ICS 141: Discrete Mathematics for Computer Science I Spring 2025

Problem Set 5

Kyle Berney *Due: Tuesday, April 1, 2025 at 1:30pm*

1 Relative Asymptotic Growths (20 pts)

Fill in the table with "Yes" or "No" in each empty box, indicating whether $f(n)$ is O, o, Ω, ω, or Θ of $g(n)$. Assume that $k \geq 1$, $\epsilon > 0$, and $c > 1$ are constants. For each row, justify your choice, by simplifying expressions, applying asymptotic definitions or properties, and/or using any known facts.

	$f(n)$	$g(n)$	O	o	Ω	ω	Θ
a.	$\log^k n$	n^ϵ					
b.	c^n	n^k					
c.	$n^{\log c}$	$c^{\log n}$					
d.	$\log(n!)$	$\log(n^n)$					

2 Runtime Analysis (15 pts)

Consider the following pseudocode:

```
1: LoopyLoop(A[1...n])
2:     for i = 2 to n
3:         for j = 1 to i − 1
4:             for k = 1 to 10
5:                 A[i] = k · A[j]
```

(a) **(10 pts)** Compute the exact number of times line 5 executes. Show your work (using summations).

(b) **(5 pts)** State the asymptotic runtime of LoopyLoop($A[1 \dots n]$) using Θ notation.

3 Find All Modes (25 pts)

(a) **(15 pts)** Given an array A of n positive integers, a histogram is an array H where each index represents a value and the corresponding entry stores the count of occurrences of that value in the dataset (i.e., $H[i]$ is the number of occurrences of the integer i in A). For example, the histogram for $A = [3, 1, 2, 3, 3, 2, 1, 4]$ is $H = [2, 2, 3, 1]$.

Design an iterative algorithm Histogram($A[1 \dots n], max$) that returns an array $H[1 \dots max]$ representing the histogram for the input array A. Provide psuedocode and prove that your algorithm correctly computes the histogram for A using a loop invariant. State precisely the loop invariant and clearly show the initialization, maintenance, and termination conditions. ***No points will be given for an algorithm without a proof of correctness (i.e., loop invariant).***

(b) **(10 pts)** Given an array A of n positive integers, a mode of A is an element that occurs at least as often as each of the other elements. Note that when more than one element appears the maximum number of times, there exists multiple modes. For example, the modes of $A = [2, 3, 1, 1, 2]$ are 1 and 2.

Using the algorithm developed in part (a), design an iterative algorithm FindAllModes($A[1 \dots n]$) that returns an array containing all of the modes of A. Provide psuedocode and a high-level description of your algorithm that justifies why your algorithm is correct. ***No points will be given for an algorithm without a justification of correctness.***

4 Binary Search (40 pts)

Consider the following psuedocode for finding an element x in a sorted array $A[1 \ldots n]$.

```
1:  BinarySearch(A[1...n], x)
2:      left = 1
3:      right = n
4:      while left ≤ right
5:          mid = ⌊(left + right)/2⌋
6:          if x == A[mid]
7:              return mid
8:          else if x < A[mid]
9:              right = mid − 1
10:         else
11:             left = mid + 1
12:     return Not Found
```

(a) **(15 pts)** State precisely the loop invariant for the **while** loop in lines 4-11 and prove that this loop invariant holds. Your proof should clearly show the initialization, maintenance, and termination conditions.

(b) **(20 pts)** Using strong induction, prove that the **while** loop in lines 4-11 will execute at most $1 + \log n$ times. *Hint: Observe what happens to the size of the subarray $A[left \ldots right]$ after each iteration.*

(c) **(5 pts)** Using part (b), conclude that the runtime of BinarySearch is $\Theta(\log n)$.