# CSE 1121 Structured and Object Oriented Programming Language

Department of Computer Science and Engineering
DUET, Gazipur-1700

Md. Abu Bakkar Siddique
Lecturer, Department of CSE, DUET
Room # 7024, New Academic Building
Phone/Mobile: +880249274034-53 Ext: 3281, +8801944275646
Email: absiddique@duet.ac.bd, absduet@gmail.com

## ❖ Reference Books

1. Programming in ANSI C (6th Edition) by E Balagurusamy
2. Teach Yourself C++, Herbert Schildt, McGraw-Hill

## ❖ Class Hours

| Section-A | Section-B |
|---|---|
| **Theory:** 2 Lecture per week<br>Tuesday 10:55 AM – 11:50 AM<br>Thursday 10:55 AM – 11:50 AM<br>**Room No** – TWB 201<br>**Lab:** Sunday 8:00 AM – 10:25 AM/<br>01:50PM – 04:30PM<br>**Lab** – SMAD Lab | **Theory:** 2 Lecture per week<br>Tuesday 11:55 AM – 12:50 PM<br>Thursday 11:55 AM – 12:50 PM<br>**Room No** – TWB 218<br>**Lab:** Sunday 8:00 AM – 10:25 AM/<br>01:50PM – 04:30PM<br>**Lab** – NW Lab |

## ❖ Course Outline

1. Introduction and Course Overview
2. Revision of C Constants, Variable, data types, Operator & Expressions
3. Control structures and loops
4. Arrays, Pointer and Strings
5. User defined Function, Recursion, Structure and Union
6. Introduction to Object Oriented Programming Concept
7. Introduction to the Classes & Objects
8. Constructors, destructors and copy constructors
9. Array of objects, object pointers, and object references
10. Function & Operator overloading
11. Inheritance (single and multiple inheritances)
12. Polymorphism, abstract classes, virtual functions and overriding
13. Exception Handing
14. Template functions and classes
15. Object Oriented I/O
16. Multi-threaded Programming

## ❖ Marks Distribution

| Total Marks | 300 | |
|---|---|---|
| Class Attendance | 10% | 30 |
| Class Test | 20% | 60 |
| Final Exam | 70% | 210 |
| Total | 100% | 300 |

## ❖ Grading System

| Numerical Grade | Letter Grade | Grade point |
|---|---|---|
| 80% or above | A+ (A Plus) | 4.00 |
| 75% to less than 80% | A (A Regular ) | 3.75 |
| 70% to less than 75% | A- (A Minus) | 3.50 |
| 65% to less than 70% | B+ (B Plus) | 3.25 |
| 60% to less than 65% | B (B Regular) | 3.00 |
| 55% to less than 60% | B- (B Minus) | 2.75 |
| 50% to less than 55% | C+ (C Plus) | 2.50 |
| 45% to less than 50% | C (C Regular) | 2.25 |
| 40% to less than 45% | D | 2.00 |
| Less than 40% | F | 0.00 |

# Lecture 01 (Introduction to C programming)

- **Today's Outline**
  - **History of C**
  - **Importance of C**
  - **What is C used for?**
  - **C vs. Related language**
  - **Basic Structure of C program**
  - **“Hello World Program”**
  - **Process of creating, compiling and executing a C program**

# History of C

- ***C*** evolved from two previous programming languages, ***BCPL*** and ***B*** by "***Dennis Ritchie***" at ***Bell Laboratories*** on ***DEC PDP-11*** machine in ***1972***.
  - In ***1967***, BCPL (Basic Combined Programming Language) was developed by ***Martin Richards*** as a language for writing operating system software & compilers.
  - In ***1970***, ***Ken Thompson*** developed ***B*** using many features of ***BCPL*** and early versions of ***UNIX*** operating system at Bell Laboratories on ***DEC PDP-7*** machine.
- Initially, ***C*** widely known as the development language of the ***UNIX*** operating system but today virtually all new major operating systems are written in C and/or C++.
- By the late ***1970's***, C had evolved into what is now referred as "***Traditional C***".

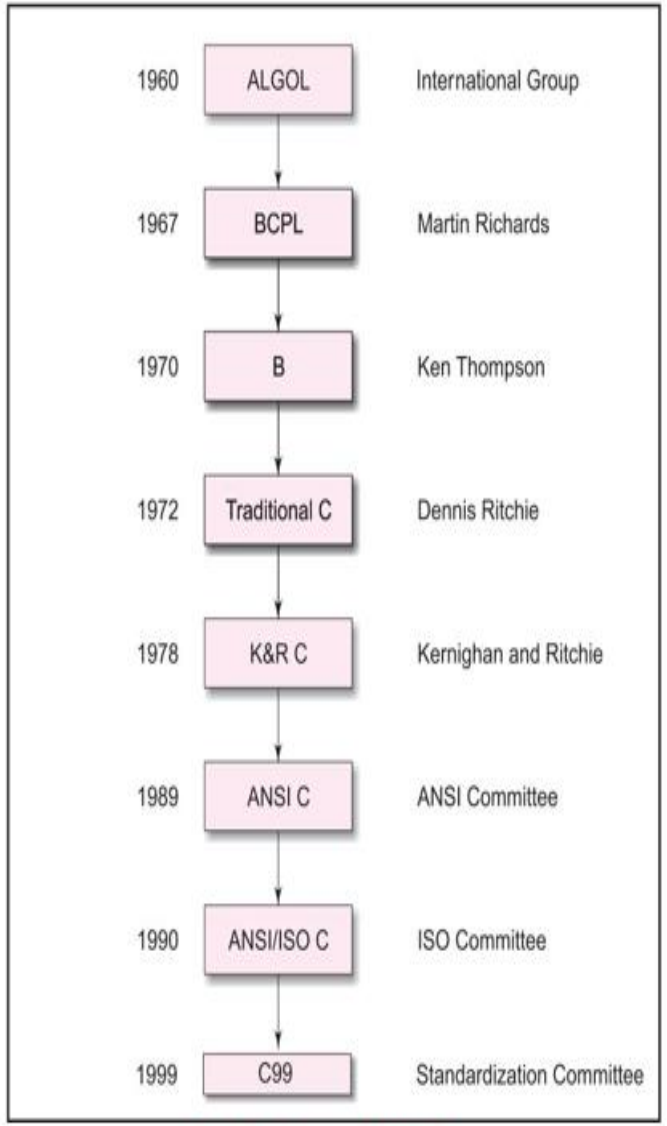



Fig. 1.1 *History of ANSI C*

# History of C (Cont.)

- Rapid expansion of C over various types of computers led to many variations of the language that were similar but often incompatible which was serious problem for program developers who needed to develop platform independent code.
- A technical committee (X3J11) was created under ***ANSI*** to define a standard version of C. In ***1989***, the committee approved a standard version of which was known as ***ANSI C (C89)***.
- In ***1990***, this standard was approved by ***ISO*** referred to as ***ANSI/ISO Standard C*** which was updated in ***1999*** referred to as ***C99.***
- In **2007** – work on new C standard C1X announced.
  - In this course: **ANSI/ISO Standard C (C89/C90)**

# Why teach C?

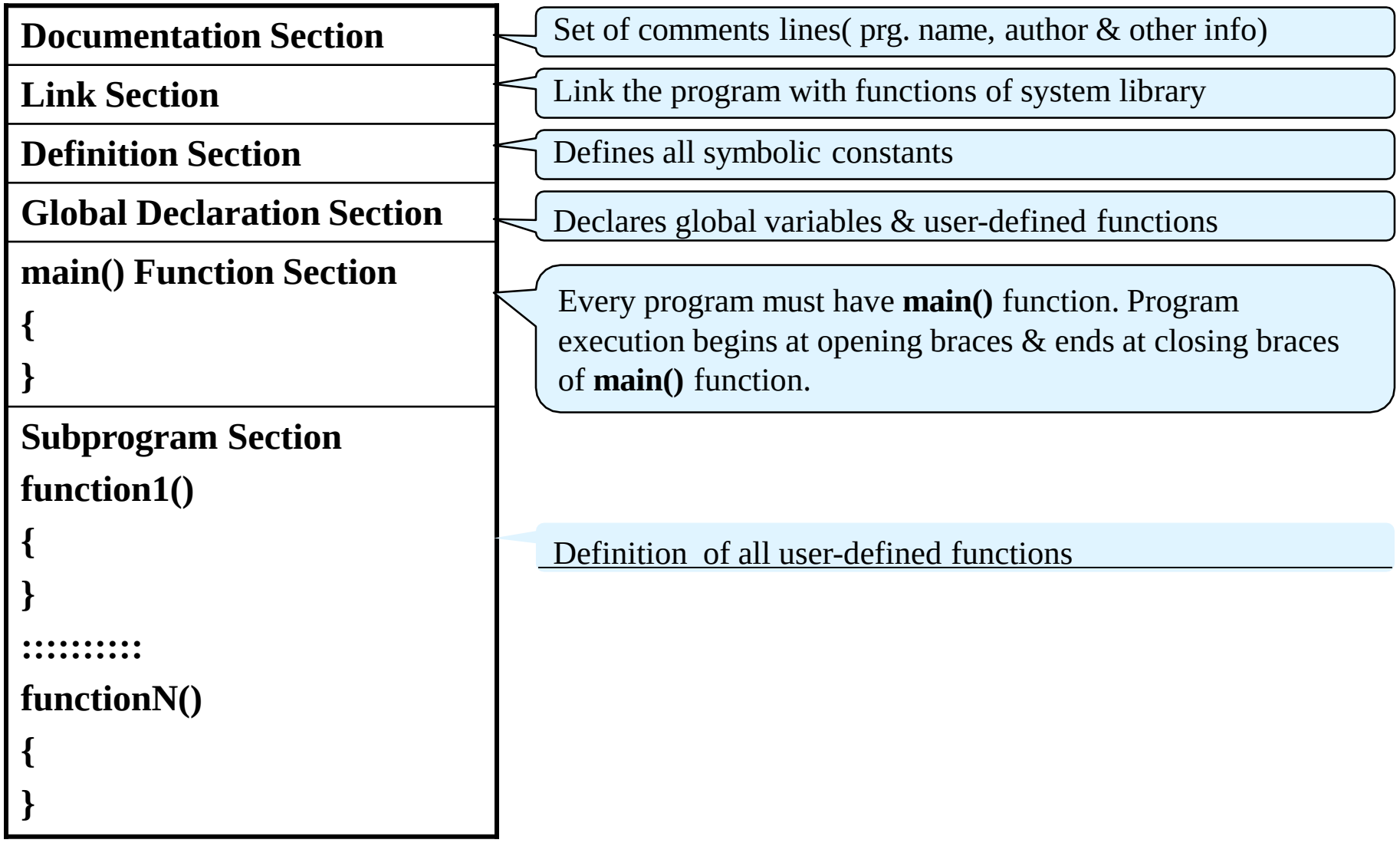
- C is *small* (only 32 keywords).
- C has rich set of *built-in functions* and support variety of *data types* & *operators*.
- C is *highly portable* (Machine independent).
- C is *structured*.
- C has *ability to extend* itself.
- C is *stable* (the language doesn't change much).
- C is *quick running* (code written in c is efficient & fast).
- C is the *basis for many other languages* (C++, C#, Java, Perl etc).
- C is a *Programmers Language*.

➢ ***It may not feel like it but C is one of the easiest language to learn.***

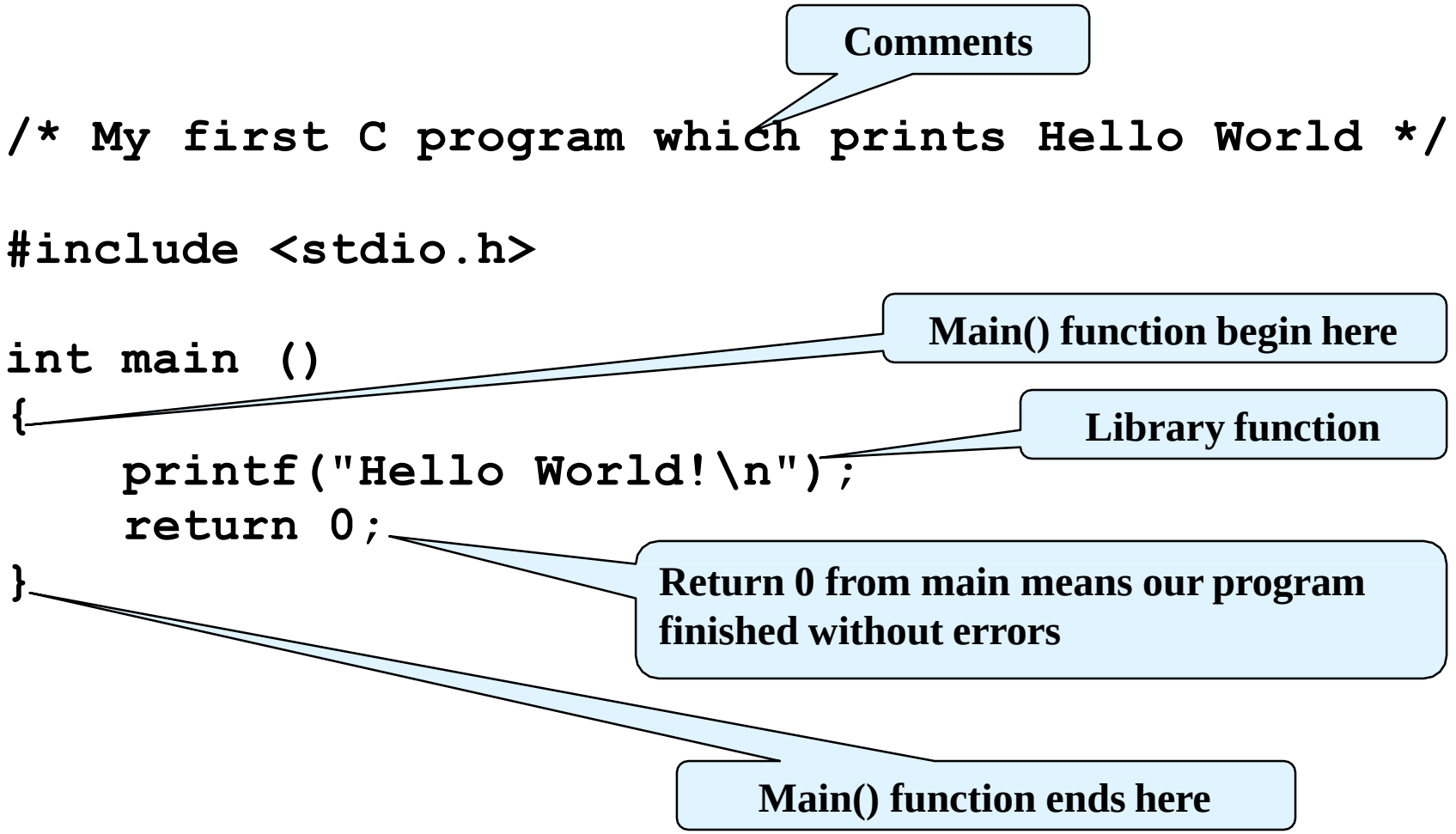
# What is C used for?

- **Systems programming:**
  - OSes, like Linux, Windows, Mac
  - Microcontrollers: automobiles and airplanes
  - Embedded processors: phones, portable electronics, etc.
  - DSP processors: digital audio and TV systems.

# Basic Structure of C Program



# Hello World Program



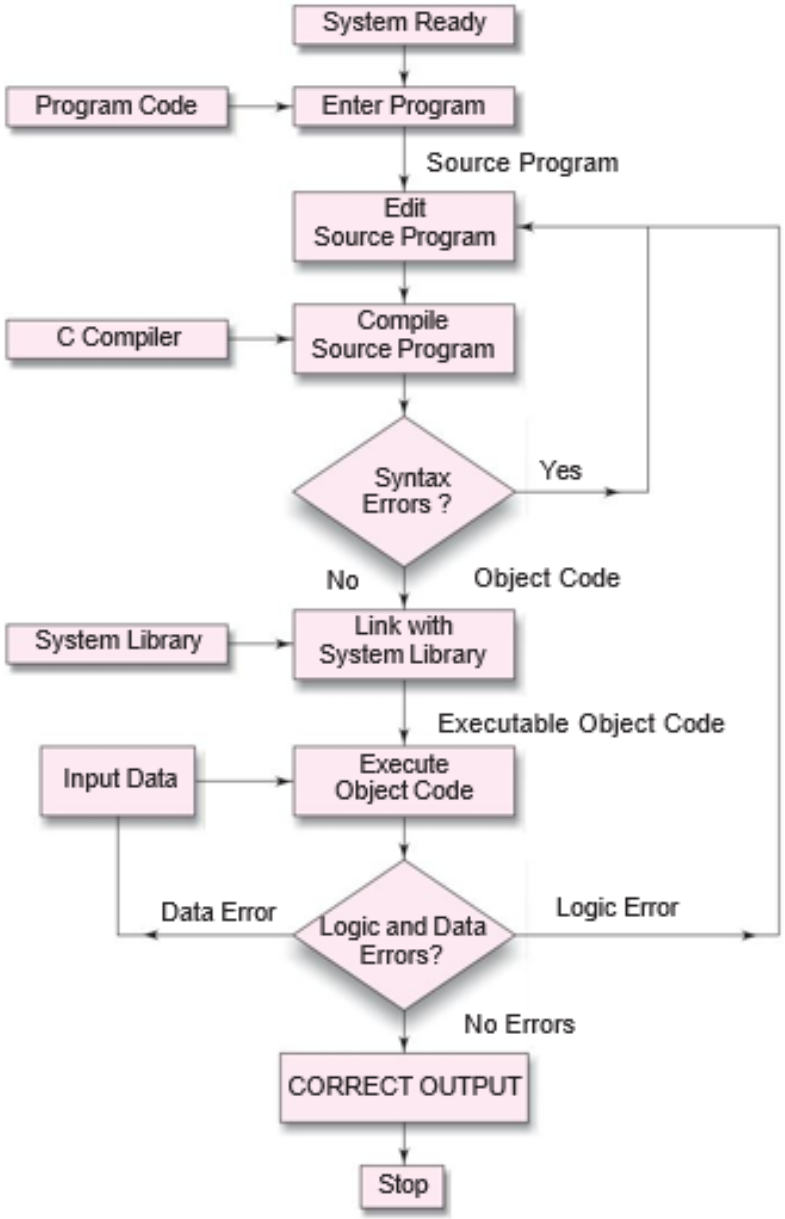



Fig. 1.10 *Process of compiling and running a C program*

## Just Remember

- Every C program requires a **main()** function (Use of more than one **main()** is illegal). The place **main** is where the program execution begins.
- The execution of a function begins at the opening brace of the function and ends at the corresponding closing brace.
- C programs are written in lowercase letters. However, uppercase letters are used for symbolic names and output strings.
- All the words in a program line must be separated from each other by at least one space, or a tab, or a punctuation mark.
- Every program statement in a C language must end with a semicolon.
- All variables must be declared for their types before they are used in the program.
- We must make sure to include header files using **#include** directive when the program refers to special names and functions that it does not define.
- Compiler directives such as **define** and **include** are special instructions to the compiler to help it compile a program. They do not end with a semicolon.
- The sign # of compiler directives must appear in the first column of the line.
- When braces are used to group statements, make sure that the opening brace has a corresponding closing brace.
- C is a free-form language and therefore a proper form of indentation of various sections would improve legibility of the program.
- A comment can be inserted almost anywhere a space can appear. Use of appropriate comments in proper places increases readability and understandability of the program and helps users in debugging and testing. Remember to match the symbols /* and * appropriately.

# Any Question?

# Thank You All