Programming Quiz – COSC 6334.001

School of Engineering & Computer Sciences
A & M University at Corpus Christi
Dr. Minhua Huang
Out: Oct 15, Due: Nov 15

October 15, 2019

Objective:

Practices on Design, Implement, and Analyze Algorithms.

Task

- Design, Implement [1], and analyze algorithms for one of the following problems.
- Report your design on
  - Describing your algorithms, such as data structures, the functionalities, the properties, ...
  - Analyze the time complexity of each of the algorithms.
  - The report needs to have at least two pages.

Problem Set

1. **Problem 1.** Design greedy algorithm(s) for the ACTIVITY-SELECTION problem (chapter 16) satisfying following requirements:
   (a) Each activity is an object which has attributes of a starting time and finishing time (You might have more....)
   (b) Design a **minimum priority queue** data structure for the finish times of input activities
   (c) Design an algorithm to compute the maximum-size subset of mutually compatible activities.
   (d) Describe the greedy-choice property of your algorithm(s)
   (e) Describe the optimal substructure of your algorithm(s)
   (f) Determine the time complexity of your algorithm(s)
   (g) Test your design by randomly generate a set of n activities.

2. **Problem 2.** Design a dynamic programming algorithm(s) for the ACTIVITY-SELECTION problem (chapter 16) based on the given recurrence satisfying the following requirements.

$$c[i,j]\ ^{2} = \begin{cases} 0 & \text{if } S_{ij} = \phi \\ max_{a_k \in S_{ij}}\{c[i,k] + c[k,j] + 1\} & \text{if } S_{ij} \neq \phi \end{cases}$$

   (a) Each activity is an object which has attributes of a starting time and finishing time (You might have more....)
   (b) Design an algorithm to sort finish times of input activities on the time complexity of O(n $\log_2$ n) .
   (c) Design an algorithm to compute c[i,j] as well the maximum-size subset of mutually compatible activities.
   (d) Determine the time complexity of your algorithm(s)
   (e) Test your design by randomly generate a set of n activities.

[1] You can use any programming language

[2] c[i,j] is the size of an optimal solution for $S_{i,j}$ and $S_{i,j}$ is the set of activities that start after activity a_i finishes and that finish before activity a_j starts.