2020 Algorithms Design Final Exam

It's a closed book and notes exam. You have 90 minutes.

Question	Points	
1	25	
2	15	
3	15	
4	15	
5	15	
Total:	70	

Arithmetic series: $\sum_{k=1}^{n} k = \frac{1}{2}n(n+1)$

Geometric series: $\sum_{k=0}^{n} x^k = \frac{1-x^{n+1}}{1-x}$

Harmonic series: $\sum_{i=1}^{n} \frac{1}{i} = \ln n + O(1)$

1. Short-answer questions.

Indicate whether each of the following statement is true or false. (15 points)

a) All running time recurrences can be solved by using the Master Theorem.

b) The best-case running time of A is $\Omega(g(n))$ implies the running time of A is $\Omega(g(n))$.

c) The worst-case running time of Merge-sort is $\theta(n^2)$.

d) When we implement a dynamic programming algorithm in a top-down manner, there may be some subproblems that are never calculated.

e) The lower bound of a problem is defined by its known best algorithm.

Answering the following questions with short answers. (10 points)

f) Solve the following recurrence using the recursion tree method. (5 points)

$$T(n) = 2T\left(\frac{n}{3}\right) + c$$

$$T(1) = d$$

where both c and d are constants.

g) Describe the three steps of the Loop Invariant Analysis. (5 points)

You can choose three problems from problem 2 to problem 5.

2. **Input-Output Problem (15 points)**

$$z_{i,u} = \begin{cases} 0 & if\ u \le 0 \\ z_{i-1,u} & if\ u < w_i \\ \max\left(z_{i-1,u}, z_{i-1,u-w_i} + v_i\right) & otherwise \end{cases}$$

The knapsack recurrence is given above. Suppose we have a knapsack capacity of 4 and four items a_1, a_2, a_3, a_4 with weights [2, 1, 3, 1] and values [15, 10, 20, 8], respectively, fill in the table of $z_{i,u}$ and give the final optimal total value.

3. Probability Analysis (15 points)

Bucket-Search: Suppose $A[1..n]$, where $A[i] \in [0,1)$ and distributed uniformly, has been distributed to buckets $B[0..n-1]$, such that $A[i]$ is inserted to bucket $B[\lfloor nA[i] \rfloor]$. Write a search algorithm (either describe the algorithm or give pseudo-code) for searching a value $x \in [0,1)$ in $A[1..n]$ using the buckets (you can assume $A[1..n]$ already in the buckets). Derive the expected running time of your Bucket-Search using indicator random variables.

4. Dynamic Programming (15 points)

Consider a modification to the activity-selection problem in which each activity a_i has, in addition to a start and finish time $[s_i, f_i)$, a value v_i. The objective is no longer to maximize the number of activities scheduled, but to maximize the total value of the activities scheduled. That is, we wish to choose a set A of compatible activities from $S = \{a_1, a_2 \ldots, a_n\}$, such that $\sum_{a_k \in A} v_k$ is maximized.

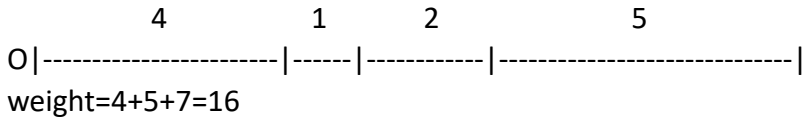
Give the recurrence when using dynamic programming to find the optimal set of activities, and describe the optimal substructure in this problem (**no need to prove**).

Hint: you can use $S_{i,j}$ to denote the activities in S that start after activity a_i finishes and finish before activity a_j starts.

5. Greedy Algorithm (15 points)

Let S = {s_1,…,s_n} be a set of n intervals on the x-axis. Let L_i be the length of s_i, $1 \le i \le n$. A packing of S is a placement of the intervals end to end, in any order, such that there are no gaps between the intervals and no two intervals overlap (see the figure below, where the numbers shown above each interval are the *length*, assuming the packing begins at the origin O).

```
        2               4           1               5
O|------------|-----------------------|------|-----------------------------|
weight=2+6+7=15
```



weight=4+5+7=16

The weight of a packing is the sum of the x-coordinates of the left endpoints of the intervals. For instance, in the first packing, the x-coordinates of the left endpoints of the four intervals are 0, 2, 6, and 7, which gives a weight of 15.

Please give a greedy strategy of packing to minimize the weight. And prove your strategy does satisfy the greedy-choice property.