Homework 1

Salim M'jahad
msm2243

1 - list of functions from the lowest growth rate to the highest:

- 2 / N
- 37
- $\sqrt{N}$
- N
- N*log(log(N))
- $N * log(N) \equiv N * log(N^2)$
- $N * log^2(N)$
- $N^{1.5}$
- N^2
- $N^2 * log(N)$
- N^3
- $2^{N/2}$
- 2^N

2 - a - the sequence in dollars for the fine on day N is $v_N = 2^{2^{N-1}}$

b - to reach D dollar we do the inverse which is log base 2 of log base 2 $N = log_2(log_2(D))$

3 - a - for this program fraction an addition and an assignment of value will be done for n times, and all that will be done for 23 times. Thus, it will take 2*N*23 TU (Time Units). O = N

b - for this program fraction an addition and an assignment of value will be done for - $\sum_{i=1}^{N} i \approx N^2/2$ times. As seen in the book, we consider the worst case for the Big-Oh. O = N^2

c - this program fraction uses recurrence. Every time the method is called the number is split into k parts and the program returns 1 when n <= k. The method is called around $log_k N + constant$ thus O = log(N)

4 - a. 2.5 ms

b. 2.5*log(500)/log(100) = 3.37

c. 12.5 ms

d. 62.5 ms

5 - since the A_i are ordered, the array is sorted. We can use binary search to find i (which will be the position of $A_i + 1$). The running time of the algorithm is O(logN)