

**An Autonomous Institution**

Accredited by NBA – AICTE and Accredited by NAAC – UGC with ‘A+’ Grade  
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

## **DEPARTMENT OF AIML**

# **PROBLEM SOLVING AND C PROGRAMMING**

**I YEAR - I SEM**

## **UNIT 2 – C Programming Basics**

**TOPIC 2 – Structure & Execution of a ‘C’ program**

Documentation Section

---

Link Section

---

Definition Section

---

Global Declaration Section

---

main() Function Section

{

Declaration Part
------------------

Executable Part
-----------------

}

---

Subprogram Section

Function 1
------------

Function 2
------------

-
---

Function N
------------

can be viewed as a group of building blocks called functions.

Function is a subroutine that may include one or more statements designed to accomplish a specific task.

### Documentation Section

Documentation section consists of a set of **comment lines** giving information about the program, the author and other details, which the programmer would like to have.

### Link Section

Link section provides instructions to the **compiler to link functions** together.

### Preprocessor Section

Preprocessor section defines all symbolic constants.

## Declaration Section

There are some variables that are used in **more than one function**.

These variables are called global variables and are declared in the global scope

that is **outside of all the functions**.

The main function also declares all the user-defined functions

## Function Section

Every program **must have one main() function** section.

The main function contains two parts:

### Declaration part

It declares all **the variables** used in the executable part

### Executable part

There should be at least **one statement** in the executable part.

Both parts must appear between the **opening and the closing braces**.

Program execution begins at the opening brace and ends at the closing brace.

g brace of the main function section is the logical end of the program. Statements in the declaration and executable parts **end with a semicolon**(

## Function Section

The function section contains all the user-defined functions that **are called** in the program.

User-defined functions are generally placed **immediately after the main function** in the program. They may appear in any order.

The function section, except the main function section may be absent when they are not used.

## BASIC STRUCTURE OF A 'C' PROGRAM:

**Documentation section**  
[Used for Comments]

**Link section**

**Definition section**

**Global declaration section**  
[Variable used in more than one function]

**main()**  
{  
**Declaration part**  
**Executable part**  
}

**Subprogram section**  
[User-defined Function]  
**Function 1**  
**Function 2**  
:  
:

## Example:

→ `//Sample Prog`

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

→ `void fun();`

→ `int a=10;`

→ `void main()`  
`{`  
`clrscr();`  
`printf("a value inside main(): %d", a);`  
`fun();`  
`}`

→ `void fun()`  
`{`  
`printf("\na value inside fun(): %d", a);`  
`}`

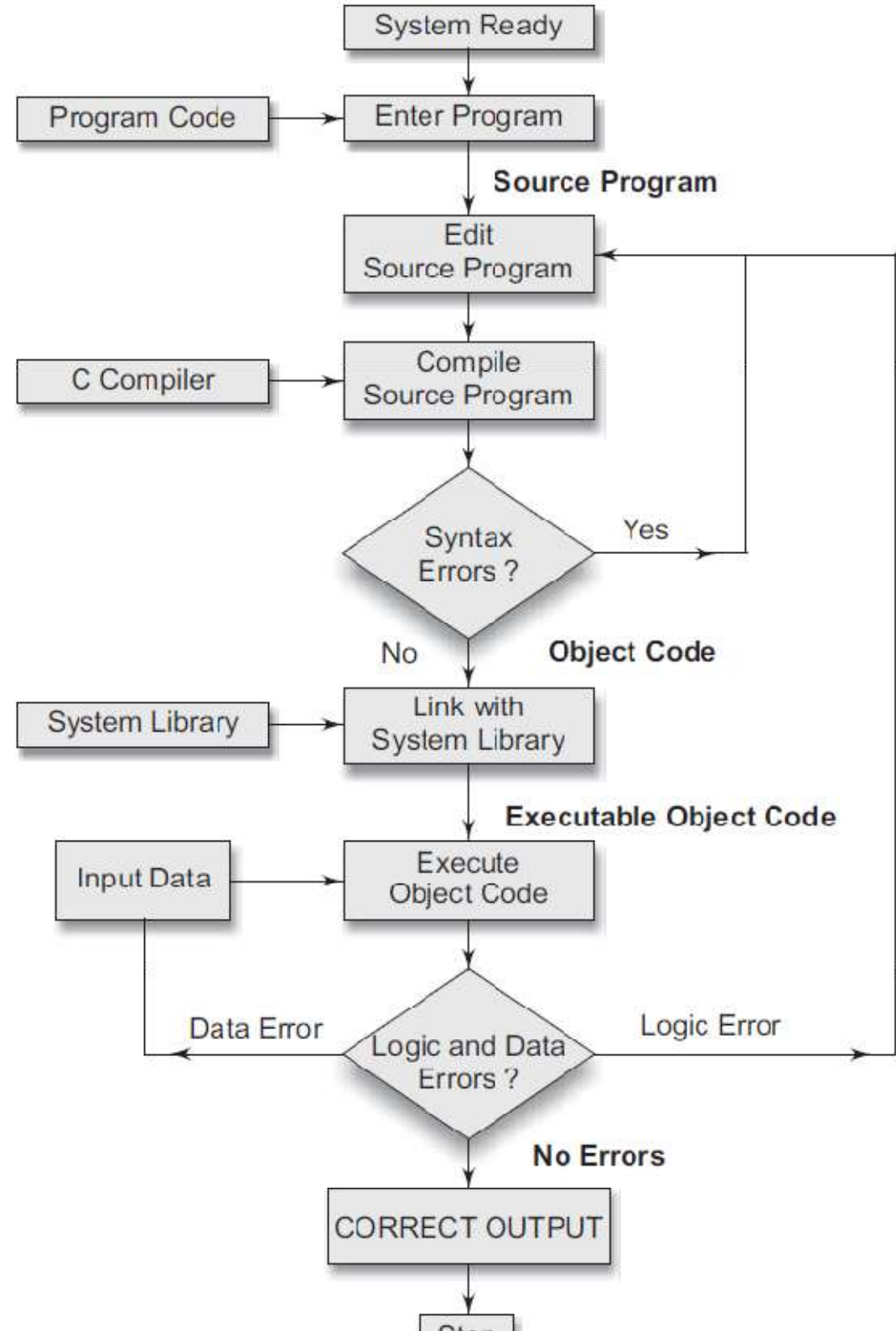
a program written in  
as a series of steps.

the program;

g the program;

the program with  
t are needed from the

d  
g the program.



## The Program

am must be **entered into a file**.

ame can consist of letters, digits and special characters, followed b

of valid file names are:

m.c

created with the help of a text editor (ex. notepad) and some **stand**  
**am that is entered into the file is known as the source program**  
the original form of the program



g

Assume that the source program has been created in a file named ebg1.p  
The program is ready for compilation.

The program instructions are **now translated into a form that is suitable**  
by the computer.

Verification is done after examining each instruction for its correctness.

If everything is alright, the compilation proceeds silently and the translated program  
is saved in a file with the name **ebg1.o**.

The program is known as **object code**.

the process of putting together other program files and functions to form a program.

For example, if the program is using `exp()` function, then the object code of this function is brought from the **math library** of the system and linked to the main program. Linking is **automatically done** (if no errors are detected) in most of the systems.

When all the errors in the **syntax and semantics** of the language are discovered, the compilation process ends right there.

Any errors that should be corrected in the source program with the help of the editor. After correction, compilation is done again.

The compiled and linked program is called the executable object code and is saved in another file named **a.out**.

Different systems use different compilation commands for linking various

## g the Program

a simple task

**executable object code** into the computer memory and execute the in  
ution, the program may **request for some data to be entered** thro

he program does not produce the desired results.

omething is wrong with the program logic or data.

ld be necessary to correct the source program or the data.

ource program is modified, the **entire process** of compiling, linking  
should be repeated.

e linker always assigns the same name a.out.

mpile another program, this file will be overwritten by the execut  
rogram.