Panther ID (No name, please):

COP 6405: Analysis of Algorithms MIDTERM EXAM; Spring 2019

Max Score = 75 pts; 4 pages; Max Time = 75 minutes;

Read This: This is a "closed book" exam and has a total of 4 pages. To maximize partial credit, write down the basic ideas behind your solution and fill in the details later, if time permits. However, the heart of the problem is in the step that gives the recurrence relation with a clear definition of what the notation means. Analyze the worst-case time complexity of all algorithms you are asked to design.

---

1. [38] Mr. A. Mazing is a waiter in a busy restaurant with an amazing memory. When he sees a customer more than once, he remembers precisely how much that person spent at the restaurant on the previous visit. Unfortunately, Mr. Mazing usually forgets the items that person ordered from the menu. Your goal is to design a *dynamic programming* algorithm that takes as input the restaurant menu with prices and the total money (\$$M$) spent by a customer, and reports what items that customer and his/her table ordered from the menu. You may assume that this restaurant includes all taxes and tips in the menu prices and are therefore not added separately. You may also assume that all prices are integral (rounded off) and therefore the total amount paid by any customer is also integral. If there is more than one combination of items that results in the same amount of money spent, the algorithm may report any one. The menu has n items denoted by $\{X_1, X_2, \ldots, X_n\}$ with prices $\{p_1, p_2, \ldots, p_n\}$.

   Write down the basic idea behind your solution in plain English. Write down your recurrence relation and clearly explain every variable used in that recurrence relation. Analyze the worst-case time complexity of your algorithm.

   Notes: If the customer orders r_i plates of item X_i, then the cost is $r_i * p_i$ on item X_i from the menu. If the customer orders $\{r_1, r_2, \ldots, r_n\}$ plates of the n items on the menu for the table, then the total cost of the order is given by $\sum_{i=1}^{n} r_i * p_i$. Thus your job is to find the number $\{r_1, r_2, \ldots, r_n\}$ representing the number of orders of each of the n items on the menu such that:

   $$M = \sum_{i=1}^{n} r_i * p_i.$$

   Hints: It may help you to write down a recursive solution for this problem and identify the subproblems. Assume that a customer spends a total of \$$M$ at the restaurant. Imagine storing a 2-dimensional array $C[i, B]$ defined as follows: $C[i, B] = 1$, if there exists some set of orders from the items $\{x_1, x_2, \ldots, x_i\}$ (i.e., only the first i items from the menu that adds up to B. Otherwise $C[i, B] = 0$. If the menu has only 1 item, x_1 with price p_1, what are the possible values of B for which $C[1, B] = 1$. Figure out which entry from the array C you need to fill in order to solve the problem. Next write down a recurrence relation to solve the problem. Finally, provide pseudo-code to solve the problem and analyze.

[Blank page.]

2. [37] You have conducted a large phone poll in which you phoned a large number of randomly selected individuals and asked them to rate (on a scale of 0-10) how strongly they believe that *"Artificial Intelligence" is more important in our lives than "Emotional Intelligence"*. A rating of 0 meant that they completely disagreed, a rating of 10 meant that they totally agreed, and all other values indicated a graded response in between. You have kept records of each of the calls, recording the time you talked to the individual, their rating on the question, and their gender (male/female/undisclosed). You may assume that you have records on n phone calls.

   The data analyst from your company is performing an experiment in which she needs to pick a large number of random time intervals (i.e., a random start time and a random finish time) and compute the magnitude of the difference between the average rating of the men and the average rating of the women polled during this time interval. Show how this can be facilitated by using an appropriately designed *augmented RB-tree*. Besides the standard queries to the RB-tree, your data structure should allow for fast implementations of a query that computes the "magnitude of the difference between the average rating of the men and the average rating of the women polled in a given time interval".

   You may assume that the time of collecting a person's rating is unqiue (i.e., no two people were polled at the same time). If during a given time interval, the ratings polled (along with gender M/F) were 3(M), 7(F), 5(F), 3(M), 3(M), then the required difference is $|3 - 6| = 3$. Describe the design by answering the three questions below. You do not need to provide any pseudocode other than what is asked below in (c).

   (a) [5] State what **augmented** piece(s) of information you will store in each node (besides KEY, LEFT, RIGHT, COLOR) in order to facilitate the analyst's queries.

   (b) [2] Explain very briefly why this information can be maintained in the presence of inserts and deletes in the RB-tree.

(c) [30] Write down an efficient algorithm for answering the analyst's queries using the augmented fields suggested by you in (a) above. Analyze the time complexity of your algorithm. The code for OS-RANK is provided for your benefit and may be used as you please. If you wish you can directly edit the code printed below.

OS-RANK(x, y)
▷ Finds rank of node x in subtree rooted at node y
▷ Recursive version discussed in class
▷ RB tree stores KEY, LEFT, RIGHT, COLOR in each node
▷ Every node is augmented with SIZE information

$r \leftarrow \text{SIZE}[\text{LEFT}[y]] + 1$
if $(x == y)$ **then return** r
else if $(\text{KEY}[x] < \text{KEY}[y])$ **then return** OS-RANK$(x, \text{LEFT}[y])$
else return r + OS-RANK$(x, \text{RIGHT}[y])$
end if
