Chapter 7: What's The Plan?: Algorithmic Thinking

It has nothing to do with Al Gore – but thinking of him will at least get you pronouncing it correctly!

Algorithm

- A precise, systematic method for producing a specified result
- We have already seen some:
  - Hex to binary
  - Computing ISBN check digit
  - Estimating area
  - Estimating three letter words
  - Figuring out how many desserts are possible

Algorithms in Everyday Life

- Some algorithms are learned—arithmetic
- Some we figure out ourselves—looking up a phone number, sorting tests into alphabetical order
- Others require written instructions—recipe, assembly instructions, driving directions

Five Essential Properties of Algorithms

Think of a problem such as ISBN computation

1. **Input specified**
   - Data to be transformed during the computation to produce the output
   - Must specify type (numeric, string, fraction), amount, and form of data

2. **Output specified**
   - Data resulting from the computation—intended result
   - It is possible to have no output
Five Essential Properties (cont’d)

3. Definiteness
   - Specify the sequence of events
   - Details of each step, including how to handle errors
   - Recipes may not satisfy - imprecise

4. Effectiveness
   - The operations are doable

5. Finiteness
   - Must eventually stop

Language of Algorithms

- **Natural language**
  - For people, we use a natural language like English/Spanish
  - **Ambiguity** is common in natural language
    - *I am going to the bank* (Citibank or riverside)
    - Answer the questions in green.
    - A picture of Nick in Providence (who is in Providence, Nick or the picture?)
    - royal purple gown (is royal purple the color or is in a regal gown that is purple?)
    - I saw the man with the telescope (who has the telescope, me or the man?)

- **Programming Language**
  - Formal languages designed to express algorithms
  - Formal Language: set of words often defined by means of a formation rules. Meaning is precise.
  - Precisely defined; no ambiguity
  - if you want an A
    - if you are an auditory learner
      - ask lots of questions
    - otherwise ask no questions.
  - Programming language decides which "if" the "otherwise" applies to.
  - Formal language typically consists of
    - variable declarations
    - sequence control (sequential, conditional, loop)

Example of variable declaration:

Variable declarations
- weight is a real; represents weight of produce in pounds
- prodName is a string; represents name of produce item

Integer, real, or string is termed the type of the variable.

Similar to types in high school: jocks, nerds, …

What they do and what they look like.
sequence control
- sequential
  - Add the shortening cream with the sugar.
  - Add the eggs
- conditional
  - if the eggs are large, use two
  - otherwise use three
- loop
  - Bake thirty minutes
test the cake with toothpick
  - While toothpick does not come out clean
    {   bake five more minutes
test the cake with toothpick
}

In Class Feb 17th  \text{\textsuperscript{th}}  \text{groupsize} \leq 2
- Logon to the CSILM system. Select \textit{Algorithms – Peg Interchange}.
- Answer the questions written in green.
- Turn in a paper/pencil answer sheet at the end of class.
- If you choose to turn in the assignment via email, complete the super* questions as well.

Finding the check digit of an ISBN number
\textit{Input:} the \textit{first nine digits of an ISBN number}
\textit{Output:} the checkDigit
\text{Variables:}
- \textit{a,b,c,d,e,f,g,h,i} - integers, represent 1\textsuperscript{st} through 9\textsuperscript{th} digit of ISBN
- \textit{Sum, integer} - weighted sum of digits
- \textit{remainder} - remainder of Sum/11
- \textit{checkDigit} - final digit
\text{Step 1: Perform the following computation}
\textit{Sum} = 10^9a + 9^8b +8^7c +7^6d +6^5e + 5^4f + 4^3g + 3^2h + 2^1i
\text{Step 2: Take the remainder of Sum divided by 11}
\text{Step 3: If the remainder is 0, the checkDigit is 0}
\text{Otherwise, checkDigit is 11 minus remainder}

At seats
- Write the algorithm to add the parity row/column.
- Be as formal as possible – state input, outputs, and instructions.
The Context of a Program

- Programs can fulfill the five properties of an algorithm, be unambiguous, and still not work right because it is executed in the wrong context. Part of the context can be the input (like only working if you have exactly 9 integers of the ISBN number).
- Context matters: Navigation instructions
  - “From Old Main go past the Cazier Library and turn right.” Assumes you got to the Library a specific way. If you do not, the directions will fail.
  - Start the car and go North 10 miles. Assumes you know how to drive, have a car, have gas.

Program vs. Algorithm

- A program is an algorithm that has been customized to solve a specific task under a specific set of circumstances using a specific language
- Algorithm is a general method; program is a specific method
  - Algorithm: sort nametags
  - Program: sort these 73 nametags in Java

An Algorithm: Alphabetize Nametags

- Imagine nametags in a stack, not organized
- You want to alphabetize by first name
- How do you solve this problem?

One idea

- input: set of nametags on the table
- output: nametags in a stack in alpha order
  - Look through all the nametags on the table. Put a sticky note (or your finger) on the largest one so far.
  - When you have looked at all remaining nametags, move the largest nametag to the top of the “done stack” on another table.
- Repeat the previous steps until there are no more left
Analysis

• Does it sort correctly?
• How much work does it do?
• Count how many comparisons I do:
  – Find largest of n
  – Find largest of n-1
  – Find largest of n-2
  – …
  – Find largest of 1
  – n+(n-1) + (n-2) + … + 1

In Class Feb 18  groupsize ≤ 2

• Give an algorithm that you design for alphabetizing nametags (different from the insertion sort I gave).
• Have a person from a different group follow the algorithm PRECISELY and see if it works. List any problems that occurred. Revise the algorithm to deal with the problems.
• How fast is the algorithm? If time, have a race between two groups with different algorithms
• Turn in a paper/pencil answer sheet at the end of class.
• If you choose to turn in the assignment via email, that is also permitted.
• This assignment score will replace the lowest of your previous inclass assignments. It will not have a separate entry in the gradebook.

Things to remember

Rather than submitting two files (and having to zip them together), just paste the screen shot into your word document. You can email as a LAST RESORT. It is NOT the preferred way of submitting.

If you submit homework 6 as homework 5, there are two problems:
1. you erase your previous homework 5. It now looks late. And if the grader isn’t done with it, it is GONE.
2. The grader never finds homework 6.

How can we tell which is the best algorithm?

• We want fastest!
• We count “operations” – how many times we had to look at each nametag or how many times we had to do a comparison.
• Consider my “selection sort” of n nametags.
  1. Repeat n times:
     – Find the biggest and set it aside (k operations if there are k nametags)
  Total work: n + (n-1) + (n-2) + … 1 = ???
  If n = 100, work is about 5050 operations
  We call this an n² algorithm as the time grows with the square of the problem size.
Jeremy/Thayne Algorithm

Input: unsorted nametags
Output: sorted nametags
1. Obtain nametags from teach or UTF
2. Place cards on table with writing face up
3. Review the ABC song in your minds
4. Starting for left to right, place the names in approximate order
5. Then analyze your results and refine
6. Verify by singing the ABC song while reading the names of your cards

Similar to a Shell sort. Harder to analyze. Placing in approximate order requires (1) estimating location (1 operation) and then looking at surrounding nametags to find the right location. Let’s say you have to look at 10 each time. If there are 100 nametags – 1000 operations!

Megan/Chandler Algorithm

Input: stack of nametags
Output: alphabetized stack of nametags
1. Look through the big stack and pull out all the A’s
2. Then sort those by the second letter. After that, by the third letter.
3. Repeat with each of the 26 letters.

Like a bucket sort, only more handling of nametags. If there are 100 names (about 4 in each bucket), Find all the A’s takes 100 operations. Then I have to pick them up 8 times (if 8 is the length of the longest name). Just to sort the A’s is 800 operations. To sort all would be about 26*800/2 (as sometimes finding all the names beginning with a give letter would be fast) = 10400

Ashley/Rebecca Algorithm

• separate into piles by first letter
• Then sort each pile into piles by second letter. And so forth
• Now combine the piles

Bucket sort. Depends on length of string. If the longest name is 10 letters and you have 100 names, how many times did you pick up each nametag? =100*10 = 1000

In general, might have many more than 26 characters (upper lower case, hyphens, apostrophe). Picking up all the stacks starts to affect timing.

Ryan/Preston/Kevin Algorithm

1. Each person took a stack of names
2. We then each individually sorted them by first letter.
3. Then we put them together and put them again in different piles using the first letter of the name

We won the race

Parallel algorithm. Good idea if you have multiple processors. This is like a regular bucket sort, but you would divide the time by 3 as there were three “processors” all working at the same time. = 100*10/3 = 333
Clelia/Stephanie/Pete Algorithm
Input: stack of nametags
Output: alphabetized stack of nametags
1. Separate into piles by first letter
2. Pick up stacks one at a time and sort using insertion sort

Combination algorithm. First step takes \( n \) operations. At that point, you have fairly small stacks. About \( n/26 \) in size.

To do an insertion sort of \( k \) things is \( k + (k-1) + (k-2) + \ldots + 1 = k(k+1)/2 \)

Clearly this is better than a regular insertion sort as the piles are so much smaller.
First step is 100 operations. Second step is \( 26 \times (4 \times 5/2) = 260 !!! \)

Recursion
- I remember two traumatic experiences from gradeschool.
  1. The only word I mis-spelled in second grade was ‘room’. I wrote my \( r \) sloppily and the teacher CLAIMS she thought I wrote voom. Now really!
  2. I defined a word in terms of itself. That was not allowed. This is why I love math and computer science. Recursive definitions are applauded.

Consider this recursive algorithm
Called a mergesort
Divide the nametags into two halves.
Sort each half
Merge the two groups together

Can you see the recursion? How good of an idea is it?

Consider this recursive algorithm
Called a quicksort
Pick a random nametag as a pivot.
Divide into two groups – one larger than the pivot and one smaller.
Sort the two pieces.

Can you see the recursion? How good of an idea is it?
Recursive drawings

- Many shapes are “self-similar” - their overall structure is repeated in sub-pictures.
- Peano curve:

Recursive drawings (cont.)

“Koch snowflake”: Every straight line in one version has a upside down v inserted.

Recursive drawings (cont.)

We assume the little fish also has a fish inside.
Let's try a few recursive problems

• Sorting, recursively
• Finding the student with the highest GPA, recursively

Let's try a few recursive problems

• Draw the following pattern (given the number of * in the first row):
  
  ******
  ****
  **
  **
  ******

Let's try a few recursive problems

Write a method with one positive int parameter called n. The method will write 2n-1 integers.
Here are the patterns of output for various values of n:
n=1: Output is: 1
n=2: Output is: 1 2 1
n=3: Output is: 1 2 1 3 1 2 1
n=4: Output is: 1 2 1 3 1 2 1 4 1 2 1 3 1 2 1
And so on. Note that the output for n always consists of the output for n-1, followed by n itself, followed by a second copy of the output for n-1.
In Class Feb 20th groupsize ≤ 2

- Logon to the CSILM system. Select Recursion Tower of Hanoi.
- Answer the questions written in green.
- Turn in a paper/pencil answer sheet at the end of class.
- If you choose to turn in the assignment via email, complete the super* questions as well.