Oblig 4

**Oblig 4: Graph Algorithms, Algorithm Analysis, and Approximation Algorithms Part 1: Algorithm Analysis 1. The Big-O notation is used to measure the running time of an algorithm. It can tell us how complex a given algorithm is. Explain what the following Big-O running time means**

**a. O(n)**

Goes through the entire list

**b. O(n log n)**

Goes through the entire list and then halves it for each iteration

**c. O(2n )**

Goes through the entire list twice

**d. O(log n)**

Halves it for each iteration

**e. O(n!)**

Having to go through the all the last steps to be able to go through the current one

**f. O(1)**

Instantly gets an output

**g. O(n2 )**

Goes through the entire list twice

**2. You are given two algorithms to solve a computational problem. One runs in O(log n) time and the other runs in O(n log n) time. Which of them will you choose? Provide reasons for your choice.**

O(log n) because its far more efficient and spends less time for each iteration.

**3. Use the algorithm below to answer questions (a) – (d). Use the Big-O notation when presenting the efficiency of the algorithm.**

1. This algorithm takes n amount of index array with the value 0 to n-1 and finds the minimum and maximum value in the array.
2. It makes the first value as max and min and check if the next is bigger than max or smaller than min and put the new value in if the checking is true. It goes through the whole array and finds the minimum and maximum value.

**a. What does this algorithm compute?**

Checks the input and compares it to the min-value or max-value. If it’s smaller than the min-value then it becomes the min-value. If it’s bigger than the max-value then it becomes the max-value. At the end the you do the operation (max-value)-(min-value) and get its output of off the algorithm.

**b. What is its basic operation?**

Comparisons

**c. How many times is the basic operation executed?**

2n times

**d. What is the efficiency class of this algorithm?**

O(n)

**Part 2: Graph and Approximation algorithms You are provided with a graph data structure in Java that implements the adjacency list representation. The graph implementation already contains important operations such as addNode, removeNode, addEdge, removeEdge, etc. Your tasks**

**1. Implement the Depth First Search (DFS) algorithm. You should implement your solution using either the recursive technique or non-recursive (stack).**

**2. Implement the Approximate Vertex Cover algorithm. Use the unit tests contained in the test package to test the correctness of your implementations before submission. For the DFS, uncomment the test for the method you have not implemented to avoid failed test result. Goodluck! 68 Fundamentals of the Analysis of Algorithm Efficiency**