CSE 100 Midterm 1 - Harjot Grewal

1) a) $n \log n = O(n^2)$ $n \log n \leq c(n^2)$ $n = 4$

$c \geq 1/4$

True

$4 \log 4 \leq c(16)$

$4 \leq c(16)$

$\frac{1}{4} \leq c$

b) $\log \log n = O(\log n)$

$\log(\log(n)) \leq c \log(n)$

True

c) $\log^{50} n = O(n^{.1})$ $\log_{50}(n) \leq c(n)^{.1}$

$n = 1$ $0 \leq 1$ False

d) $4^n = O(2^n)$ $4^n \leq c(2^n)$

False $4^n \leq 2^n$

e) $100^{100} = \Theta(1)$ $100^{100} \leq O(1)$

$100^{100} \leq 1$

False

f) if $f = O(g)$, then $g = \Omega(f)$ $f \leq c(g)$ $g \geq c(f)$

$\frac{f}{g} \leq c$ $\frac{g}{f} \geq c$

True

g) if $n^3 = \Omega(n^2)$ $n^3 \geq c(n^2)$ $n \geq c$

True

Harjot Grewal

Problem 1 Cont.

h) False

i) $100 + 200n + 300n^2 = \Theta(n^2)$

True

$100 + 200n + 300n^2 \leq c(n^2)$

j) $\sum_{i=1}^{n} \log i = \Theta = (n \log n)$

True

$O: n \log n \leq c(n \log n)$

$\Omega: n \log n \geq c(n \log n)$

CSE 100 Midterm 1 - Harjot Grewal

2) A[1...8] = ⟨7, 4, 2, 9, 4, 3, 1, 6⟩

Before j=5 starts, A[1...8] is ⟨2, 4, 7, 9, 4, 3, 1, 6⟩

J=2 A[1...8] = ⟨7, 4, 2, 9, 4, 3, 1, 6⟩

J=3 A = ⟨4, 7, 2, 9, 4, 3, 1, 6⟩

J=4 A = ⟨2, 4, 7, 9, 4, 3, 1, 6⟩

CSE 100 Midterm 1 - Harjot Grewal

3) a) $n = 10$ $A[1...10] = \langle 5, 2, 15, 1, 7, 6, 8, 5, 4, 3 \rangle$
An unsorted array would terminate in $\Omega(n^2)$

b) $n = 10$ $A[1...10] = \langle 1, 2, 3, 4, 5, 5, 6, 7, 8, 15 \rangle$
A sorted array would terminate in $O(n)$

CSE 100 Midterm 1 - Harjot Grewal

4) Recurrence: $T(n) = 8T(\frac{n}{2}) + \Theta(n^2)$

$T(n) = aT(n/b) + cn^b$ $a=8$ $b=2$ $c=\Theta$ $\Rightarrow$ $\boxed{\Theta(n^3)}$

CSE 100 Midterm 1 - Harjot Grewal

5) a) Runtime of Line 3: $\Theta(n)$ - recursive tree cost n at every level

b) Runtime of line 4: $\Theta(n)$ - recursive tree cost n at every level

c) Runtime of Line 5: $\Theta(n)$ - merge takes linear time

CSE 100 Midterm 1 - Harjot Grewal

6) Formally prove that $50n+15 = O(n^2)$ using def. $O(\cdot)$.

$f(n) = 50n+15$
$g(n) = O(n^2)$

$f(n) = O(g(n))$ $\quad f(n) \leq C \cdot g(n)$

$$\frac{f(n)}{g(n)} \leq C$$

$$\frac{50n+15}{n^2} \leq C \text{ for all } n \geq 1$$

$$\frac{50n^2+15n^2}{n^2} > \frac{50n+15}{n^2} \text{ for all } n > 1$$

$$65 > \frac{50n+15}{n^2} \text{ for all } n > 1$$

$$\frac{50n+15}{n^2} \leq 65, \; f(n) = O(g(n))$$

CSE 100 Midterm 1 - Harjot Grewal

7) Heapsort (A)

1. for i = A.length downto 2
2. exchange A[1] with A[i]
3. A.heap-size = A.heap-size - 1
4. Max-Heapify(A, 1)

Assume that the heap A you're given is already a max-heap

CSE 100 Midterm 1 - Harjot Grewal

8) Insertion-Sort(A)

1. for j=2 to A.length ← Loop Invariant
2. key = A[j]
3. //Insert A[j] into the sorted sequence A[1...j-1].
4. i = j-1
5. while i>0 and A[i]>key
6. A[i+1] = A[i]
7. i = i-1
8. A[i+1] = key

Loop Invariant: Line 1

Initialization/Base: The array is already sorted at line 1.

Maintenance/Inductive Step: The while loop finds the correct index on where to place the new number.

Termination: Once the for loop ends at j=A.length because of line 1, the array must be sorted.

CSE 100 Midterm 1 - Harjot Grewal

9) Selection-Sort(A)

```
1. n = A.length
2. for j = 1 to n-1        ← Loop Invariant
3.     smallest = j
4.     for i = j+1 to n
5.         if A[i] < A[smallest]   ← Loop Invariant
6.             smallest = i
7.     exchange A[j] with smallest]
```

Loop Invariant: Line 2 & Line 5

The loop invariant starts at smallest element and then sorts it.

CSE 100 Midterm 1 - Harjot Grewal

10) Assume that Merge(A, p, q, r) successfully merges two sorted arrays, A[p...q] and A[q+1...r], into a sorted array A[p...r].

Merge-Sort Loop Invariant: Line 1, its the base case

Initialization/Base Case: You start by finding the q and split the array from there

Maintenance/Inductive Step: At every call to merge sort, if you aren't at the base case you'll split up the array again

Termination/Conclusion: At the base case you call merge which will sort the individual arrays at every level.