Student Name: ______________________ ________ J#: ___________________

CSC 323 Algorithm Design and Analysis, Spring 2018
Instructor: Dr. Natarajan Meghanathan

Exam 1 (Take Home) **Due: Feb. 20th @ 1 PM, Hardcopy in Class** **Max. Points: 100**

Answering and Submission: Print this Exam (one page per paper; DO NOT print front and back)

Answer in the space provided. You can also answer on the back side of a paper. Feel free to use additional sheets for any answer if needed.

Submit a hardcopy (either printed or handwritten-version in class at 1 PM; handwritten-versions should be legible enough to read).

Q1 - 12 pts) Solve the following recurrence relation (without using Master Theorem)
$C(n) = C(n/2) + \log n$, for $n > 1$. $C(1) = 0$

Student Name: ______________________ ________ J#: ___________________

Q2 - 10 pts) Let $T(n) = 7T(n/2) + 3n^2 + 2$.
Using Master Theorem and the Limits approach, show that $T(n) = O(n^3)$

Q3 - 10 pts) Consider an array of prime integers in the range [1...20] with the entries randomly distributed. Find the average number of comparisons for a sequential search in the array. Show all work.

Q4 - 8 pts) Let $T(n) = 3T(n/4) + n\log n$. Solve the recurrence using Master Theorem.

Student Name: ______________________ ________ J#: ___________________

Q5 - 15 pts) Two pairs of integers (a, b) and (c, d) are said to be symmetric if b = c and a = d. For example, given an array of pairs { {31, 57}, {80, 90}, {25, 70}, {90, 80}, {70, 25}, {11, 20}, {10, 5}, {30, 40} }, the symmetric pairs are:
{80, 90} and {90, 80}
{25, 70} and {70, 25}

Design a hash table-based algorithm of time complexity Θ(n) to identify the symmetric pairs in an array of 'n' pairs.
Show the working of your algorithm for the above array of pairs with a hash function H(K) = K mod 7.

Student Name: ______________________ ________ J#: ___________________

Q6 - 15 pts) Consider the following two variants of the pseudo code for the Insertion Sort algorithm. Using each variant, sort the array: 5_1 5_2 5_3 5_4 5_5. Note that $5_1, 5_2, \ldots, 5_5$ are five different instants of integer 5 and need to be treated as separate elements (that are of the same numerical value). Determine the number of comparisons encountered with each of the two variants of the algorithm to sort the above array and what is the final sorted array (include the suffixes of the elements throughout your work).

```
Input: Array A[0...n-1]
Begin
for (index i = 1 to n-1) do
    v = A[i]
    index j = i-1
    while (index j ≥ 0) do
        if (v ≥ A[j]) then
            break 'j' loop
        else
            A[j+1] = A[j]
        end if
        j = j-1

    end while
    A[j+1] = v
End
```

Pseudo Code - I

```
Input: Array A[0...n-1]
Begin
for (index i = 1 to n-1) do
    v = A[i]
    index j = i-1
    while (index j ≥ 0) do
        if (v > A[j]) then
            break 'j' loop
        else
            A[j+1] = A[j]
        end if
        j = j-1

    end while
    A[j+1] = v
End
```

Pseudo Code - II

Student Name: ______________________ ________ J#: ___________________

Q7 - 15 pts) Develop a recursive version of the Bubble Sort algorithm.
(a) Write the pseudo code of the algorithm and justify that it is recursive and works correctly.
(b) Write the recurrence relation for the algorithm and solve it using one of the two approaches discussed in class, as appropriate. Solve the recurrence relation and show that the time complexity of the recursive algorithm is $\Theta(n^2)$.

Student Name: ______________________ ________ J#: ___________________

Q8 - 15 pts) The number of inversions is an array is the number of (i, j) pairs (where i and j are index positions and each pair is considered only once) such that A[i] > A[j] and i < j.

For example, the following array has 5 inversions as shown.

Index	0	1	2	3	4
Array, A	**10**	**50**	**40**	**20**	**30**

Index Pairs	Inversions
(1, 2)	A[1] > A[2]
(1, 3)	A[1] > A[3]
(2, 3)	A[2] > A[3]
(1, 4)	A[1] > A[4]
(2, 4)	A[2] > A[4]

(a) Modify the pseudo code of the Insertion Sort algorithm so that it can compute the number of inversions in an array. Write the modified pseudo code and justify your modification.
(b) Analyze whether the modification would have any impact on the asymptotic time complexity of Insertion sort.
(c) Run your modified version of the Insertion sort algorithm on the above array and show that it can determine the number of inversions to be 5.