SIENA*college*
Computer Science

Computer Science 385
Design and Analysis of Algorithms
Siena College
Spring 2024

Problem Set 2

Due: 9:00 AM, Thursday, February 15, 2024

You may work alone or in a group of 2 or 3 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, 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.

All GitHub repositories must be created with all group members having write access and all group member names specified in the `README.md` file by 4:00 PM, Wednesday, February 7, 2024. This applies to those who choose to work alone as well!

There is a significant amount of work to be done here, and you are sure to have questions. It will be difficult if not impossible to complete the assignment if you wait until the last minute. A slow and steady approach will be much more effective.

Getting Set Up

In Canvas, you will find a link to follow to set up your GitHub repository, which will be named `ps2-yourgitname`, for this lab. Only one member of the group should follow the link to set up the repository on GitHub, then others should request a link to be granted write access.

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.

Your submission requires that all required code deliverables are committed and pushed to the master for your repository's origin on GitHub. If you see everything you intend to submit when you visit your repository's page on GitHub, you're set.

Programming Task: Exhaustive Search

Sometimes an exhaustive search algorithm considers all permutations of a set. It is not obvious though how to generate these permutations in a program. Write a program that prints all permutations of integers $1..n$. Your program should take n as an input value, and it should work for any $n \geq 1$. If you would like to use a language other than Java, ask, and it will likely be approved. (12 points)

Take a highly recursive approach: choose each next possible unused value and follow it by all permutations of the remaining values. Do not use algorithms like those in Levitin Section 4.3. We're focusing on brute force solutions here.

Question 1: Explain clearly how your program works, including the purpose of any major variables and/or parameters. (3 points)

Homework Set Problems

Question 2: Levitin Exercise 2.3.1, p. 67 (2 points)

Question 3: Levitin Exercise 2.3.2, p. 67 (2 points)

Question 4: Compare the asymptotic growth rates of the following pairs of functions and decide if one grows faster than the other or if they grow at the same rate. Prove your answers using limits. All logarithms are base 2, unless otherwise noted. (5 points)

a. $\frac{2}{9}n^3 + 6n^2 - 12$ and n^3

b. $n \log^2 n$ and $n \log n$

c. $4^{\log n}$ and n

d. 2^n and $2^{\frac{3n}{2}}$

e. n^a and n^b, where a and b are arbitrary fixed constants such that $a > b > 0$.

Question 5: Prove the following using limits (4 points):

a. $n \in \Omega(\log n)$

b. $2n^4 + 16n \in \Theta(n^4)$

c. $n \in \Theta(n + n^{\frac{1}{2}})$

d. $\frac{2}{5}n^2 - 17n + 3 \in O(n^2 \log n)$

Question 6: For each of the following functions, indicate the class $O(g(n))$ to which the function belongs. Use the "smallest" $g(n)$ possible to obtain the tightest bound, unless otherwise specified. Prove your assertions using the definition of $O(g(n))$ (*i.e.*, produce constants). (6 points)

a. $7n^2 + 9n - 1200$

b. $(n^2 + 5n)^3$

c. $n \log n$, use $g(n) = n^2$

Question 7: For each of the following functions, indicate the class $\Omega(g(n))$ to which the function belongs. Use the "largest" $g(n)$ possible to obtain the tightest bound unless otherwise specified. Prove your assertions using the definition of $\Omega(g(n))$ (*i.e.*, produce constants). (6 points)

a. $7n^2 + 9n - 1200$

b. $2^n + n^{100}$

c. $n \log n$, use $g(n) = n$

Question 8: For each of the following functions, indicate the class $\Theta(g(n))$ to which the function belongs. Prove your assertions using the definition of $\Theta(g(n))$ (*i.e.*, produce constants). (6 points, 2 for the first, 4 for the second)

a. $7n^2 + 9n - 1200$ (hint: you've already done this, just bring it all together)

b. $5n^3 - 128n^2 - 50$

For the next four questions, consider this very wintry code segment.

```
for ( i = 1; i <= n; i++ )
    System.out.println("Snowstorms!");
    System.out.println("Noreasters!");
    for ( j = 1; j <= i; j++ ) {
        System.out.println("Blizzards!");
        System.out.println("Squalls!");
    }
}
```

Question 9: Trace the code to count how many messages it prints for both $n = 2$ and $n = 3$. (1 point)

Question 10: Write an expression involving two summations that counts the number of messages it prints for any value of n. (2 points)

Question 11: Simplify your expression to get a closed form formula for the number of messages it prints. Show your work. (2 points)

Question 12: Check your formula from the previous question by substituting $n = 2$ and $n = 3$ to see if it matches your hand-traced answers to the first question in this section. (1 point)

Question 13: Suppose you have a gift certificate from your favorite restaurant worth exactly \$$d$. You want to buy exactly two different meals from the restaurant, and you want their cost to exactly equal \$$d$ so that you spend the full amount of the gift certificate. Write a brute force algorithm that takes as input an array of the costs of the meals you like and outputs true if there are two distinct meals in the array whose costs sum to \$$d$ For example, suppose the array that contains the meal prices is P = {4.00, 2.25, 7.25, 10.00, 11.00, 8.20, 12.75, 3.50, 9.99}, and the gift certificate's value d is \$10.00. Then the output would be true because \$2.25 + \$7.25 = \$10.00. For this exercise, design a brute force algorithm solving this problem using $\Theta(n^2)$ operations in the worst case. Use the pseudocode below to get started. You do not need to write an implementation, just a pseudocode algorithm. (10 points)

```
ALGORITHM GiftCertificate($P$, $d$)
    //Input: a set of meal prices $P[0..n-1]$
    //Input: a gift certificate value $d$
    //Output: true only if 2 distinct prices in $P$ add to exactly $d$
```

Question 14: What does it mean for a sorting algorithm to be *stable*? (1 point)

Question 15: Describe two circumstances where sorting of data is needed, such that for one of the circumstances it is important that the algorithm used for sorting is stable, and one where it does not matter. (2 points)

Question 16: Is bubble sort as we studied in class a stable sorting algorithm? Explain briefly. (1 point)

Question 17: Is selection sort as we studied in class a stable sorting algorithm? Explain briefly. (1 point)

Question 18: Levitin Exercise 1.3.1, p. 23 (4 points)

Question 19: Count the number of character comparisons made by the **BruteForceStringMatch** algorithm when applied to the pattern and text below. Explain how you arrived at your answer. (4 points)

text: `FROM_COMPUTER_SCIENCE_COMES_COMPUTING_FUN`

pattern: `COMPUTERS`

Note that the text is 41 characters long, and the pattern is 9 characters long.

Question 20: Levitin Exercise 3.2.6, p. 107 (3 points)

For the next 4 questions, suppose you wish to take an input string and count how many substrings you can form from it such that each substring starts with `[` and ends with `]`. For example, the input string `]385[` contains 0 such substrings, and the input string `B[][RA[[]CK][[E]T[S` contains 15 such substings.

Question 21: List the 15 substrings of the input string above that start with `[` and end with `]`. (3 points)

Question 22: Write pseudocode for a very brute-force algorithm that will solve this problem. Do not try to be efficient. (5 points)

Question 23: Determine the efficiency class of your algorithm for an input string of length n, and briefly justify. Specify both best and worst cases if appropriate (don't worry about trying to nail down an average case). (4 points)

Question 24: Now write a more efficient (but still just brute-force) algorithm to solve this problem in $\Theta(n)$ time. (5 points)

Question 25: Consider an instance of the Knapsack Problem in which there are n=4 items with weights 13, 7, 4, and 15 and the knapsack can hold up to a total weight of 24. The item values are 30, 15, 40, and 60. Enumerate all the packings of the knapsack that would be considered using an exhaustive search algorithm solving this problem, and circle the packing that would be selected as the best solution. (5 points)

Grading

This assignment will be graded out of 100 points.

Feature	Value	Score
Recursive Permutations	12	
Q1	3	
Q2	2	
Q3	2	
Q4	5	
Q5	4	
Q6	6	
Q7	6	
Q8	6	
Q9	1	
Q10	2	
Q11	2	
Q12	1	
Q13	10	
Q14	1	
Q15	2	
Q16	1	
Q17	1	
Q18	4	
Q19	4	
Q20	3	
Q21	3	
Q22	5	
Q23	4	
Q24	5	
Q25	5	
Total	100	