COMP250
Assignment 2

Simon Foucher
260 223 197

Question 1 – (30 points) – Asymptotics
a) Prove that (n+10)^1.3 +n+1 is O(n^1.3)

(n+10)^1.3 +n+1 is O(n^1.3) implies that there exists a value n_o such that for some constance c > 0; for any given n > n_o; (n+10)^1.3 +n+1 < c * n^1.3

So we assume that
(n+10)^1.3 +n+1 <= c * n^1.3

Observing the (n+10) term, As n increases, the fix contribution of 10 gets diluted more and more, such that for really high values of n, (n + 10) ~ n
(Ex: n = 1,000,000,000 = 1 x 10^9 and n+10 = 1,000,000,010 ~ 1 x 10^9)

Since (n+10)^1.3 > n ^ 1.3

We can transform
(n+10)^1.3 +n+1 <= c * n^1.3

to
(n)^1.3 +n+1 < c * n^1.3
without violating the inequality

Then, since we chose n_o > 0, we can divide both sides to get
1 + n^(-0.3) + n^(-1.3) < c

As n increases, the left hand side will decrease since all n present have negative powers. The right hand side, c, stays constant, therefore the inequality holds.

b) Prove that n^3 is not O(n^2)

For that to be true, we would need
n^3 <= c * n^2

Since n > 0

we can divide by n^2 and get

n <= c

We can never bound n by a constance c, no matter what the choice of that constance is, therefore the inequality can not hold for any c and all n > n_0

c)Prove that (n log(n)) is Ω(log(n!))
By definition, we need
n * log(n) >= c * log(n!)
log(n^n) >= log((n!^c))
n^n >= (n!)^c

Base case when n = 1
1^1 >= (1!)^c
Choosing c = 1 makes the inequality hold.

General case:
k^k >= k!

Next case k+1:
(k+1)^(k+1) >= (k+1)!
(k+1)(k+1)^k >= (k+1)(k!)
(k+1)^k >= k!

Since k^k >= k!, we can make the following substitution without violating the inequality
(k+1)^k > k^k
k+1 > k

Done!

Question 2 – (30 points) - Recursion

Example where N = 4 and the median is 3:

A = {2, 3, 7, 9}
B = {0, 1, 4, 9}
C = {0, 1, 2, 3, 4, 7, 9, 9}

a) (20 points) Write a recursive algorithm, with running time O(log(N)), that solves this problem.

Algorithm: Observe middle element of each arrays (if even size array, take average). Compare both values. Throw out upper half of array with greater value and lower half of array with smaller value. Once you get 2 arrays containing a single element, pick the smallest one.

Example
{1,2,3,7,9}{0,1,4,9,10} ... Answer: (0,1,1,2,**3**,4,7,9,9,10)

{1,2,3,7,9}{0,1,4,9,10}

Observe 3 < 4
{1,2,**3**,7,9}{0,1,**4**,9,10}

{3,7,9}{0,1,4}

Observe 7 > 1
{3,**7**,9}{0,**1**,4}

{3,7}{1,4}

Observe (3+7)/2 = 5 < (1+4)/2 = 2.5
{3,7}{1,4}

{3}{4}
Pick 3
{3}

b) (10 points) Write the recurrence relation for your algorithm. Show that the running time is indeed O(log(N)) by deriving an explicit formula for the recurrence relation. Use the substitution method.

t(n) = c + t(n/2)
Set n = 2^k; k = log(n)
t(n) = c + t(2^(k-1))
t(n) = c + c + t(2^(k-2))
t(n) = c + c + c + t(2^(k-3))
...

t(n) = c + c + ...[k-1 times]... + t(k-k)
t(n) = c * k
t(n) = c * log(n)