

CS 212 – DESIGN AND ANALYSIS OF ALGORITHMS

HW 1

George Washington University

Question 1

2

(Study/Review of the principle of mathematical induction) (PMI). Prove the following using PMI:

▣ $1 + 2 + 3 + \dots + n = n(n+1)/2$

▣ $1^3 + 2^3 + 3^3 + \dots + n^3 = [n(n+1)/2]^2$

Question 2

3

Asymptotic Big O Notation: Note the following definition. $f(n) = O(g(n))$ if there exists n_0 and a constant k such that $f(n) \leq k g(n)$ for all $n \geq n_0$. To prove that a function is Big O of function g , it suffices to find constants n_0 and k , and then to prove that the inequality holds. The goal for this exercise is to find the constants n_0 and k , and then to explain in 1-2 sentences why the inequality holds. (Your proof should be slightly more than “Clearly!” or “Obviously.”)

- $4n^3 + 2n^2 + 17n + 40 = O(n^3)$
- $(1 + 2 + \dots + n) + (1^2 + 2^2 + \dots + n^2) = O(n^3)$
- $\log(1) + \log(2) + \dots + \log(n) = O(n \log(n))$
- $\log(1^2) + \log(2^2) + \dots + \log(n^2) = O(n \log(n))$

Question 3

4

- In class, we discussed a simple algorithm (using a simple for loop) to find the largest number given an array (or list) of numbers. Here is the algorithm in its full glory.

```
X = a[0]
For int j = 1 to n-1 do
    If (X < a[j]) then X = a[j] End If
End For
```

- The algorithm does $(n-1)$ comparisons, where n is the size of the array (or list).
- Question: Is it possible to find the largest number given n numbers using LESS THAN $n-1$ comparisons. Either give a **proof** that it is not possible, or provide an algorithm uses less than $n-1$ comparisons.
- In some cases it helps to consider a corresponding formulation in terms of tennis tournament. Is it possible to find the champion using less than 127 matches, when tournament consists of 128 players?

Question 4

5

- Given n numbers, propose an efficient algorithm to find both the minimum and the maximum numbers. Given 100 numbers, how many comparisons does it do?