Lahore University of Management Sciences (LUMS)
CS202/EE202: Data Structures (Spring'18)
Homework-1

Q1. Arrange each of the following groups of functions in increasing order of asymptotic (big-O) complexity. That is, if g(n) immediately follows f(n) in your list, then it must be the case that f(n) is in O(g(n)). You must explain your reasoning.

(a) Group 1:

$$\begin{aligned} f_1(n) &= n^{0.999999} \log n \\ f_2(n) &= 10000000n \\ f_3(n) &= 1.000001^n \\ f_4(n) &= n^2 \end{aligned}$$

(b) Group 2:

$$\begin{aligned} f_1(n) &= 2^{2^{1000000}} \\ f_2(n) &= 2^{100000n} \\ f_3(n) &= \binom{n}{2} \\ f_4(n) &= n\sqrt{n} \end{aligned}$$

(c) Group 3:

$$f_1(n) = 10^n$$
$$f_2(n) = n^{1/3}$$
$$f_3(n) = n^n$$
$$f_4(n) = \log_2 n$$
$$f_5(n) = 2^{\sqrt{\log_2 n}}$$

(c) Group 4:

$$f_1(n) = n^{2.5}$$
$$f_2(n) = \sqrt{2n}$$
$$f_3(n) = n + 10$$
$$f_4(n) = 10^n$$
$$f_5(n) = 100^n$$
$$f_6(n) = n^2 \log n$$

Q2. that you have two functions f and g such that f(n) is in O(g(n)), decide whether each of the following statements is true or false. If true, prove it. If false, give a counterexample.

a) log f(n) is in O(log g(n))

b) $2^{f(n)}$ is in $O(2^{g(n)})$

c) $f(n)^2$ is in $O(g(n)^2)$

Q3. Suppose you are given an array A of n integers A[1], A[2], . . . , A[n]. You want to output an nxn array B in which B[i, j] (for i < j) contains the sum of array entries A[i] through A[j]. Here is a pseudocode to solve this problem.

```
For i = 1, 2, . . . , n
        For j = i + 1, i + 2, . . . , n
                Add up array entries A[i] through A[j]
                Store the result in B[i, j]
        Endfor
Endfor
```

a) For an input of size n, give a bound of the form O(f(n)) on the running time of this algorithm.
b) For this same function f, show that the running time of the algorithm for an input of size n is also in Ω(f(n)).
c) What does this tell you about the asymptotically tight bound on the running time of this algorithm?
d) Give a different algorithm to solve this problem, with an asymptotically better running time.
e) (A clear pseudocode is enough)

Q4. a) What is the asymptotic time complexity of the following program fragment. Show your working.

```
for (int i=1; i<=n; i*=2) {
        j=i;
}
```

b) i) What is the asymptotic time complexity of the following program fragment. Show your working. Give both the upper and lower bound.

```
int x, y, n;
```

```
. . .
/* P is a 1-D array of size n integers and W is a 2-D array of size n x n
integers */
        for (x = 0; x < n; x++) {
                for (y = x+1; y < n; y++) {
                        W[x][y] = func(P, x, y);
                }
. . .
        }
```

The function func() called from the main program above is defined as follows:

```
int func(int *array, int i, int j){

        int m, val = 0;
        for (m = i; m <= j; m++) {
                val += array[m];
        }
        return (val);
}
```

b) ii) What is being stored in the 2-D array W?

b) iii) Change the program fragment given above so that it becomes more efficient. Mention the time complexity of the improved code.

Q5. Following is C++ like pseudo code of a function that takes a number as an argument, and uses a stack S to do processing.

```
void bar(int n)
{
        Stack S; // Say it creates an empty stack S
        while (n > 0)
        {
// This line pushes the value of n%2 to stack S
                push(&S, n%2);

                n = n/2;
        }

// Run while stack is not empty
            while (!isEmpty(&S))
                cout << pop(&S); // pop an element from S and print it
}
```

Describe what does the above function do in general?

Q6. a) Describe a way to detect a loop in a singly linked list and mention the time complexity of your algorithm?

b) Describe a way to reverse the elements of a stack only using stack operations like push() and pop() and mention the time complexity of your algorithm?

c) Given a binary search tree on integers, describe an algorithm that returns an array containing the values from the tree in the range [low...high] inclusive. The array should preserve the order of the values and mention the time complexity of your algorithm. **(You are just expected to provide a description of your algorithms in this question)**

Q7. We want to implement an Abstract Data Type (ADT) with operations **ShowMin, GetMax, Insert, and isPresent**. The operation **ShowMin** reports the min without deletes it, while **GetMax** both reports the max and deletes it. The operation **isPresent(x)** returns a boolean value representing whether x is in the ADT or not. For each of the following parts, describe a data structure (either a standard data structure or a modification of a standard data structure or a data structure of your own design) that implements the ADT and fulfills the constraints. The value n denotes the number of items in the data structure.

a) **ShowMin, isPresent, Insert and GetMax** each take time **O(log n)** in the worst case.
b) **ShowMin and Insert** each take time **O(1)** in the worst case.
c) **GetMax** takes time **O(1)** in the worst-case and **isPresent** takes time **O(log n)** in the worst case