Homework 0

Problem 1: (Stable Marriage Problem)

- **(i)** SMP is the problem of finding a stable matching between two disjoint equally sized sets. SMP can be presented as computationally as shown as below.
  - **Input:**
    - A set of men and a set of women which are sized equally.

      Men = $\{m_1, m_2, ..., m_n\}$ Women = $\{w_1, w_2, ..., w_n\}$

      - Ordered preference list for both sets which contains all members of the opposite sex.

        Ex: $P_{Men(i)} = \{w_8, w_{14}, ..., w_3\}$ ($P_{Men(i)}$ equally sized as Men)
        $P_{Women(i)} = \{m_8, m_{12}, ..., m_7\}$ (equally sized as Women)
  - **Output:**
    - A stable matching of partners Ex: SM = $\{(m_4, w_{14}), \ldots, (m_6, w_2)\}$
    - To have a stable match;
      - SM have to be sized as equally as Men and Women.
      - Nobody should be left out in SM.
        - $\forall_{w \in Women} \exists_{m \in Men} ((m, w) \in SM)$
        - $\forall_{m \in Men} \exists_{w \in Women} ((m, w) \in SM)$
      - There have to be no blocking pairs which means, at the end if m_x and w_x are not partners, there must be no case such as, m_x prefers w_x to $P_{Man}(w_i)$ and/or w_x prefers m_X to $P_{Women}(m_i)$.

- **(ii)**
  - **Input:**
    - A set of 3 NBA teams: T = $\{t_1, t_2, t_3\}$
    - A set of 3 players: P = $\{p_1, p_2, p_3\}$
  - **Output:**
    - A stable matching of 3 NBA teams and players. SM_{t-p}
  - **Example of input and output:**
    - **Input:**
      - T = $\{t_1, t_2, t_3\}$
      - P = $\{p_1, p_2, p_3\}$
      - $P_{player(1)} = \{t_1, t_2, t_3\}$, $P_{player(2)} = \{t_2, t_1, t_3\}$, $P_{player(3)} = \{t_1, t_2, t_3\}$
      - $P_{teams(1)} = \{p_2, p_1, p_3\}$, $P_{teams(2)} = \{p_1, p_2, p_3\}$, $P_{teams(3)} = \{p_1, p_2, p_3\}$
    - **Output:**
      - $SM_{t-p} = \{(t_1, p_2), (t_2, p_1), (t_3, p_3)\}$

<u>Problem 2</u>: (Gale-Shapley Algorithm)

- **(i)**
  - While any man is free (let's call him x),
    - he proposes to a woman he most prefers.
    - If the woman (let's call her y) is single, then they become a couple (x, y) and this pair is added to the set SM.
    - If y is paired with another man (x'), then there are two possibilities:
      - If y prefers x over x', then she breaks the pair and is coupled with x. Then, (x, y) is added to SM. x' starts proposing to his other preferences.
      - If y prefers x' over x, then x remains single and continues proposing until he is paired with a woman.
  - Return final SM as the stable matching result.

- **(ii)**
  - Let's say that the set of men and the set of women are sized as n.

  - If we go over the algorithm by using the same sets that explained at above, we can analyze the asymptotic time complexity of this algorithm.

  - In the worst case, the algorithm will iterate n times, and, in each iteration, the man will propose to a woman that he didn't propose before. This operation will iterate as many times as the while loop statement which iterates n times. In conclusion, in worst case iteration of the algorithm takes n^2 times. In big-oh notation the asymptotic time complexity of the algorithm is $O(n^2)$.

Problem 3:

```
Teams = ['Lakers','Warriors','Celtics']
Players = ['LeBron','Curry','Kyrie']

Pteams = {'Lakers':['LeBron', 'Curry','Kyrie'],
            'Warriors':['Curry','LeBron','Kyrie'],
            'Celtics':['LeBron','Curry','Kyrie']
            }

Pplayers = {'LeBron':['Lakers','Warriors','Celtics'],
            'Curry':['Warriors', 'Lakers','Celtics'],
            'Kyrie':['Lakers', 'Warriors','Celtics']
            }

for i in Pteams:
    Pteams[i].reverse()

for i in Pplayers:
    Pplayers[i].reverse()

Contracts = {}
teamsWithFreeContrats = list (Teams)
numOfContracts = len (Teams)

while teamsWithFreeContrats:
    T = teamsWithFreeContrats.pop()
    P = Pteams[T].pop()
    if P not in Contracts:
        Contracts[P]=T
    else:
        pickedTeam= Contracts[P]
        if Pplayers[P].index(T) > Pplayers[P].index(pickedTeam):
            Contracts[P] = T
            teamsWithFreeContrats.append(pickedTeam)
        else:
            teamsWithFreeContrats.append(T)

print (Contracts)
```

Output:

{'LeBron': 'Lakers', 'Curry': 'Warriors', 'Kyrie': 'Celtics'}