CSI 2110C Computer Science Fall 2016 University of Ottawa

Assignment #1 (35 points, weight 5%)
Due: Wednesday September 28, 9:30PM

- **The assignment is either to be uploaded on the blackboard electronically (you may type or write by hand legibly and scan it. Only a single PDF file is accepted) or to the assignment box in paper (write your name in every page).**
- **Late submission is accepted for two days with 50% minus - Within 24 hours, 25% minus and within 48 hours, 50% minus.**

1. (2 marks) Given an *n*-element array A, Algorithm X executes an *O(n)*-time computation for each even number in A and an *O(log n)*-time computation for each odd number in A.

   (a) What is the best-case running time of Algorithm X?

   (b) What is the worst-case running time of Algorithm X?

2. (9 marks) Use the definition of "*f(n) is O(g(n))*" to prove the following statements.

(a) $f(n)=7n^3+3n^2-2n+100$ is $O(n^3)$.

(b) $f(n) = (n^2 + 1)/(n + 1)$ is $O(n)$.

(c) $f(n)=n!$ is $O(n^n)$.

(d) $f(n) = \log_2 n$ is $O(\log_{10} n)$.

(e) $f(n) = n^3$ **is not** $O(100n^2)$.

(f) $f(n) = 2^{n+1}$ is $\Theta(2^n)$

3. Given an array, A, of n integers, give an $O(n)$-time algorithm that finds the longest subarray of A such that all the numbers in that subarray are in sorted order. Your algorithm outputs two integers: the initial and final indices of the longest subarray. A solution that uses extra memory that is in $O(1)$ is worth 100%; if you use $\Theta(n)$ extra memory your solutions is worth 80%.
Example: If n = 10 and A= [8,6,7,10,–2,4,5,6,2,5] then the algorithm outputs 4 and 7, since A[4..7] = [–2, 4, 5, 6] is the longest sorted subarray.

(a)(4 marks) Give the algorithm pseudocode.

(b)(1 mark) Justify your big-Oh (1 mark).

4. Suppose you are given a sorted array, A, of n distinct integers in the range from 1 to n +1, so there is exactly one integer in this range missing from A. Give an O(log n)-time algorithm for finding the integer in this range that is not in A. Give pseudocode (3 marks). Justify your big-Oh (1 mark). Hint: the algorithm resembles binary search

(a)(4 marks) Give the algorithm pseudocode.

(b)(1 mark) Justify your big-Oh (1 mark).

5. (4 marks) Fill a table showing a series of following queue operations and their effects on an initially empty queue *Q* of integer objects. Here *Q* is implemented with an Array of size 7.

Operation	**Output Q**
enqueue (4)	
dequeue ()	
dequeue ()	
enqueue (44)	
enqueue (7)	
enqueue (6)	
dequeue ()	
isEmpty()	
enqueue (3)	
enqueue(5)	
dequeue ()	
dequeue ()	
dequeue ()	
dequeue ()	
enqueue(32)	
enqueue(39)	
enqueue(9)	
size()	
enqueue (32)	
size()	
dequeue ()	
enqueue (6)	
enqueue (5)	
Dequeue ()	

front()	
size()	
enqueue (9)	

6. (2 marks) Give an example of a positive function $f(n)$ such that $f(n)$ is neither $O(n^2)$ nor $\Omega(n^2)$. Explain both assertions.

7. (3 marks) Give a big-Oh characterization, in terms of n, of the running time of the following method. Show your analysis!

```
public void Ex(int n)
int a = 1;
for (int i = 0 ; i < n*n ; i++)
  for (int j = 0; j <= i; j++)
     if( a <= j)
       a = i;
}
```

8. Give a big-Oh characterization (in terms of the number *n* of elements stored in the queue) of the running time of the following method. Show your analysis!

(a) (4 marks) Describe how to implement the queue ADT using two stacks. That is: write pseudocode algorithms which implement the *enqueue()* and *dequeue()* methods of the queue using the methods of the stack.

(b) (1 mark) What are the running times of your *dequeue()* and *enqueue()* algorithms?