Chapter 10
Algorithmic Thinking
The Letter Algorithm

- Bauby was paralyzed and could communicate only by blinking one eyelid.
- An assistant would say or point to a letter and Bauby would indicate if it was the right one.
- Point to letters until the correct one is reached.
- Repeat to spell words and sentences.
The Letter Algorithm

- This process is an algorithm: A precise, systematic method for producing a specified result
- We invent algorithms all the time
- An algorithm need not use numbers
- The agent running an algorithm may be a human being, rather than a computer
- There are better and poorer versions of this algorithm
The Letter Algorithm

• Making the process faster
  - Completion: The assistant can guess a word before it is all spelled out
  - Ask the letters in frequency order, and work from the most-frequently-used letter downward
Programs are Algorithms

• **programs**: algorithms that have been specialized to a specific set of conditions and assumptions
  – usually written in a specific programming language
• usually the words program and algorithm are used interchangeably
Algorithms vs. Heuristic Processes

- not all processes given in the book are algorithms
- the process to find information on the web using a search engine was not an algorithm
  - not systematic
  - not guaranteed to find it (process could fail)
  - called a **heuristic process**: helpful procedure for finding a result
Algorithm Properties

• An algorithm *must* have five properties:
  1. Input specified
  2. Output specified
  3. Definiteness
  4. Effectiveness
  5. Finiteness
1. Input Specified

- The **input** is the data to be transformed during the computation to produce the output.
- What data do you need to begin to get the result you want?
- Input precision requires that you know what kind of data, how much and what form the data should be.
2. Output Specified

• The output is the data resulting from the computation (your intended result)
• Frequently the name of the algorithm contains the output:
  – “Algorithm to compute batting average”
• Output precision also requires that you know what kind of data, how much and what form the output should be (or even if there will be any output at all!)
3. Definiteness

- Algorithms must specify every step and the order the steps must be taken in the process.
- Definiteness means specifying the sequence of operations for turning input into output.
- Details of each step must be spelled out (including how to handle errors).
4. Effectiveness

- For an algorithm to be effective, each of its steps must be doable
- The agent must be able to perform each step without any outside help or extraordinary powers
5. Finiteness

• The algorithm must stop, eventually!
• Stopping may mean that you get the expected output OR you get a response that no solution is possible
• Finiteness is not usually an issue for non-computer algorithms
• Computer algorithms often repeat instructions with different data and finiteness may be a problem
Algorithm Fact #1

1. Algorithms can be specified at different levels of detail
   – Algorithms use functions to simplify the algorithmic description
   – These functions (such as scan) may have their own algorithms associated with them
Algorithm Fact #2

2. Algorithms always build on functionality previously defined and known to the user
   – Assume the use familiar functions and algorithms
   – For example, “scan through” would use the Intersecting Alphabetized Lists (IAL) from Chapter 5
Algorithm Fact #3

3. Different algorithms can solve the same problem differently, and the different solutions can take different amounts of time (or space)
How Do We Know it Works?

• Algorithm solution is clear and simple and efficient
• Then, how do we know it works?
• If there is no loop, the program runs, gets to an end, and we can check the result
• What if there is a loop?
  – Programs with loops cannot be absolutely verified…there are too many possible cases
Then, what?

• The way to know that an algorithm works is to know why it works…

• Strategy for knowing why it works:
  – Find one or more properties that ensure the algorithm works
  – Explain, using the program, why they make it work.
Correctness

• anyone who creates an algorithm needs to know why it works
  – finding the algorithm's correctness-preserving properties and explaining why they do the job