WRIGHT STATE UNIVERSITY   
Department of Computer Science and Engineering

**CS7200: Algorithm Design and Analysis**

Spring 2022 **Assignment 1 (Due: Feb 15) (5 pts)**      Prasad

1. [GRADED] Implement a program for determining whether an assignment of women to men is *stable* or not as discussed in class. For this assignment you must use Python 3 or higher to implement your algorithm as it is relatively straightforward. Further, the goal is to help you understand the notion of stability, so make sure to test it adequately. For an efficient implementation, minimize the number of “*potential* cause of instability” checks, that is, the number of (woman, man) pairs to be checked to ensure that the solution is (un)stable. Name your main Python scriptas **assignment1.py**,and make sure your code is well-documented**.**
2. Assume that the input file (stableInput.txt) is a sequence of (2 + 2\**n* + n\*m)-lines, with (i) *n* in the first line on its own signifying the number of women and men, (ii) *m* in the second line signifying the number of assignments to be tested, (iii) the following (*2\*n*) lines encoding the preference lists – *n* lines for women and *n* lines for men – each containing space separated list of (*n+1*) names as required in the stable marriage problem, and (iv) and *n*\**m* lines encoding *m*-assignments, each assignment given as *n*-lines, each line containing a woman-man pair, as follows:

For example:

2

3

Alice Xavier Zeus

Carol Zeus Xavier

Xavier Alice Carol

Zeus Alice Carol

Alice Xavier

Carol Zeus

Alice Xavier

Carol Zeus

Alice Zeus

Carol Xavier

1. Write the output as *m*-lines, each line containing three numbers, with either 1 or 0 to signify whether or not the assignment is a *stable match,* followed by *n2,* followed by *the number of unassigned pairs the algorithm checks to determine stability or rule out instability*. You must name input file, output file, and the Python program as follows: **stableInput.txt**, **stableOutPut.txt**, **assignment1.py**. Points will be deducted if you do not follow this naming convention and file extension.

For example, for the above input, *one possible output* can be:

1 4 0

1 4 0

0 4 1

1. [NOT GRADED – NOT REQUIRED TO BE TURNED IN] Do Problem 4 in Chapter 2 on Pages 67-68 of the Kleinberg and Tardos text. Be able to justify your ordering of the growth rate functions in each case.
2. **TURN IN**: Upload one tar archive file per team of upto 3 that contains the following files:

* README.txt: This document should explain your solution, how to launch your application, and any other relevant information. Additionally, all team member names and email addresses must be present in this document.
* Sample input data and generated output data from your tests: **stableInput.txt, stableOutPut.txt, stableInput0.txt, stableOutPut0.txt, stableInput1.txt, stableOutPut1.txt, stableInput2.txt, stableOutPut2.txt**, … (Include enough to convince yourself your code works!) Make **stableInput.txt** and **stableOutput.txt** the same as one of your other tests so that even if we overwrite **stableInput.txt** and **stableOutput.txt**, all your test cases are preserved.
* All source code file(s) in your working directory.
* Use the following command exactly as written to tar up your working directory:
  + **tar -zcvf assignment1.tgz yourWorkDirectoryName/**
  + If you do not know how to use **tar**, you can use **zip** as a last resort. However, for ease of grading, we strongly suggest learning about and using **tar**.

Submit your archive to Assignment 1 DropBox on Pilot by **February 15**. (Only one submission per team.)

1. **Grading Criteria:** Your assignment will be graded on correctness based on the test inputs we will provide as well as on the well-documented source code. Late submissions will be penalized at the rate of 25% / day and no submissions will be accepted after February 18 on Pilot.