Chapter 1 – Introduction to Computers and C++ Programming

Outline

1.1 Introduction
1.2 What Is a Computer?
1.3 Computer Organization
1.4 Evolution of Operating Systems
1.5 Personal Computing, Distributed Computing and Client/Server Computing
1.6 Machine Languages, Assembly Languages and High-level Languages
1.7 The History of C
1.8 The C Standard Library
1.9 The Key Software Trend: Object Technology
1.10 C++ and C++ How to Program
1.11 Java and Java How to Program
1.12 Other High-level Languages
1.13 Structured Programming
1.14 The Basics of a typical C Program Development Environment
Chapter 1 – Introduction to Computers
and C++ Programming

Outline
1.15 Hardware Trends
1.16 History of the Internet
1.17 History of the World Wide Web
1.18 General Notes About C and this Book
Objectives

• In this chapter, you will learn:
  – To understand basic computer concepts.
  – To become familiar with different types of programming languages.
  – To become familiar with the history of the C programming language.
  – To become aware of the C standard library.
  – To understand the elements of a typical C program development environment.
  – To appreciate why it is important to learn C in a first programming course.
  – To appreciate why C provides a foundation for further study of programming languages in general.
1.1 Introduction

• We will learn
  – The C programming language
  – Structured programming and proper programming techniques

• This course is appropriate for
  – Technically oriented people with little or no programming experience
  – Experienced programmers who want a deep and rigorous treatment of the language
1.2 What is a Computer?

• Computer
  – Device capable of performing computations and making logical decisions
  – Computers process data under the control of sets of instructions called computer programs

• Hardware
  – Various devices comprising a computer
  – Keyboard, screen, mouse, disks, memory, CD-ROM, and processing units

• Software
  – Programs that run on a computer
1.3 Computer Organization

- Six logical units in every computer:
  1. Input unit
     - Obtains information from input devices (keyboard, mouse)
  2. Output unit
     - Outputs information (to screen, to printer, to control other devices)
  3. Memory unit
     - Rapid access, low capacity, stores input information
  4. Arithmetic and logic unit (ALU)
     - Performs arithmetic calculations and logic decisions
  5. Central processing unit (CPU)
     - Supervises and coordinates the other sections of the computer
  6. Secondary storage unit
     - Cheap, long-term, high-capacity storage
     - Stores inactive programs

1.4 Evolution of Operating Systems

- Batch processing
  - Do only one job or task at a time
- Operating systems
  - Manage transitions between jobs
  - Increased throughput
    - Amount of work computers process
- Multiprogramming
  - Computer resources are shared by many jobs or tasks
- Timesharing
  - Computer runs a small portion of one user’s job then moves on to service the next user
1.5 Personal Computing, Distributed Computing, and Client/Server Computing

• Personal computers
  – Economical enough for individual

• Distributed computing
  – Computing distributed over networks

• Client/server computing
  – Sharing of information across computer networks between file servers and clients (personal computers)
1.6 Machine Languages, Assembly Languages, and High-level Languages

Three types of programming languages

1. Machine languages
   - Strings of numbers giving machine specific instructions
   - Example:
     \[ +1300042774 \]
     \[ +1400593419 \]
     \[ +1200274027 \]

2. Assembly languages
   - English-like abbreviations representing elementary computer operations (translated via assemblers)
   - Example:
     \[ \text{LOAD} \quad \text{BASEPAY} \]
     \[ \text{ADD} \quad \text{OVERPAY} \]
     \[ \text{STORE} \quad \text{GROSSPAY} \]
1.6 Machine Languages, Assembly Languages, and High-level Languages

Three types of programming languages (continued)

3. High-level languages
   - Codes similar to everyday English
   - Use mathematical notations (translated via compilers)
   - Example:
     \[ \text{grossPay} = \text{basePay} + \text{overTimePay} \]
1.7 History of C

• C
  – Evolved by Ritchie from two previous programming languages, BCPL and B
  – Used to develop UNIX
  – Used to write modern operating systems
  – Hardware independent (portable)
  – By late 1970's C had evolved to "Traditional C"

• Standardization
  – Many slight variations of C existed, and were incompatible
  – Committee formed to create a "unambiguous, machine-independent" definition
  – Standard created in 1989, updated in 1999
1.8 The C Standard Library

• C programs consist of pieces/modules called functions
  – A programmer can create his own functions
    • Advantage: the programmer knows exactly how it works
    • Disadvantage: time consuming
  – Programmers will often use the C library functions
    • Use these as building blocks
  – Avoid re-inventing the wheel
    • If a premade function exists, generally best to use it rather than write your own
    • Library functions carefully written, efficient, and portable
1.9 The Key Software Trend: Object Technology

• Objects
  – Reusable software components that model items in the real world
  – Meaningful software units
    • Date objects, time objects, paycheck objects, invoice objects, audio objects, video objects, file objects, record objects, etc.
    • Any noun can be represented as an object
  – Very reusable
  – More understandable, better organized, and easier to maintain than procedural programming
  – Favor modularity
1.12 Other High-level Languages

• Other high-level languages
  – FORTRAN
    • Used for scientific and engineering applications
  – COBOL
    • Used to manipulate large amounts of data
  – Pascal
    • Intended for academic use
1.13 Structured Programming

• Structured programming
  – Disciplined approach to writing programs
  – Clear, easy to test and debug and easy to modify

• Multitasking
  – Specifying that many activities run in parallel
1.14 Basics of a Typical C Program Development Environment

- Phases of C++ Programs:

1. **Edit**
   - Program is created in the editor and stored on disk.

2. **Preprocess**
   - Preprocessor program processes the code.

3. **Compile**
   - Compiler creates object code and stores it on disk.

4. **Link**
   - Linker links the object code with the libraries.

5. **Load**
   - Loader puts program in memory.

6. **Execute**
   - CPU takes each instruction and executes it, possibly storing new data values as the program executes.
1.15 Hardware Trends

• Every year or two the following approximately double:
  – Amount of memory in which to execute programs
  – Amount of secondary storage (such as disk storage)
    • Used to hold programs and data over the longer term
  – Processor speeds
    • The speeds at which computers execute their programs
1.16 History of the Internet

• The Internet enables
  – Quick and easy communication via e-mail
  – International networking of computers

• Packet switching
  – The transfer of digital data via small packets
  – Allows multiple users to send and receive data simultaneously

• No centralized control
  – If one part of the Internet fails, other parts can still operate

• TCP/IP

• Bandwidth
  – Information carrying capacity of communications lines
1.17 History of the World Wide Web

- **World Wide Web**
  - Locate and view multimedia-based documents on almost any subject
  - Makes information instantly and conveniently accessible worldwide
  - Possible for individuals and small businesses to get worldwide exposure
  - Changing the way business is done
1.18 General Notes About C and This Book

- **Program clarity**
  - Programs that are convoluted are difficult to read, understand, and modify

- **C is a portable language**
  - Programs can run on many different computers
  - However, portability is an elusive goal

- **We will do a careful walkthrough of C**
  - Some details and subtleties are not covered
  - If you need additional technical details
    - Read the C standard document
    - Read the book by Kernigan and Ritchie