CS 5633 Analysis of Algorithms – Spring 09

2/5/09

4. Homework

Due **2/12/09** before class

Always justify your answers. All algorithms should be as efficient as possible and all runtimes should be as tight as possible.

1. **Selection sort decision tree (5 points)**
   The decision tree on the slides in class was for insertion sort of three elements. Draw the decision tree for selection sort for three elements. Use the pseudo-code for selection sort that was given in the solutions for homework 1, and annotate the decision tree with comments about the location in the pseudocode.

2. **Randomized code snippets (5 points)**
   Consider the following code snippet, where `RandomInteger(i)` takes $O(1)$ time and returns an integer between 1 and i, each with probability $1/i$.

```
for(i=2; i<=n; i++){
  if(RandomInteger(2)==1){
    for(j=1; j<=n; j++){
      print(‘‘hello’’);
    }
  }

  if(RandomInteger(i)==i){
    for(j=1; j<=n; j++){
      for(k=1; k<=n; k++){
        print(‘‘hello’’);
      }
    }
  }
}
```

   a) What is the best case runtime, in terms of n, of this code snippet? Describe what triggers a best-case scenario.

   b) What is the worst case runtime, in terms of n, of this code snippet? Describe what triggers a worst-case scenario.

   c) Now analyze the **expected** runtime. Clearly define your random variable. *Hint: Break your random variable into multiple random variables.*

FLIP OVER TO BACK PAGE ⟹

3. **Radix sort with most significant digit first (5 points)**
   Try to sort the numbers

   $$852, 117, 505, 143, 156, 856, 806, 145, 555, 588, 112$$

   using radix sort but starting with the **most** significant digit (so, from left to right, not from right to left).

   Why would a program that implements this strategy be much more complicated than the radix sort that starts with the least significant digit? (Hint: What kind of variables or data structures would you have to maintain?)

4. **Almost best case (2 points)**

   Consider (deterministic) quicksort which takes the first array-element as the pivot. Design an example input of 100 numbers which causes quicksort to always split $\frac{1}{3} : \frac{2}{3}$ in each recursive call. (Round fractions as necessary.)

5. **Radix Sort vs. Insertion Sort (2 points)**

   Given n numbers between 0 and n^n. Which sorting algorithm is faster: Insertion Sort or Radix Sort? Justify your answer.