22C:44 Homework 5

Due by 5pm on Thursday, 4/5

The first problem is worth 40 points and the second is worth 10 points.

1. This is a programming exercise whose purpose is to compare the running of different algorithms for the selection problem. You can use any platform and any programming language for your implementation. However, you are responsible for knowing enough about the programming environment of your choice to complete this assignment.

   - Write three different functions for the selection problem, called `SortAndSelect`, `RandomSelect`, and `Select`. A brief description of each is given below.

     `SortAndSelect` sorts the input first by using a worst case $\Theta(n \lg n)$ time sort algorithm such as `HeapSort` and then returns the kth element from the sorted array.

     `RandomSelect` is the randomized selection algorithm described in class and in the text book. This runs in expected $\Theta(n)$ time and in worst case $\Theta(n^2)$ time.

     `Select` is the worst case $\Theta(n)$ tine selection algorithm described in class and in the book.

   - To examine the running time of the functions, you will need to generate two "types" of input. A brief description of each input type and how to generate it, is given below.

     Random input: Generate a random sequence of numbers using the following code.

     ```
     RandomPermutation (A[1..n]) {
          for i ← 1 to n-1 do {
               j ← Random(i, n);
               swap(A, i, j);
          }
     }
     ```

     In the above code, `Random(i, n)` returns a random integer in the range `i` through `n`. The above code randomly permutes the elements in the array `A[1..n]`. So if we initialize the array `A[1..n]` to be the sequence $1, 2, \ldots, n$, and call `RandomPermutation` we get a random permutation of the first n natural numbers.

     Almost sorted input: Start with the sorted input $1, 2, \ldots, n$ in an array `A[1..n]` and swap a randomly chosen element with the element immediately after it. In particular, swap $\lfloor n/10 \rfloor$ times.

   - Run each of the three algorithms on inputs of each of the two types. Do this for input sizes $1000, 2000, 3000, \ldots, 10,000$. Generate 10 inputs for each size and record the average running time (over the 10 inputs of that size). So you would need to perform a total of $3 \times 2 \times 10 \times 10 = 600$ runs and generate a total of 60 running times. Use the functions to determine the median of the input. Tabulate your results separately for each input type, plot the running times, and explain your results. In particular, for each input type compare the running times of the three algorithms against each other and explain your results. Also, for each input type compare the running times against what is theoretically predicted and explain your results.

     Finally, pick your favorite selection algorithm from among the three and explain your choice.

2. Suppose that the universe contains the million numbers 1 through 10^6. Further suppose that the probability distribution P telling us the likelihood of an element in the universe being chosen into S is as follows:

$$P(k) = \begin{cases} 1/10^7 & 1 \leq k \leq 10^4 \\ 1/9 \times 10^7 & 10^4 < k \leq 10^5 \\ 1/9 \times 10^8 & 10^5 < k \leq 10^6 \end{cases}$$

Given this probability distribution, prove or disprove the following claim: $h(k) = k \bmod 1000$ is a simple uniform hashing function.
