

CSCI-GA.1170-001 Homework 1

Ankit Sati

TOTAL POINTS

30.5 / 43

QUESTION 1

Same loop variable used.

1 Asymptotic Comparisons 8.5 / 10

✓ - **2 pts** *Incorrect c*

+ **0.5** *Point adjustment*

1c) 0.5 for \$O\$

1 Polynomial grows faster than logarithm

QUESTION 2

2 Order of growth 12 / 12

✓ - **0 pts** *Correct*

QUESTION 3

3 Bubble Sort 10 / 21

✓ + **3 pts** *For part (g) - Will be correctly compensated by other deductions.*

✓ - **4 pts** *Missing/incorrect solution for c*

✓ - **3 pts** *Missing/incorrect solution for d*

✓ - **2 pts** *Missing/incorrect solution for e*

✓ - **3 pts** *Missing/incorrect solution for g*

✓ - **0 pts** *Correct*

✓ - **0.5 pts** *Missing termination condition for loop invariant in part (b)*

✓ - **0.5 pts** *Missing base case condition for loop invariant in part (b)*

✓ - **0.5 pts** *Missing maintenance condition for loop invariant in part (b)*

✓ - **0.5 pts** *Logic error in part (a)*

2

Q. 1.

a. $f(m) = (m^3)^3$; $g(m) = m^{(m^3)}$

$f(m) = m^{3m}$.

taking log both sides. $g(m) = m^{m^3}$

$f(m) = 3m \log m$.

$g(m) = m^3 \log m$.

$$\lim_{m \rightarrow 0} \frac{f(m)}{g(m)} = \frac{3m \log m}{m^3 \log m} = \frac{3}{m^2} = 0.$$

Since $\lim_{m \rightarrow 0} \frac{f(m)}{g(m)} = 0$. (We can conclude)

Answer.

① $f(m) = o(g(m))$ 2. $f(m) = O(g(m))$ (Little & Big Oh.)

b) $f(m) = m!$; $g(m) = (m+1)!$

$f(m) = 1 \times 2 \times 3 \times \dots \times (m-1) \times m$; $g(m) = 1 \times 2 \times 3 \times \dots \times (m-1) \times m \times (m+1)$

$f(m) \approx m^m$; $g(m) \approx m^{m+1}$

$$\lim_{m \rightarrow 0} \frac{f(m)}{g(m)} = \frac{m^m}{m^{m+1}} = \frac{m^{(m-m-1)}}{m} = \frac{1}{m} = 0.$$

We can conclude the following.

Answer.

① $f(m) = o(g(m))$ 2. $f(m) = O(g(m))$ (Little & Big Oh.)

c. $f(n) = n^{0.99} + 15 (\log_2 n)^{100}$; $g(n) = \log_{16} n^2$

$f(n) = n^{0.99} + 15 (\log_2 n)^{100}$; $g(n) = n^{2 \cdot \log_{16} 4} = n^{2 \times 1/2} = n$

$\boxed{\text{Asymptotically equal}} = f(n) \Theta g(n)$ $g(n) = n$

Alternatively we can take log both sides

$f(n) = 0.99 \log n + 1500 \log \log n$; $g(n) = \log n$

we can conclude that $\boxed{\begin{array}{l} \textcircled{1} f(n) = \Theta g(n) \\ \textcircled{2} f(n) = O g(n) \\ \textcircled{3} f(n) = \Omega g(n) \end{array}}$ \leftarrow Answer

4. $f(n) = n^{0.000001}$; $g(n) = (\log n)^{1000001}$

Taking logs both the sides

$f(n) = 0.000001 \log n$; $g(n) = 1000001 \log \log n$

$f(n) \geq g(n)$

we can conclude $\boxed{\begin{array}{l} \textcircled{1} f(n) = \Omega g(n) \\ \textcircled{2} f(n) = \omega g(n) \end{array}}$ \leftarrow Answer

5. $f(n) = n^{5/\log_2 n}$; $g(n) = 10000$

Take logs both sides

$f(n) = \frac{5}{\log_2 n} \times \log n$; $g(n) = \log 10000$

$f(n) = 5$; $g(n) = \log 10000$

\therefore Asymptotically equal

$$f(n) \approx g(n)$$

$$\begin{array}{|l} \textcircled{1} f(n) \Theta g(n) \\ \textcircled{2} f(n) O g(n) \\ \textcircled{3} f(n) \Omega g(n) \end{array}$$

→ Answer.

Problem 1-2.

$$1. \frac{n^2 + 2}{1 + n^3 \cdot 2^{-n}} \approx \frac{n^2}{n^3 \cdot 2^{-n}} = \frac{2^n}{n} \approx \Theta(2^n) \leftarrow \text{Answer.}$$

$$2. \log(\sqrt{\log(35n)}) \cdot \log(37n^3 + 45) = \log_2 n^{1/3} \log_2 n^3 \\ \approx \frac{1}{3} \times 3 \times \log n \times \log n = (\log n)^2 \approx \Theta(\log n)^2 \leftarrow \text{Answer.}$$

$$3. \log(n!) + 10^{205} n \approx \log(n^n) + n \approx n \log n + n \\ \approx \Theta(n \log n). \leftarrow \text{Answer.}$$

$$4. \frac{6^n - 1000}{2^n + 1} \approx \frac{6^n}{2^n} \approx 3^n \approx \Theta(3^n). \leftarrow \text{Answer.}$$

$$5. n^{2/\log n} + 1000 \approx n^{2 \cdot 1/\log n} \approx \sqrt[n]{n^2} \\ \approx \Theta(1) \leftarrow \text{Answer.}$$

$$6. 2^{n + \log n} \approx 2^n \cdot 2^{\log n} \approx 2^n \cdot n \cdot \log 2 \\ \approx 2^n \cdot n \approx \Theta(n 2^n) \leftarrow \text{Answer.}$$

1 Asymptotic Comparisons 8.5 / 10

✓ - 2 pts *Incorrect c*

+ 0.5 *Point adjustment*

1c) 0.5 for O

1 Polynomial grows faster than logarithm

$$f(n) \approx g(n)$$

$$\begin{array}{|l} \textcircled{1} f(n) \Theta g(n) \\ \textcircled{2} f(n) O g(n) \\ \textcircled{3} f(n) \Omega g(n) \end{array}$$

→ Answer.

Problem 1-2.

$$1. \frac{n^2 + 2}{1 + n^3 \cdot 2^{-n}} \approx \frac{n^2}{n^3 \cdot 2^{-n}} = \frac{2^n}{n} \approx \Theta(2^n) \leftarrow \text{Answer.}$$

$$2. \log(\sqrt{\log(35n)}) \cdot \log(37n^3 + 45) = \log_2 n^{1/3} \log_2 n^3 \\ \approx \frac{1}{3} \times 3 \times \log n \times \log n = (\log n)^2 \approx \Theta(\log n)^2 \leftarrow \text{Answer.}$$

$$3. \log(n!) + 10^{205} n \approx \log(n^n) + n \approx n \log n + n \\ \approx \Theta(n \log n). \leftarrow \text{Answer.}$$

$$4. \frac{6^n - 1000}{2^n + 1} \approx \frac{6^n}{2^n} \approx 3^n \approx \Theta(3^n). \leftarrow \text{Answer.}$$

$$5. n^{2/\log n} + 1000 \approx n^{2 \cdot 1/\log n} \approx \sqrt[n]{n^2} \\ \approx \Theta(1) \leftarrow \text{Answer.}$$

$$6. 2^{n + \log n} \approx 2^n \cdot 2^{\log n} \approx 2^n \cdot n^{\log 2} \\ \approx 2^n \cdot n \approx \Theta(n 2^n) \leftarrow \text{Answer.}$$

$$7. 2^m + 10 (\log m)^{\log m} \approx 2^m = \Theta(2^m) \leftarrow \text{Answer.}$$

$$8. 2^{3m} + 4^m \approx 2^{3m} + 2^{(m+1)} \approx 2^{3m} + 2^m \\ \approx \Theta(2^{3m}) \leftarrow \text{Answer.}$$

b) Functions per asymptotic growth. (least to maximum.)

$$\cancel{1, (\log m)^2, m \log m, 2^m, 2^m, m 2^m, 3^m,}$$

$$1, (\log m)^2, m \log m, 2^m, 2^m, m 2^m, 2^{3m}, 3^m.$$

2 Order of growth 12 / 12

✓ - 0 pts Correct

Problem 1-3

A.

```
bubbleSort(A,n)
for j=0 to n-1
    checkSwap=false
    for j = 0 to n-1 2
        if A[j] > A[j+1] then
            swap( A[j], A[j+1] )
            checkSwap = true
        end if
    end for
    if(not checkSwap) then
        break
    end if
end for
end bubbleSort return A
```

B.

Proof of Correctness Loop Invariant

After each iteration of the loop greatest element of the array is always placed at right most position. So, at the end of i iteration right most i elements are sorted and in place. After the N-1 iterations the right most N-1 items are sorted and this implies that all the N items are sorted.

Worst Case Time Complexity: $O(n*n)$

Worst case occurs when array is reverse sorted.Example:9,8,7,6,5,4

Best Case Time Complexity: $O(n)$.

Best case occurs when array is already sorted.Example:4,5,6,7,8,9

C.

D.

```
void bubbleSort(int arr[], int n)
{
    int i, j;
    bool swapped;
    for (i = 0; i < n-1; i++)
    {
        swapped = false;
        for (j = 0; j < n-i-1; j++)
        {
            if (arr[j] > arr[j+1])
            {
                swap(&arr[j], &arr[j+1]);
            }
        }
    }
}
```

```

        swapped = true;
    }
}
if (swapped == false)
    break;
}
}

```

In this example, we need to take the best case time complexity.
Best Case Time Complexity: $O(n)$

E.

```

void insertionSort(int arr[], int n)
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i-1;
        while (j >= 0 && arr[j] > key)
        {
            arr[j+1] = arr[j];
            j = j-1;
        }
        arr[j+1] = key;
    }
}

```

The while loop executes only if $i > j$ and $arr[i] < arr[j]$. Therefore total number of while loop iterations (For all values of i) is same as number of inversions.

Therefore overall time complexity of the insertion sort is $O(n + f(n))$ where $f(n)$ is inversion count.

If the inversion count is $O(n)$, then the time complexity of insertion sort is $O(n)$.

F.

An array example for which bubble sort runs for $O(N)$ and insertion sort runs in $O(N^2)$: {1,2,5,3,4}

In bubble sort:

$A = \{1, 2, 5, 3, 4\}$

Comparing 1 and 2: Since $2 > 1$ no swapping will happen. Array stays the same {1,2,5,3,4}

Comparing 2 and 5: Since $5 > 2$ no swapping will happen. Array stays the same {1,2,5,3,4}

Comparing 5 and 3: $5 > 3$ it will swap 5 and 3. Array becomes {1,2,3,5,4}

Comparing 5 and 4: $5 > 4$ it will swap 5 and 4. Array becomes {1,2,3,4,5}

Since the array has become sorted in one iteration, time complexity remains as $O(N)$.

In insertion sort:

$A=\{1,2,5,3,4\}$

Starting j from the second element i.e 2

Key element=2, $A[j-1]=1$

Comparing 2(key element) with $A[j-1]=1$,since $1<2$ so won't check for rest and will move on to the next element.

Array stays the same $\{1,2,5,3,4\}$

Key element=5, $A[j-1]=2$

Comparing 5(key element) with $A[j-1]=2$,since $2<5$ so won't check for rest and will move on to the next element.

Array stays the same $\{1,2,5,3,4\}$

Key element=3, $A[j-1]=5$

Comparing 3(key element) with $A[j-1]=5$,now $5>3$ so it will start checking for all prior elements until 3 is placed in its proper position(which will again take $O(N)$ time complexity).

Array becomes $\{1,2,3,5,4\}$

Key element=4, $A[j-1]=5$

Comparing 4(key element) with $A[j-1]=5$,now $5>4$ so it will start checking for all prior elements until 4 is placed in its proper position(which will again take $O(N)$ time complexity).

Array becomes $\{1,2,3,4,5\}$

So,insertion sort becomes $O(N^2)$ in this case.

3 Bubble Sort 10 / 21

✓ + 3 pts For part (g) - Will be correctly compensated by other deductions.

✓ - 4 pts Missing/incorrect solution for c

✓ - 3 pts Missing/incorrect solution for d

✓ - 2 pts Missing/incorrect solution for e

✓ - 3 pts Missing/incorrect solution for g

✓ - 0 pts Correct

✓ - 0.5 pts Missing termination condition for loop invariant in part (b)

✓ - 0.5 pts Missing base case condition for loop invariant in part (b)

✓ - 0.5 pts Missing maintenance condition for loop invariant in part (b)

✓ - 0.5 pts Logic error in part (a)

2 Same loop variable used.