other user defined functions (optional)
- Let’s discuss these in detail..

Comments



File Edit Search Run Compile Debug Project Options Window Help

```
//Problem 1.1: If the price of one kg mango is 45 Rs. then find the price
//of 7.5 kg mangoes.

#include<stdio.h> // including stdio.h header file
#inclu[illegible]

// ma[illegible]
int main()
{ // s[illegible]
  // [illegible]
  in[illegible]er_kg   = 45; // declaring variable of int type
  float no_of_kg = 7.5; // declaring variable of float type
  flo[illegible]
  t_pr = pr_per_kg * no_of_kg; // writing formula as expression
  pr[illegible]pr); // displaying the result
  getch(); // getch() function to hold the screen
}// end of function block
```

19:1
F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt-F9 Compile F9 Make F10 Menu

- Two forward slashes ‘ // ’ (double forward slashes), are used to write single line *comment*
- The next combination ‘ /*……..*/ ’ (forward slash with asterisk) is used for commenting multiple lines
- These comments are not being executed by compiler
- *** *comment* can appear anywhere in a program where a white space can appear**

Comments

Enhances readability of program

Real life example

Fuel consumption:
16.0 L/100 km

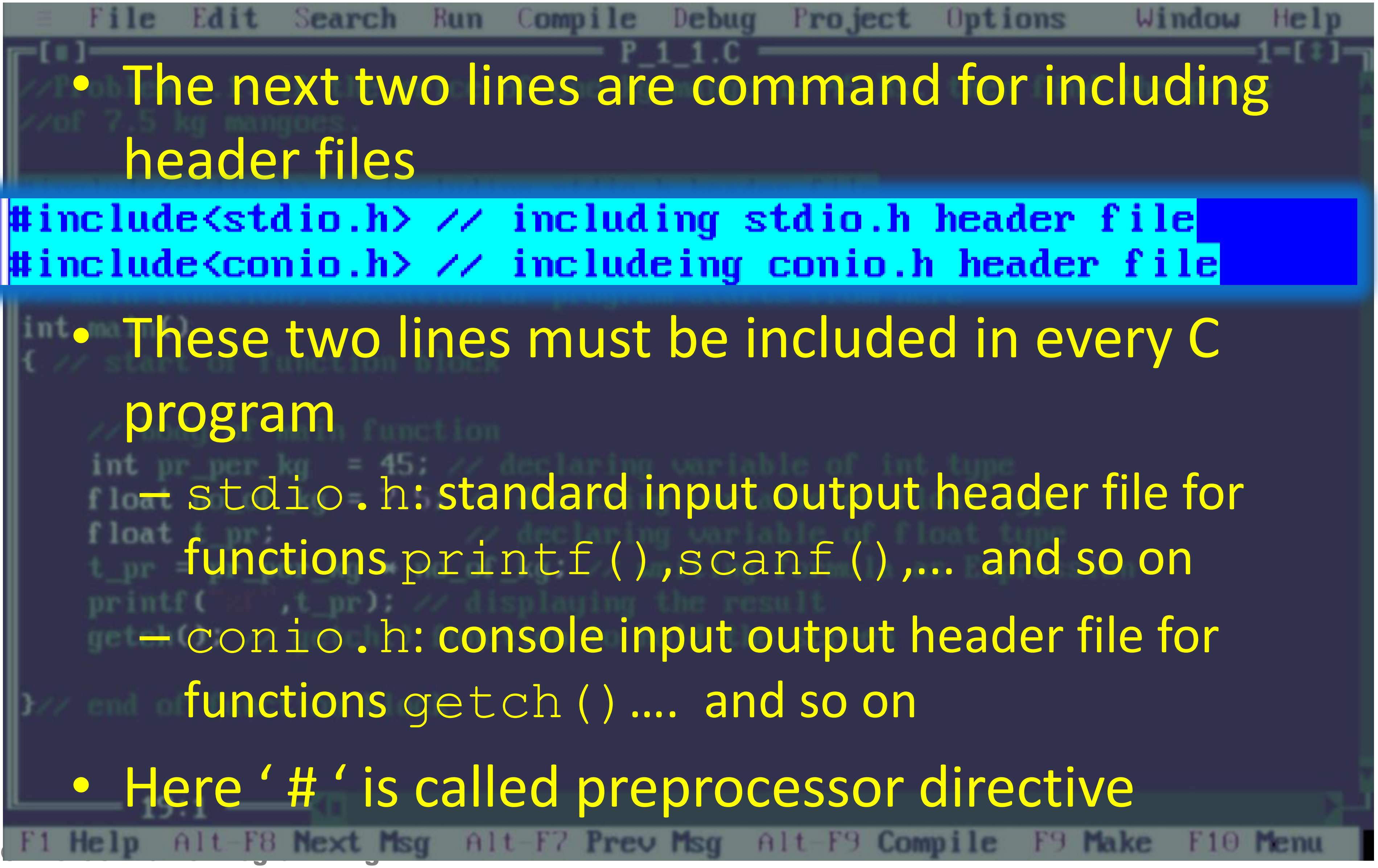
Carbon fiber
engine bonnet

Front
tyre: 255/35
ZR19

C code example

```
//Prog. Name: addition of integers
//Another format
/*Prog. Name: addition of integers
Student Name: Mr J. Bond
UID: 007
Section: M4571 */
#include<stdio.h>
#include<conio.h>
int main()
{
}
```

Header files


File Edit Search Run Compile Debug Project Options Window Help
P_1_1.C
F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt-F9 Compile F9 Make F10 Menu

- The next two lines are command for including header files

```
#include<stdio.h> // including stdio.h header file
#include<conio.h> // includeing conio.h header file
```

- These two lines must be included in every C program
  - `stdio.h`: standard input output header file for functions `printf()`,`scanf()`,… and so on
  - `conio.h`: console input output header file for functions `getch()`…. and so on
- Here ‘ # ‘ is called preprocessor directive

Header files

Real life example

Header for tires

Header for oil

Header for speakers

Ferrari car as output

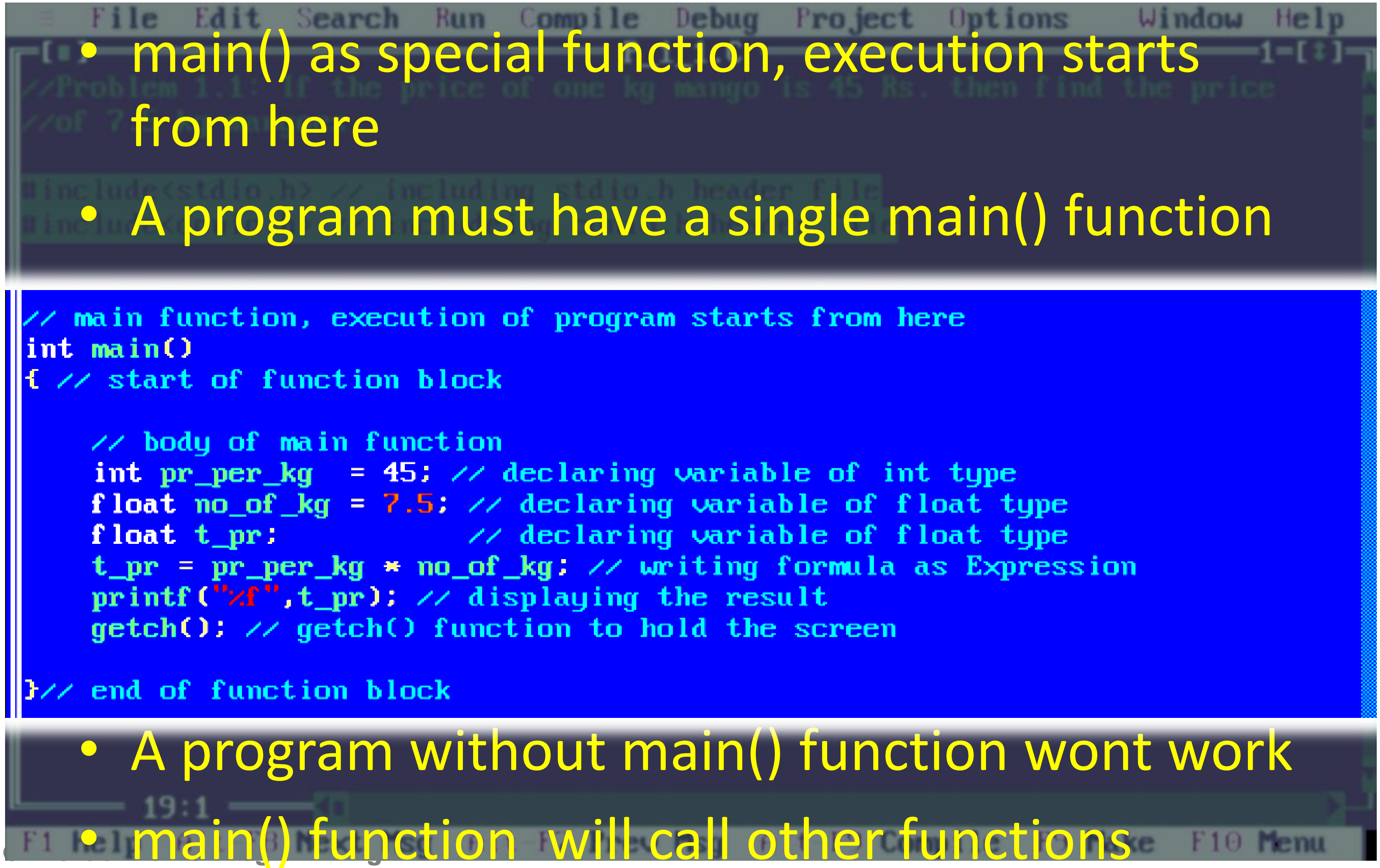
C code example

```
//Sample program
#include<stdio.h>  //header file for printf()
#include<conio.h> //header file for getch()
int main()
{
   //stdio.h is providing printf() function
    printf("Car is under process");
    //conio.h is providing getch() function
    getch();
}
```

Output:

```
Car is under process
```

Function


File Edit Search Run Compile Debug Project Options Window Help
19:1
F1 Help
F10 Menu

- main() as special function, execution starts from here
- A program must have a single main() function

```
// main function, execution of program starts from here
int main()
{ // start of function block

    // body of main function
    int pr_per_kg  = 45; // declaring variable of int type
    float no_of_kg = 7.5; // declaring variable of float type
    float t_pr;           // declaring variable of float type
    t_pr = pr_per_kg * no_of_kg; // writing formula as Expression
    printf("%f",t_pr); // displaying the result
    getch(); // getch() function to hold the screen

}// end of function block
```

- A program without main() function wont work
- main() function will call other functions

main function in real life

Real life example

Before driving the car, we must start the car, a car won’t be driven without starting

C code example

Similarly our program must have a main function to start a program

```
//Prog. Name: display name
#include<stdio.h>
#include<conio.h>
int main()
{
printf(“My name is Aman”);
getch();
}
```

Output:

```
My name is Aman
```

Next Class: Components of C Identifier and Keywords Data Type

cse101@lpu.co.in