BLG335E, Analysis of Algorithms I, Fall 2016, Project 2
Handed Out: 21.10.2016
Due: 04.11.2016

Part A. Probabilistic Analysis (20 pts)

1) Suppose that a malicious employee at the post office opened the envelopes that would to be delivered by the mailmen that day and took the letters out of them. Then, he placed these letters into envelopes at random. Let n be the total count of mails delivered that day. What is the expected number of correctly delivered mails? Use indicator random variables to solve the problem.

2) Assume that an n-bit integer takes any value in range 0 –2^{n-1} with equal probability. What is the general formulation for the expected value which shows the position of leftmost "1" bit? Explain why it is different from the hiring problem given in the slides.

Hint: For n=5, let X be the random variable as defined in the problem. P[X=3]=P[{00010}]+ P[{00011}]. Remark that the leftmost bit of the integer is in the 0th position.

Part B. QuickSort (80 pts)

Using the plain text obtained from Herman Mellville's(1819-1891) famous novel Moby Dick, "mobydick.txt" were designed by randomly swapping each line of the original text and then randomly swapping each word in every line. Every line in mobydick.txt is defined as follows:

OriginalLineID {word1_word2_....._wordn} {wordorder1_wordorder2_..._wordordern}

1)(50 pts) Implement the Quick Sort algorithm as defined in slides to sort every line by their original line IDs. (It is important to select the pivot value as shown in the slides).

Define a counter which counts how many times the quicksort function is called. Increase this counter at the start of the function before the operations. Let M and N be values defined by the user. You should print the original line ID of the Nth element of the array for Mth quicksort operation. For example let M=100 and N=500, the printed value will be "353". An example usage is given below:

```
int quicksort_counter = 0;

void quickSort(…)
{
	quicksort_counter++;

	if (quicksort_counter == M)
	{
		//print the ID of Nth element
	}
	…
	…
```

2)(15 pts) After correctly ordering the lines in the previous part, to order the words in each line, use Randomized Quick Sort which selects the pivot element randomly. Then, save the obtained text in a

file named “novel.txt”. Instead of seeding the random number generator with instant time, use `srand(250)`. **Do this feeding before starting to sort each line.** Remember that different compilers have different random number generation procedures. Notice that, g++ compiler should create the first random number as `782206782`. Report the steps done for sorting Line 551 of the text which contains the most famous sentence of the novel.

3)(15 pts) Report the average running time of quicksort algorithm by sorting K sized arrays containing elements from “numbers_to_sort.txt”. Save the sorted array in a text file named “numbers.txt” Comment on the results.

Some Important Points

Your program should run from the command line with one of the following formats:

./studentID_AoA1_P2 1 M N –For first and second question.

./studentID_AoA1_P2 2 K – For the third question.

All your code must be written in C++ using object oriented approach and it should compile and run on Linux using g++.

If you have any questions, you can contact Res. Asst. Yusuf Hüseyin Şahin (sahinyu@itu.edu.tr).