Design and Analysis of Algorithms	Due: 26th Feb, 2010
Homework	
Instructor: Piyush Kumar	*Problem Set 3*

Write in your own words as far as possible, give proofs of correctness of the algorithms whereever needed. Each student MUST submit his own copy written in his own words. Discuss the problems with your partner and then write the answers in your own words. This homework will be due BEFORE class on the date posted on the website. Late submissions will not be accepted.

Exercise 1

Consider the BFS algorithm presented in class. Given a digraph G = (V, E) and a starting vertex s ϵ V , this algorithm computes for each vertex u ϵ V the value d[u], which is the length (number of edges) on the shortest path from s to u. For purposes of robust network communications, it is sometimes useful to know the number of shortest paths. The objective of this problem is to modify the BFS algorithm of class to compute the number of shortest paths from s to each vertex of G. We will do this in two steps.

1. First run the standard BFS on G, starting from s. Explain how to use the result of this BFS to produce a new digraph G' = (V' , E'), where V' $\subseteq$ V and E' $\subseteq$ E, such that every path in G' that starts at s, is a shortest path in G starting from s, and conversely, every shortest path in G starting from s is a path in G' .
2. Explain how to take the result of part (a) to compute for each vertex u ϵ V' , a quantity c[u], which is the number of paths in G' from s to u. (Hint: This can be done by a modification of BFS.) Both of your algorithms should run in O(V + E) time.

Hint: Be careful in how you compute c[u]. In general, the number of shortest paths from one node to another can be exponential in the size of V . (You might think about why this is.) This means that your algorithm cannot generate the paths one by one. Rather, it must employ some method that can count up many paths at once

Exercise 2

We will do a peculiar experiment for this problem. We will toss a coin. If we get heads, we will stop. Otherwise, we will keep tossing till we get a head. You are required to calculate the expected number of tails for this experiment.

Exercise 3

Unit Length Intervals
Describe an efficient algorithm that, given a set $\{x_1, x_2, x_3, \ldots, x_n\}$ of points on the real

line, determines the smallest set of unit-length closed intervals that contains all of the given points. Argue your algorithm is correct.

Exercise 4

Sorting
Describe an algorithm, that given n integers in the range 0 to k, preproceses its input and then answers any query about how many of the n integers falls into a range $[a..b]$ in $O(1)$ time. Your algorithms should use $\Theta(n + k)$ preprocessing time.