Computer Science I
COP 3502 – Introduction

Computer Science Department
University of Central Florida

Computer Science 1: Introduction

- How is COP3502 different than COP 3223?
  - COP 3223 teaches how to program in C
    - Language basics, variable declarations, conditional expressions, if statements, loops, functions, arrays, structures, etc.
    - **This will not be covered in this class**
    - You will need to freshen up on your C very quickly
      - If you need help, but a good C-language book or find a quality reference online
      - My favorite is "**C by Dissection**"
  - With respect to the C language, we will cover:
    - Pointers, 2D arrays, and linked lists

Computer Science 1: Introduction

- The goals of Computer Science I:
  - Improve knowledge of standard data structures and abstract data types
  - Improve knowledge of standard algorithms used to solve several classical problems
  - Cover some mathematical concepts that are useful for the analysis of algorithms
  - Analyze the efficiency of solutions to problems

Computer Science 1: Introduction

- The goals of Computer Science I:
  - In COP 3223, we only cared if we found a solution to the problem at hand
    - Didn't really pay attention to the efficiency of the answer
  - For this class:
    - We learn standard ways to solve problems
    - And how to analyze the efficiency of those solutions
    - Finally, we simply expand upon our knowledge of our use of the C programming language

Computer Science 1: Introduction

- Teaching Method:
  - This class is NOT used to teach you C
    - The focus of COP 3223 (not this class) is to teach you C
      - You should know C already
    - In COP 3223, majority of time was spent going of syntax
    - Programs were often shown in class
    - Programs were even written during class
      - Essentially a requirement for any course teaching a programming language

Computer Science 1: Introduction

- Teaching Method:
  - Majority of this class is used covering algorithm analysis, abstract data types, and new data structures
  - The teaching of these concepts dictate more explanation and less of a focus on “code”
    - Some code will be shown on the PowerPoint slides
      - Such as after we explain a new abstract data type
      - We’ll show the code of how you would implement it
    - **<u>However, writing of actual code will most likely never be done in class</u>**
      - Again, that is not the purpose of this class

Computer Science 1: Introduction

- Example Problem:
  - We will now go over two solutions to a problem
    - The first is a straightforward solution that a COP 3223 student should be able to come up with
      - Doesn't' care about efficiency
    - The second solution is one that a COP 3502 student should be able to come up with after some thought
      - Cares about efficiency
  - Hopefully this example will illustrate part of the goal of this course

Computer Science 1: Introduction

- Max Number of 1's:
  - You are given an nxn integer array
    - Say, for example, a 100x100 sized array
  - Each row is filled with several 1's followed by all 0's
    - Example:
      - Row 1 may have 38 1's followed by 62 0's
      - Row 2 may have 73 1's followed by 27 0's
      - Row 3 may have 12 1's followed by 82 0's
      - You get the idea
  - The goal of the problem is to identify the row that has the maximum number of 1's.

Computer Science 1: Introduction

- Max Number of 1's:
  - Straightforward COP 3223 style solution:
    - Make a variable called MaxOnes and set equal to 0
    - For each row do the following:
      - Start from the beginning of the row on the left side
      - Scan left to right, counting the number of 1's until the first zero is encountered
      - If the number of 1's is greater than the value stored in MaxOnes, update MaxOnes with the number of 1's seen on this row
  - Clearly, this works
  - But let's see how long this algorithm will take

Computer Science 1: Introduction

- Max Number of 1's:
  - Analysis of Straightforward Solution:
    - Basically we iterate through each square that contains a 1, as well as the first 0 in each row
    - If all cells were 0, we would only "visit" one cell per row, resulting in n visited cells
    - However, if all cells were 1's, we would "visit" all of the cells (n^2 total)
      - So in the worst case, the number of simple steps the algorithm takes would be approximately n^2
    - This makes the running time of this algorithm $O(n^2)$
      - The meaning of this Big-O will be discussed later in the semester

Computer Science 1: Introduction

- Max Number of 1's:
  - Analysis of Straightforward Solution:
    - There seems to be extra work done here
    - Once we know that a row has 12 1's, for example, it seems pointless to start checking at the beginning of the next row
      - Why not just start at column 12
      - If it's a 0, then that row can't be the winner
      - If it is a 1, then clearly there is no point in going back, on that row, and checking the previous 11 squares
    - This idea leads to a more efficient algorithm

Computer Science 1: Introduction

- Max Number of 1's:
  - More Efficient COP 3502 style algorithm:
  1. Initialize the current row and current column to 0
  2. While the current row is less than n (or before the last row)
     a. While the cell at the current row and column is 1
        - Increment the current column
     b. Increment the current row
  3. The current column index represents the maximum number of 1's seen
  4. Now let's trace through a couple of examples

Computer Science 1: Introduction

- Max Number of 1's:
  - Example 1:

1	1	0	0	0	0
0	0	0	0	0	0
1	1	1	0	0	0
1	1	1	0	0	0
1	1	1	1	1	0
1	1	1	1	0	0

Computer Science 1: Introduction

- Max Number of 1's:
  - Example 2:

0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1

Computer Science 1: Introduction

- Max Number of 1's:
  - Analysis of Better Solution:
    - How many steps will this algorithm take, in terms of n?
      - Each "step" taken by the algorithm either goes to the right or down in the table.
      - There are a maximum of n-1 steps to the right
      - And a maximum of n-1 steps down that could be taken
      - Thus the maximum number of "steps" that can be done during this algorithm is approximately 2n
        - And this is the worst case
      - So the running time of this algorithm is O(n)
        - An improvement of the previous algorithm
        - Input size of 100 for n
        - n^2 would be 10,000 steps and 2n would be 200 steps

Computer Science 1: Introduction

- Implementing an Algorithm in C:
  - In this class, you will have an opportunity to improve upon your ability to write programs that implement an algorithm you have learned
  - You must know the syntax of C in order to properly and effective do this
  - There's no set way to create code to implement an algorithm
    - But this example shows some steps you can take in doing so

Computer Science 1: Introduction

- Implementing an Algorithm in C:
  - Here are some issues to think about:

1. What data structures are going to be used?
2. What functions are going to be used?
3. What run-time errors should we protect against?
4. What atypical cases may we have to deal with?
5. What is an efficient way to execute the steps in the algorithm?

Computer Science 1: Introduction

- Maximum Number of 1's
  - This was a creative exercise
    - Much of what you learn in class will not be
  - We have many set algorithms and data structures that you will study
  - Occasionally you will have to come up with new ideas like this one
  - Mostly, however, you will simply have to apply the data structures and algorithms shown in class fairly directly to solve the given problems

CS1 - Introduction

Are

You

Excited?

Daily Demotivator

Computer Science I
COP 3502 – Introduction

Computer Science Department
University of Central Florida