Sample Mid Term for Review

1.

a. Show that n^2 + 2 is *not* O(n) [do it in class]

b. Let f(n) = (lg n)^k and g(n) = n^{ϵ} . Find the relationship between f and g considering asymptotic growth **(Sample Medium to hard type of question)** [do it in class]

2.

a. Iteration (s)

$$\begin{cases} T(n) = T(n-1) + 2 \\ T(1) = 1 \end{cases}$$

Ans.

T(n)= T(n-1) +2*1
T(n)= T(n-2) +2*2
T(n)= T(n-3) +2*3
T(n)= T(n-4) +2*4

Generalized recurrence relation at the kth step of the recursion:
T(n)= T(n-k) +2*K

We want T(1). So we let n-k = 1. Solving for k, we get k = n - 1. Now plug back in.

T(n) = T(n-k) + 2*k
T(n) = T(1) + 2*(n-1), and we know T(1) = 1
T(n) = 2n-1
We are done. Right side does not have any T(…)'s. This recurrence relation is now solved in its closed form, and it runs in O(n) time.

Anoher Iteration -

a. *T*(*n*) = 2*T*(*n* - 1) + 1, T(0) = 0 [do in class]

b. Other - Master, guess etc **(sample Medium to hard type of question)**

Show that in the recurrence

$$T(n) = \max_{0 \le q \le n-1} (T(q) + T(n-q-1)) + \Theta(n) ,$$

$$T(n) = \Omega(n^2).$$

Ans. Do it in class

3.

a. Radix, Bucket, Counting

(Soln. to HW6 problem 2) **(Sample Medium to hard type of question)**

b. Other sorting

Use the QuickSelect algorithm to manually compute the 4th smallest

element of the array [1, 5, 23, 0, 8, 4]. Assume that the rightmost element is used as the pivot in each case. Show what happens in each self-call, indicating the new input array and the current value of *k*.

```
QS (1, 5, 23, 0, 8, 4), k = 4

L = [1,0], E = [4], G = [5, 23, 8]

k' = k - |L| - |E| = 4 - 3 = 1


QS (5, 23, 8), k = 1
```

L = [5], E = [8], G = [23]

k = 1

```
QS (5), k = 1

    L = [ ],  E = [5],  G = [ ]

            |L| < k ≤ |L| + |E|, so return 5
```

Can be from Quicksort, Mergesort etc,

4.

a. Chap 1, Proof etc.

Prove any one of the following:

Theorem 1. Every comparison based sorting algorithm has, for each n, running on input of size n, a worst case in which its running time is $\Omega(n\log n)$.

Theorem 2. The average case running time of QuickSort is $O(n\log n)$.

b. Sorting

Algorithm	**Worst Case / Stable / In-Place**
Insertion Sort	**$O(N^2)$ / Yes / Yes**
Bubble Sort	**$O(N^2)$ / Depends / Yes**
Heap Sort	**O(N log N) [Later]**
Merge Sort	**O(N log N) / Yes / No**

Quick Sort	**$O(N^2)$ / No / If Partition is in Place**
Binary Search	**O(N) - Later**
Radix Sort	**O(N) / Yes / No**
Counting Sort	**O(N) / Yes / No**

c. Amortization

A *twisty table* is a kind of data structure that stores Strings and that uses an array in the background (assume it is already initialized and has as many available slots as necessary without resizing) and that has three primary operations:

- AddOne(x) – adds String x to the background array in the next available slot
- AddThree(x,y,z) – adds Strings x, y, and z to the background array by placing them in the next three available slots.
- ClearAll() – removes all Strings currently in the array by setting them to null.

Show that the amortized running time of ClearAll is O(1). Do the steps required by filling in the following table. Hint: For your amortized cost function, try charging 2 cyberdollars for AddOne, 6 cyberdollars for AddThree and 0 cyberdollars for ClearAll.

The cost function c:

```
c(AddOne)   = 1
c(AddThree) = 3
c(ClearAll) = k, where k is number of non-null Strings in the
                                  array
```

Your amortized cost function ĉ:

$$
\begin{aligned}
\hat{c}(\text{AddOne}) &= 2 \\
\hat{c}(\text{AddThree}) &= 6 \\
\hat{c}(\text{ClearAll}) &= 0
\end{aligned}
$$

Show that your amortized cost function never results in negative amortized profit:

If AddOne is executed k times and AddThree is executed m times, the profit will be k + 3m, which is enough to pay for a ClearAll operation, which would cost k + 3m.

What bound (upper or lower) can Amortized analysis give?