Chapter 1

Introduction to Computers and C++ Programming
Overview

1.1  Computer Systems
1.2  Programming and Problem Solving
1.3  Introduction to C++
1.4  Testing and Debugging
1.1

Computer Systems
A computer program is...
- A set of instructions for a computer to follow

Computer software is...
- The collection of programs used by a computer
  - Includes:
    - Editors
    - Translators
    - System Managers
Hardware

- Three main classes of computers
  - PCs (Personal Computer)
    - Relatively small used by one person at a time
  - Workstation
    - Larger and more powerful than a PC
  - Mainframe
    - Still larger
    - Requires support staff
    - Shared by multiple users
Networks

- A number of computers connected to share resources
  - Share printers and other devices
  - Share information
Computer Organization

- Five main components
  - Input devices
    - Allows communication to the computer
  - Output devices
    - Allows communication to the user
  - Processor (CPU)
  - Main memory
    - Memory locations containing the running program
  - Secondary memory
    - Permanent record of data often on a disk
Computer Memory

- Main Memory
  - Long list of memory locations
    - Each contains zeros and ones
    - Can change during program execution
  - Binary Digit or Bit
    - A digit that can only be zero or one
  - Byte
    - Each memory location has eight bits
  - Address
    - Number that identifies a memory location
Larger Data Items

• Some data is too large for a single byte
  • Most integers and real numbers are too large

• Address refers to the first byte

• Next few consecutive bytes can store the additional bits for larger data

Display 1.2
Data or Code?

- ‘A’ may look like 01000001
- 65 may look like 01000001
- An instruction may look like 01000001

- How does the computer know the meaning of 01000001?
  - Interpretation depends on the current instruction

- Programmers rarely need to be concerned with this problem.
  - Reason as if memory locations contain letters and numbers rather than zeroes and ones
Secondary Memory

- Main memory stores instructions and data while a program is running.
- Secondary memory
  - Stores instructions and data between sessions
  - A file stores data or instructions in secondary memory
Secondary Memory Media

- A computer might have any of these types of secondary memory
  - Hard disk
    - Fast
    - Fixed in the computer and not normally removed
  - Floppy disk
    - Slow
    - Easily shared with other computers
  - Compact disk
    - Slower than hard disks
    - Easily shared with other computers
    - Can be read only or re-writable
Memory Access

- Random Access
  - Usually called RAM
    - Computer can directly access any memory location

- Sequential Access
  - Data is generally found by searching through other items first
    - More common in secondary memory
The Processor

- Typically called the CPU
  - Central Processing Unit
  - Follows program instructions
  - Typical capabilities of CPU include:
    
    add
    subtract
    multiply
    divide
    move data from location to location
Computer Software

- The operating system
  - Allows us to communicate with the computer
  - Is a program
  - Allocates the computer’s resources
  - Responds to user requests to run other programs

- Common operating systems include…
  - UNIX
  - Linux
  - DOS
  - Windows
  - Macintosh
  - VMS
Computer Input

- Computer input consists of
  - A program
  - Some data

Display 1.3
High-level Languages

- Common programming languages include ...

  C    C++    Java    Pascal    Visual Basic    FORTRAN    Perl    PHP    Lisp    Scheme    Ada    C#    Python

- These high – level languages
  - Resemble human languages
  - Are designed to be easy to read and write
  - Use more complicated instructions than the CPU can follow
  - Must be translated to zeros and ones for the CPU to execute a program
Low-level Languages

- An assembly language command such as
  
  ```
  ADD X Y Z
  ```
  
might mean add the values found at x and y in memory, and store the result in location z.

- Assembly language must be translated to machine language (zeros and ones)
  
  0110 1001 1010 1011

- The CPU can follow machine language
Compilers

- Translate high-level language to machine language
  - Source code
    - The original program in a high level language
  - Object code
    - The translated version in machine language
Linkers

- Some programs we use are already compiled
  - Their object code is available for us to use
  - For example: Input and output routines

- A Linker combines
  - The object code for the programs we write and
  - The object code for the pre-compiled routines
  - The machine language program the CPU can run

Display 1.5
History Note

- First programmable computer
  - Designed by Charles Babbage
  - Began work in 1822
  - Not completed in Babbage’s life time

- First programmer
  - Ada Augusta, Countess of Lovelace
    - Colleague of Babbage
Section 1.1 Conclusion

- Can you…
  - List the five main components of a computer?
  - List the data for a program that adds two numbers?
  - Describe the work of a compiler?
  - Define source code? Define object code?
  - Describe the purpose of the operating system?
1.2

Programming and Problem-Solving
Algorithms

- **Algorithm**
  - A sequence of precise instructions that leads to a solution

- **Program**
  - An algorithm expressed in a language the computer can understand
Program Design

- Programming is a creative process
  - No complete set of rules for creating a program

- Program Design Process
  - Problem Solving Phase
    - Result is an algorithm that solves the problem
  - Implementation Phase
    - Result is the algorithm translated into a programming language
Problem Solving Phase

- Be certain the task is completely specified
  - What is the input?
  - What information is in the output?
  - How is the output organized?

- Develop the algorithm before implementation
  - Experience shows this saves time in getting your program to run.
  - Test the algorithm for correctness
Implementation Phase

- Translate the algorithm into a programming language
  - Easier as you gain experience with the language

- Compile the source code
  - Locates errors in using the programming language

- Run the program on sample data
  - Verify correctness of results

- Results may require modification of the algorithm and program
Object Oriented Programming

- Abbreviated OOP
- Used for many modern programs
- Program is viewed as interacting objects
  - Each object contains algorithms to describe its behavior
  - Program design phase involves designing objects and their algorithms
OOP Characteristics

- Encapsulation
  - Information hiding
  - Objects contain their own data and algorithms

- Inheritance
  - Writing reusable code
  - Objects can inherit characteristics from other objects

- Polymorphism
  - A single name can have multiple meanings depending on its context
Software Life Cycle

- Analysis and specification of the task (problem definition)
- Design of the software (object and algorithm design)
- Implementation (coding)
- Maintenance and evolution of the system
- Obsolescence
Can you…

- Describe the first step to take when creating a program?
- List the two main phases of the program design process?
- Explain the importance of the problem-solving phase?
- List the steps in the software life cycle?
Introduction to C++
Introduction to C++

- Where did C++ come from?
  - Derived from the C language
  - C was derived from the B language
  - B was derived from the BCPL language

- Why the ‘++’?
  - ++ is an operator in C++ and results in a cute pun
C++ History

- C developed by Dennis Ritchie at AT&T Bell Labs in the 1970s.
  - Used to maintain UNIX systems
  - Many commercial applications written in c

- C++ developed by Bjarne Stroustrup at AT&T Bell Labs in the 1980s.
  - Overcame several shortcomings of C
  - Incorporated object oriented programming
  - C remains a subset of C++
A Sample C++ Program

- A simple C++ program begins this way

```cpp
#include <iostream>
using namespace std;

int main()
{

}
```

- And ends this way

```cpp
    return 0;
}
```

Display 1.8
Explanation of code (1/5)

- Variable declaration line

```java
int number_of_pods, peas_per_pod, total_peas;
```

- Identifies names of three variables to name numbers
- `int` means that the variables represent integers
Explanation of code (2/5)

Program statement

cout << “Press return after entering a number.\n”;

- cout (see-out) used for output to the monitor

- “<<” inserts “Press…a number.\n” in the data bound for the monitor

- Think of cout as a name for the monitor
  - “<<” points to where the data is to end up

- ‘\n’ causes a new line to be started on the monitor
Explanation of code (3/5)

- Program statement

```cpp
    cin >> number_of_pods;
```

- `cin (see-in)` used for input from the keyboard

- “`>>`” extracts data from the keyboard

- Think of `cin` as a name for the keyboard
  - “`>>`” points from the keyboard to a variable where the data is stored
Explanation of code (4/5)

- Program statement

  total_peas = number_of_pods * peas_per_pod;

- Performs a computation
- ‘*’ is used for multiplication
- ‘=‘ causes total_peas to get a new value based on the calculation shown on the right of the equal sign
Explanation of code (5/5)

- Program statement

  ```
  cout << number_of_pods;
  ```

- Sends the value of variable `number_of_pods` to the monitor
Program Layout (1/3)

- Compiler accepts almost any pattern of line breaks and indentation
- Programmers format programs so they are easy to read
  - Place opening brace '{' and closing brace '}' on a line by themselves
  - Indent statements
  - Use only one statement per line
Variables are declared before they are used
- Typically variables are declared at the beginning of the program
- Statements (not always lines) end with a semi-colon

Include Directives
#include <iostream>
- Tells compiler where to find information about items used in the program
- iostream is a library containing definitions of cin and cout
using namespace std;

Tells the compiler to use names in iostream in a “standard” way

To begin the main function of the program

int main()
{

To end the main function

return 0;
}

Main function ends with a return statement
Running a C++ Program

- C++ source code is written with a text editor
- The compiler on your system converts source code to object code.
- The linker combines all the object code into an executable program.
C++11

- C++11 (formerly known as C++0x) is the most recent version of the standard of the C++ programming language.
  - Approved on August 12, 2011 by the International Organization for Standardization.
- C++11 language features are not supported by older compilers
- Check the documentation with your compiler to determine if special steps are needed to compile C++11 programs
  - e.g. with g++, use extra flags of –std=c++11
Run a Program

- Obtain code in Display 1.10
- Compile the code
- Fix any errors the compiler indicates and re-compile the code
- Run the program
- Now you know how to run a program on your system
Section 1.3 Conclusion

- Can you…
  - Describe the output of this line?
    ```
    cout << “C++ is easy to understand.”;
    ```
  - Explain what this line does?
    ```
    cin >> peas_per_pod;
    ```
  - Explain this?
    ```
    #include <iostream>
    ```
1.4 Testing and Debugging
Testing and Debugging

- Bug
  - A mistake in a program
- Debugging
  - Eliminating mistakes in programs
  - Term used when a moth caused a failed relay on the Harvard Mark 1 computer. Grace Hopper and other programmers taped the moth in logbook stating:
    “First actual case of a bug being found.”
Program Errors

- Syntax errors
  -Violation of the grammar rules of the language
  -Discovered by the compiler
    -Error messages may not always show correct location of errors
- Run-time errors
  -Error conditions detected by the computer at run-time
- Logic errors
  -Errors in the program’s algorithm
  -Most difficult to diagnose
  -Computer does not recognize an error
Section 1-4 Conclusion

Can you…
- Describe the three kinds of program errors?
- Tell what kind of errors the compiler catches?
- What kind of error is produced if you forget a punctuation symbol such as a semi-colon?
- Tell what type of error is produced when a program runs but produces incorrect results?
Chapter 1 -- End
Main Components of a Computer

- Processor (CPU)
- Input device(s)
- Main memory
- Secondary memory
- Output device(s)
**Memory Locations and Bytes**

- byte 1
- byte 2
- byte 3
- byte 4
- byte 5
- byte 6
- byte 7
- byte 8
- byte 9

- 3 byte location with address 1
- 2 byte location with address 4
- 1 byte location with address 6
- 3 byte location with address 7
Display 1.3

Simple View of Running a Program

Program

Data

Computer

Output
Display 1.4

Compiling and Running a C++ Program
(Basic Outline)
Display 1.5

Preparing a C++ Program for Running

1. C++ program
2. Compiler
3. Object code for C++ program
4. Object code for other routines
5. Linker
6. Complete machine language code ready to run
Display 1.6

An Algorithm

Algorithm that determines how many times a name occurs in a list of names:

1. Get the list of names.
2. Get the name being checked.
3. Set a counter to zero.
4. Do the following for each name on the list:
   Compare the name on the list to the name being checked, and if the names are the same, then add one to the counter.
5. Announce that the answer is the number indicated by the counter.
Display 1.7

Program Design Process

Problem-solving phase

Start

Problem definition

Algorithm design

Desktop testing

Implementation phase

Translating to C++

Testing

Working program
#include <iostream>
using namespace std;

int main()
{
    int number_of_pods, peas_per_pod, total_peas;

    cout << "Press return after entering a number.\n";
    cout << "Enter the number of pods: \n";
    cin >> number_of_pods;
    cout << "Enter the number of peas in a pod: \n";
    cin >> peas_per_pod;

    total_peas = number_of_pods * peas_per_pod;

    cout << "If you have \n";
    cout << number_of_pods;
    cout << " pea pods\n";
    cout << " and \n";
    cout << peas_per_pod;
    cout << " peas in each pod, then\n";
    cout << " you have \n";
    cout << total_peas;
    cout << " peas in all the pods.\n";

    return 0;
}

Sample Dialogue

Press return after entering a number.
Enter the number of pods:
10
Enter the number of peas in a pod:
9
If you have 10 pea pods
and 9 peas in each pod, then
you have 90 peas in all the pods.
Layout of a Simple C++ Program

#include <iostream>
using namespace std;

int main()
{
    Variable_Declarations

    Statement_1
    Statement_2
    ...
    Statement_Last

    return 0;
}
Testing Your C++ Setup

```cpp
#include <iostream>
using namespace std;

int main()
{
    cout << "Testing 1, 2, 3\n";
    return 0;
}
```

If you cannot compile and run this program, then see the programming tip entitled “Getting Your Program to Run.” It suggests some things you might do to get your C++ programs to run on your particular computer setup.

Sample Dialogue

Testing 1, 2, 3