SIENA
COMPUTER SCIENCE

Computer Science 385
Design and Analysis of Algorithms
Siena University
Spring 2026

Problem Set 4

Group Formation: 4:00 PM, Monday, March 23, 2026
Due: 4:00 PM, Friday, March 27, 2026

You may work alone or with a partner or two on this assignment. However, in order to make sure you learn the material and are well-prepared for the exams, you should work through the problems on your own before discussing them with your partner(s), should you choose to work with someone. In particular, the "you do these and I'll do these" approach is sure to leave you unprepared for the exams.

Submitting

Please submit a hard copy (typeset preferred, handwritten OK but must be legible) for all written questions. Only one submission per group is needed.

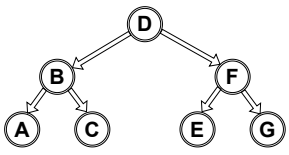
Tree Traversals

Recall the idea of tree traversals, specifically in the context of binary trees. Three types of binary tree traversals can be specified recursively, and are shown below, along with an example binary tree, which happens to be a binary search tree.

ALGORITHM PREORDER(T)
  if T not empty **then**
    visit the root of T
    PREORDER(T_L)
    PREORDER(T_R)

ALGORITHM INORDER(T)
  if T not empty **then**
    INORDER(T_L)
    visit the root of T
    INORDER(T_R)

ALGORITHM POSTORDER(T)
  if T not empty **then**
    POSTORDER(T_L)
    POSTORDER(T_R)
    visit the root of T



Question 1: For each traversal type, give the order in which tree nodes are visited on the tree above. (3 points)

Question 2: How many times is INORDER called if initially called on the root of the sample tree above as input? Be sure to count all the base cases because it gets called on a lot of empty trees... Show your work or at least briefly explain how you obtained your answer. (2 points)

For the next four questions, Suppose INORDER is called on a tree with n nodes, x of which have two children, y of which have exactly one child and z of which are leaf nodes.

Question 3: Give a formula for x in terms of n, y, and z. (1 point)

Question 4: Exactly how many base case calls to INORDER are made? Express your answer in terms of n, x, y, and/or z. (2 points)

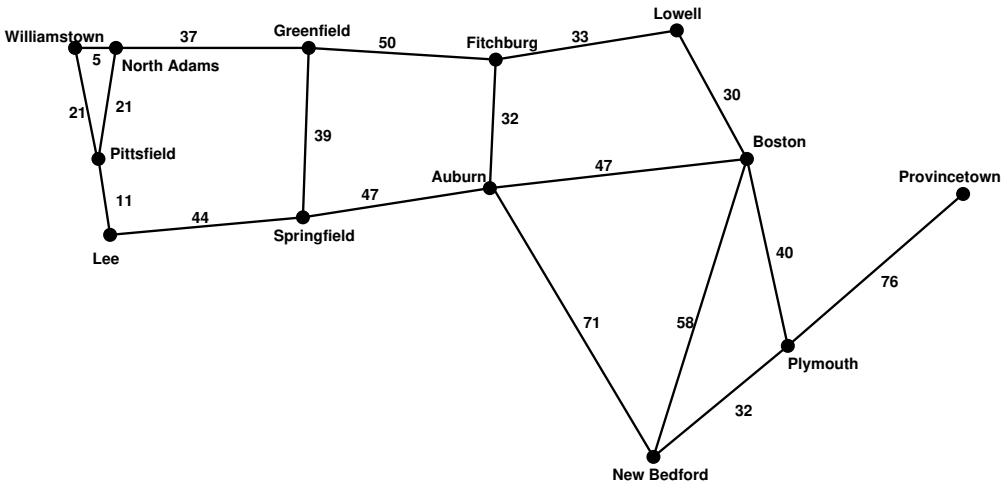
Question 5: How many total calls to INORDER are made (count all base case calls and non-base case calls)? Express your answer in terms of n, x, y, and/or z. (2 points)

Question 6: Give an upper bound on the total number of calls to INORDER in terms of just n and using Θ-notation. Note that x, y and z are all less than or equal to n. (2 points)

Question 7: For each call to INORDER, a constant number of operations are done to check for the base case and to print the data. Knowing an upper bound on the total number of calls made and that a constant number of operations are done per call, what is the Big-Θ running time of INORDER when called on a tree of n nodes? (1 point)

Graph Traversals

For the next two questions, please refer to the graph below. You may ignore the edge weights. Break ties by alphabetical order.



Question 8: Perform a breadth-first traversal of the graph starting at Auburn. Using notation as in the example in Levitin Section 3.5, show the contents of the queue and the order in which they are added. You do not need to include cross edges in your breadth-first traversal tree. (5 points)

Question 9: Perform a depth-first traversal of the graph starting at Auburn. Using notation as in the example in Levitin Section 3.5, show the contents of the stack and the order in which they are pushed. You do not need to include back edges in your depth-first traversal tree. (5 points)

Your Internship in Admissions

In college admissions, one measure of similarity between schools is the number of "cross admits" (*i.e.*, students who applied to and were offered admission to both). A large number of cross admits would mean the schools are seen by college-bound students as being similar. Your task is to take the lists of students admitted to college A and B and count the number of names that appear in both. Your input consists of two arrays: `A[0...n-1]` contains the names of n students admitted to college A, and `B[0...m-1]` contains the names of m students admitted to college B. For example, for the two unreasonably small arrays below, 2 should be returned as the answer, because students "D. Chambers" and "I. Jones" were the only ones in both arrays. You may assume there are no duplicate names in the arrays, and that no two students exist with the same name.

```
A[0...7] = {"S. Cooper", "D. Chambers", "H. Stone", "M. Brady",
            "A. Fonzarelli", "I. Jones", "P. Venkman", "L. Organa"}

B[0...4] = {"I. Jones", "M. McFly", "F. Gump", "D. Chambers", "M. Simpson"}
```

Question 10: You could easily write a brute force algorithm solving this problem in $\Theta(nm)$ time. But the number of students in both arrays is very large, and so an asymptotically more efficient algorithm is required. Write such an algorithm. You may not introduce any new data stuctures like balanced search trees or hash tables. (20 points)

ALGORITHM NUMCROSSADMITS(A, B)
//Input: $A[0..n-1]$, admits to college A
//Input: $B[0..m-1]$, admits to college B
//Output: the numbers of names that appear in both A and B

Question 11: Give the Θ efficiency class of your algorithm and briefly explain how you arrived at this bound. (3 points)

Similar Spring Climates

Suppose you have a large array of numbers representing the average April high temperatures in a set of cities, *e.g.*:

$$A = \{76.3, 44.0, 56.1, 55.8, 50.2, 62.3, 37.2, 77.9\}$$

Your task is to find the pair of average high temperatures that are closest together in value. For this example, the answer would be 56.1 and 55.8.

Question 12: Develop pseudocode for an algorithm to solve this problem the operates in better than $\Theta(n^2)$ time. (20 points)

Question 13: Give the Θ efficiency class of your algorithm and briefly explain how you arrived at this bound. (3 points)

The Viral Growth Window

For the remaining questions, consider the following problem.

As a world-famous social media influencer who is really a computer scientist at heart, you can't resist analyzing the performance of your content and promotions to see what you can learn to accelerate your path toward world domination. You have a record of your *net follower growth* on an hourly basis over a period of several years, and you would like to find your best *viral growth window*. This is the contiguous streak of days where you saw the absolute highest net growth.

For example, consider the changes in follower count shown below for a period of 15 hours. The values indicate the amount by which the follower count changed during that hour. Positive values indicate a net increase, while negative values indicate a net decrease.

```
-18 -3 +15 +21 +15 +9 -15 -21 -3 +30 +3 +15 -60 +30 +3
```

We want to calculate the maximum net follower count change for all possible contiguous time periods. Note that this means you pick a start hour and an end hour. For example, if you consider the range starting at hour 2 and ending at hour 5, then you would have a net follower count change of 60. However, if you start at hour 2 and end at hour 11, then you would have a net follower count change of 69. (Note: we're all computer scientists here, so the first hour is numbered 0.)

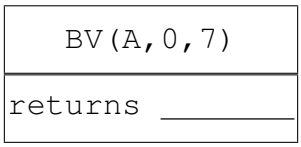
We could easily solve this problem in $\Theta(n^2)$ examining all pairs of start and end hours, where n is the number hours in the period being considered. But that's last month's news and you have a **lot** of data. We can do better!

Question 14: Improve upon this by giving a divide and conquer algorithm solving this problem that runs asymptotically faster than $\Theta(n^2)$. Start with the pseudocode below. It returns the maximum net follower change you experienced between some start hour and some end hour. A complete and correct base case is provided. (20 points)

ALGORITHM BESTVGW(A, s, e)
  //Input: an array of follower count changes $A[0..n-1]$
  //Input: start (s) and end (e) hours of the subrange to consider
  //Output: the maximum possible net follower change
  if $s = e$ **then**
    // only one hour in the range, so just the net follower change of that hour
    return $A[s]$

Question 15: Once you have your algorithm, complete the recursive call tree started below, showing all the recursive calls your algorithm will make and the return value from each. (6 points)

The root of your tree should look like what you see below, with `BV` (short for `BestVGW`) and the blank filled in with the answer that the call returns. All other nodes in your tree should be formatted similarly.



Use `A = [6, -50, 2, 13, 5, -14, 1, 6]` as your input array.

Question 16: Give a recurrence formula for the worst case running time of your algorithm. (2 points)

Question 17: What is the Θ efficiency class based on your recurrences. (You may find the closed form of your recurrence in any way you like or apply the Master Theorem, if applicable.) (3 points)

Grading

This assignment will be graded out of 100 points.

Feature	Value	Score
Question 1	3	
Question 2	2	
Question 3	1	
Question 4	2	
Question 5	2	
Question 6	2	
Question 7	1	
Question 8	5	
Question 9	5	
Question 10	20	
Question 11	3	
Question 12	20	
Question 13	3	
Question 14	20	
Question 15	6	
Question 16	2	
Question 17	3	
Total	100	