Chapter 1: An Introduction to Computer Science

Invitation to Computer Science, C++ Version, 6-th Edition
Objectives

In this chapter, you will learn about

- The definition of computer science
- Algorithms
- A brief history of computing
- Organization of the text
Introduction

- What is computer science?
  - Computer science is the study of computers?
  - Computer science is the study of how to write computer programs?
  - Computer science is the study of the uses and applications of computers and software?
Common Misconceptions about Computer Science

- **Computer science is the study of computers**
  - Incomplete – Theoretical work began (1920-1940) before computers
  - CS became an independent field of study late 1950’s, early 1960’s
  - Theoretical CS relies on formal models rather than “real” machines

- CS is no more about computers than astronomy is about telescopes. (see quote on p.3)
Common Misconceptions about Computer Science

- **Computer science is the study of how to write computer programs**
  - Programming is important, but it is just a tool for studying new ideas, representing information or testing the solution to a problem.
  - A Program is a **means to an end, not the end itself**.
  - Eg. Searching a list such as the NYC phone Directory
Common Misconceptions about Computer Science

Computer science is the study of the uses and applications of computers and software

- Examples of popular applications are word processors, databases, image software, presentation software, electronic email, Web browser, etc.
- Many people use software, but the computer scientist is responsible for specifying, designing, building and testing software packages and the systems on which they run.
The Definition of Computer Science

- Computer scientist designs and develops algorithms to solve problems

- Gibbs and Tucker definition of computer science
  - The study of algorithms, including
    - Their formal and mathematical properties
    - Their Hardware realizations
    - Their Linguistic realizations
    - Their Applications
The Definition of Computer Science (con't)

- Algorithm
  - Dictionary definition
    - A procedure for solving a mathematical problem in a finite number of steps that frequently involves repetition of an operation
    - A step-by-step method for accomplishing a task
  - Informal description
    - An ordered sequence of instructions that is guaranteed to solve a specific problem
The Definition of Computer Science (con't)

- Operations involved in designing algorithms
  - Formal and mathematical properties
    - Studying the behavior of algorithms to determine whether they are correct and efficient
  - Hardware realizations
    - Designing and building computer systems that are able to execute algorithms
The Definition of Computer Science (con't)

- **Linguistic realizations**
  - Designing programming languages and translating algorithms into these languages

- **Applications**
  - Identifying important problems and designing correct and efficient software packages to solve these problems
What an Algorithm Looks like?

An algorithm is a list that looks like

- **STEP 1**: Do something.
- **STEP 2**: Do something.
- **STEP 3**: Do something.
- ...
- ...
- ...
- **STEP N**: Stop. You are finished.
Algorithms

- We use algorithms all the time
  - Examples?
    - Following directions
    - Programming a DVD
    - Adding two numbers
    - Finding Greatest Common Divisor
Example of Algorithms

**Algorithm for Programming Your DVR**

**Step 1**  If the clock and calendar are not correctly set, then go to page 9 of the instruction manual and follow the instructions there before proceeding to step 2.

**Step 2**  Place a blank tape into the DVR disc slot.

**Step 3**  Repeat steps 4 through 7 for each program that you wish to record.

**Step 4**  Enter the channel number that you wish to record and press the button labeled CHAN.

**Step 5**  Enter the time that you wish recording to start and press the button labeled TIME-START.

**Step 6**  Enter the time that you wish recording to stop and press the button labeled TIME-FINISH. This completes the programming of one show.

**Step 7**  If you do not wish to record anything else, press the button labeled END-PROG.

**Step 8**  Turn off your DVR. Your DVR is now in TIMER mode, ready to record.
Example of Algorithms (con’t)

Algorithm for Adding Two m-Digit Numbers

Given: \( m \geq 1 \) and two positive numbers each containing \( m \) digits, \( a_{m-1} a_{m-2} \ldots a_0 \) and \( b_{m-1} b_{m-2} \ldots b_0 \)

Wanted: \( c_m c_{m-1} c_{m-2} \ldots c_0 \), where
\[
(c_m c_{m-1} c_{m-2} \ldots c_0) = (a_{m-1} a_{m-2} \ldots a_0) + (b_{m-1} b_{m-2} \ldots b_0)
\]

Algorithm:

Step 1  Set the value of \textit{carry} to 0.
Step 2  Set the value of \( i \) to 0.
Step 3  While the value of \( i \) is less than or equal to \( m - 1 \), repeat the instructions in steps 4 through 6.
Step 4  Add the two digits \( a_i \) and \( b_i \) to the current value of \textit{carry} to get \( c_i \).
Step 5  If \( c_i \geq 10 \), then reset \( c_i \) to \((c_i - 10)\) and reset the value of \textit{carry} to 1; otherwise, set the new value of \textit{carry} to 0.
Step 6  Add 1 to \( i \), effectively moving one column to the left.
Step 7  Set \( c_m \) to the value of \textit{carry}.
Step 8  Print out the final answer, \( c_m c_{m-1} c_{m-2} \ldots c_0 \).
Step 9  Stop.
The Formal Definition of an Algorithm

- Algorithm
  - A well-ordered collection of unambiguous and effectively computable operations that, when executed, produces a result and halts in a finite amount of time
The Formal Definition of an Algorithm (con’t)

- **Unambiguous operation**
  - An operation that can be understood and carried out directly by the computing agent without needing to be further simplified or explained
  - Also called a *primitive* operation
  - Examples of ambiguous statements
    - Go back and do it again (Do *what* again?)
    - Start over (From *where*?)
Ambiguous vs. Unambiguous

Step 1: Wet hair
Step 2: Lather
Step 3: Rinse
Step 4: Repeat

Ambiguous

Step 1: Wet hair
Step 2: Set WashCount = 0
Step 3: Repeat 4 to 6 until WashCount = 2
Step 4: Lather hair
Step 5: Rinse hair
Step 6: Add 1 to WashCount
Step 7: Stop

Unambiguous
High Level vs. Further Simplified

**High Level**
- Step 1: Make the crust
- Step 2: Make The cherry filling
- Step 3: Pour the filling into the crust
- Step 4: Bake at 350°F for 45 minutes

**Further Simplified and**
- Step 1: Make the crust
  - 1.1 Take one and on-third cups flour
  - Sift the flour
  - Mix the sifted flour with one-half cup butter and on-fourth cut water
  - Roll into two 9-inch pie crusts
- Step 2: Make The cherry filling
- ...

Invitation to Computer Science, C++ Version, 6E
The Formal Definition of an Algorithm (con't)

- **A primitive operation** (or a primitive) of the computing agent
  - Operation that is unambiguous for computing agent
  - Primitive operations of different individuals (or machines) vary
  - An algorithm must be composed entirely of primitives

- **Effectively computable**
  - Computational process exists that allows computing agent to complete that operation successfully
The Formal Definition of an Algorithm (con't)

- The result of the algorithm must be produced after the execution of a finite number of operations
  - Infinite loop
    - The algorithm has no provisions to terminate
    - A common error in the designing of algorithms
      Step1: Wet hair
      Step2: Lather
      Step3: Rinse
      Step4: Repeat 3-4
Categories of Algorithm Operations

All the operations used to construct algorithms belong to one of only three categories:

- Sequential Operations
- Conditional Operations
- Iterative Operations
Categories of Algorithm Operations

- **Sequential operations**
  - Carry out a single well-defined task; when that task is finished, the algorithm moves on to the next operation
  - Examples:
    - Add 1 cup of butter to the mixture in the bowl
    - Subtract the amount of the check from the current account balance
    - Set the value of $x$ to 1
    - Set the value of $y$ to $x^2 \sin(1/2)$
Categories of Algorithm Operations (con't)

- **Conditional operations**
  - Ask a question and then select the next operation to be executed on the basis of the answer to that question
  - **Examples**
    - If the mixture is too dry, then add one-half cup of water to the bowl
Categories of Algorithm Operations (con't)

- More examples about conditional operations:
  - If the amount of the check is less than or equal to the current account balance, then cash the check; otherwise, tell the person that the account is overdrawn
  - If x is not equal to 0, then set y equal to 1/x; otherwise, print an error message that says we cannot divide by 0
Categories of Algorithm Operations (con't)

- **Iterative operations**
  - Tell us to go back and repeat the execution of a previous block of instructions
  - **Examples**
    - Repeat the previous two operations until the mixture has thickened
    - While there are still more checks to be processed, do the following five steps
    - Repeat steps 1, 2, and 3 until the value of y is equal to 11
Algorithms

Why an algorithm is so important?

- If we can specify an algorithm to solve a problem, we can automate its solution by a computing agent.

Computing agent

- The machine, robot, person, or thing carrying out the steps of the algorithm.
- Does not need to understand the concepts or ideas underlying the solution.
The Importance of Algorithmic Problem Solving

- “Computer revolution” of the twentieth and twenty-first centuries
  - Has enabled us to implement algorithms that **mechanize and automate** the drudgery of repetitive mental tasks

Algorithmic solutions can be
- Encoded into some appropriate programming language
- Given to a computing agent to execute

The computing agent
- Would mechanically follow these instructions and successfully complete the task specified
- Would not have to understand
  - Creative processes that went into discovery of solution
  - Principles and concepts that underlie the problem
Can all problems be solved algorithmically?

NO!

- There are problems which have **no generalized solution** – unsolvable or intractable
- Some with an algorithm would take **so long to execute** that the algorithm is useless
- Some problems we **have not yet** discovered an algorithm for

- Kurt Godel (1930’s) - See Chapter 12
A Brief History of Computing
The Early Period: Up to 1940

- 3,000 years ago: Mathematics, logic, and numerical computation
  - Important contributions made by the Greeks, Egyptians, Babylonians, Indians, Chinese, and Persians

- 1614: Logarithms
  - Invented by John Napier to simplify difficult mathematical computations

- Around 1622: First slide rule created
The Early Period: Up to 1940 (con't)

- 1672: The Pascaline
  - Designed and built by Blaise Pascal
  - One of the first mechanical calculators
  - Could do addition and subtraction
The Early Period: Up to 1940 (con't)

- 1674: Leibnitz’s Wheel
  - Constructed by Gottfried Leibnitz
  - Mechanical calculator
  - Could do addition, subtraction, multiplication, and division

The Leibnitz’s Wheel
The Early Period: Up to 1940 (con't)

- 1801: The Jacquard loom
  - Developed by Joseph Jacquard
  - Automated loom
  - Used punched cards to create desired pattern

- 1823: The Difference Engine
  - Developed by Charles Babbage
  - Did addition, subtraction, multiplication, and division to 6 significant digits
  - Solved polynomial equations and other complex mathematical problems
Figure 1.5
Drawing of the Jacquard Loom
The Early Period: Up to 1940 (con't)

- 1830s: The Analytic Engine
  - Designed by Charles Babbage
  - More powerful and general-purpose computational machine
  - Components were functionally similar to the four major components of today’s computers
    - Mill (modern terminology: arithmetic/logic unit)
    - Store (modern terminology: memory)
    - Operator (modern terminology: processor)
    - Output (modern terminology: input/output)
Difference engine

http://www.youtube.com/watch?v=0anlyVGeWOI
The Early Period: Up to 1940 (con't)

- 1890: U.S. census carried out with programmable card processing machines
  - Built by Herman Hollerith
  - These machines could automatically read, tally, and sort data entered on punched cards
The Early Period: Up to 1940

- First Programmer
- Ada Augusta Byron, Countess of Lovelace
The Birth of Computers: 1940-1950

- Development of electronic, general-purpose computers
  - Did not begin until after 1940
  - Was fueled in large part by needs of World War II

- Early computers
  - Mark I
  - ENIAC
  - ABC system
  - Colossus
  - Z1
Figure 1.6
Photograph of the ENIAC Computer
Replacing a bad tube meant checking among ENIAC’s 19,000 possibilities.

Replacing a vacuum tube in the ENIAC
Programming the ENIAC
….and still programming
The Birth of Computers: 1940-1950 (con't)

- Stored program computer model
  - Proposed by John Von Neumann in 1946
  - Stored binary algorithm in the computer’s memory along with the data
  - Is known as the Von Neumann architecture
  - Modern computers remain, fundamentally, Von Neumann machines
- First stored program computers
  - EDVAC
  - EDSAC
The Modern Era: 1950 to the Present

- First generation of computing (1950-1959)
  - Vacuum tubes used to store data and programs
  - Each computer was multiple rooms in size
  - Computers were not very reliable
The Modern Era: 1950 to the Present (con't)

- Second generation of computing (1959-1965)
  - Transistors and magnetic cores replaced vacuum tubes
  - Dramatic reduction in size
    - Computer could fit into a single room
  - Increase in reliability of computers
  - Reduced cost of computers
  - High-level programming languages
    - The programmer occupation was born
The Modern Era: 1950 to the Present (con't)

- Third generation of computing (1965-1975)
  - Integrated circuits rather than individual electronic components were used
  - Further reduction in size and cost of computers
    - Computers became desk-sized
    - First minicomputer developed
  - Software industry formed
The Modern Era: 1950 to the Present (con't)

- Fourth generation of computing (1975-1985)
  - Reduced to the size of a typewriter
  - First microcomputer developed
  - Desktop and personal computers common
  - Appearance of
    - Computer networks
    - Electronic mail
    - User-friendly systems (graphical user interfaces)
    - Embedded systems
Figure 1.7
The Altair 8800, the World’s First Microcomputer
Beginnings of the Apple Computer

http://www.digibarn.com/history/06-11-4-VCF9-Apple30-panel.slides/index.html
The Modern Era: 1950 to the Present (con't)

- Fifth generation of computing (1985-?)
  - Recent developments
    - Massively parallel processors
    - Handheld devices and other types of personal digital assistants (PDAs)
    - High-resolution graphics
    - Powerful multimedia user interfaces incorporating sound, voice recognition, touch, photography, video, and television
The Modern Era: 1950 to the Present (con't)

- Recent developments (con't)
  - Integrated global telecommunications incorporating data, television, telephone, fax, the Internet, and the World Wide Web
  - Wireless data communications
  - Massive storage devices
  - Ubiquitous computing
<table>
<thead>
<tr>
<th>Generation</th>
<th>Approximate Dates</th>
<th>Major Advances</th>
</tr>
</thead>
</table>
| First      | 1950–1957        | First commercial computers  
                         First symbolic programming languages  
                         Use of binary arithmetic, vacuum tubes for storage  
                         Punched card input/output |
| Second     | 1957–1965        | Transistors and core memories  
                         First disks for mass storage  
                         Size reduction, increased reliability, lower costs  
                         First high-level programming languages  
                         First operating systems |
| Third      | 1965–1975        | Integrated circuits  
                         Further reduction in size and cost, increased reliability  
                         First minicomputers  
                         Time-shared operating systems  
                         Appearance of the software industry  
                         First set of computing standards for compatibility between systems |

Figure 1.8
Some of the Major Advancements in Computing
| Fourth | 1975–1985 | Large-scale and very-large-scale integrated circuits  
|        |          | Further reduction in size and cost, increased reliability  
|        |          | First microcomputers  
|        |          | Growth of new types of software and of the software industry  
|        |          | Computer networks  
|        |          | Graphical user interfaces |
| Fifth  | 1985–?   | Ultra-large-scale integrated circuits  
|        |          | Supercomputers and parallel processors  
|        |          | Laptops and handheld computers  
|        |          | Wireless computing  
|        |          | Massive external data storage devices  
|        |          | Ubiquitous computing  
|        |          | High-resolution graphics, visualization, virtual reality  
|        |          | Worldwide networks  
|        |          | Multimedia user interfaces |

**Figure 1.8**

Some of the Major Advancements in Computing
Organization of the Text

- This book is divided into six separate sections called levels
- Each level addresses one aspect of the definition of computer science
- Computer science/algorithms
Organization of the Text (con't)

- Level 1: The Algorithmic Foundations of Computer Science
  - Chapters 1, 2, 3

- Level 2: The Hardware World
  - Chapters 4, 5

- Level 3: The Virtual Machine
  - Chapters 6, 7
Organization of the Text (con't)

- Level 4: The Software World
  - Chapters 8, 9, 10, 11

- Level 5: Applications
  - Chapters 12, 13, 14

- Level 6: Social Issues
  - Chapter 15
Figure 1.9
Organization of the Text into a Six-Layer Hierarchy
Summary

- Computer science is the study of algorithms.
- An algorithm is a well-ordered collection of unambiguous and effectively computable operations that, when executed, produces a result and halts in a finite amount of time.
- If we can specify an algorithm to solve a problem, then we can automate its solution.