**Homework 0**

**Problem 1: (Stable Marriage Problem)**

* **(i)** SMPis 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 = {m1, m2, …, mn} Women = {w1, w2, …, wn}

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

Ex: PMen(i) = {w8, w14, …, w3} (PMen(i) equally sized as Men)

PWomen(i) = {m8, m12, …, m7} (equally sized as Women)

* + **Output:**
    - A stable matching of partners Ex: SM = {(m4, w14), …, (m6, w2)}
    - To have a stable match;
      * SM have to be sized as equally as Men and Women.
      * Nobody should be left out in SM.
        + ∀w∈Women ∃m∈Men ((m, w) ∈ SM)
        + ∀m∈Men ∃w∈Women ((m, w) ∈ SM)
      * There have to be no blocking pairs which means, at the end if mx and wx are not partners, there must be no case such as, mx prefers wx to PMan(wi) and/or wx prefers mX to PWomen(mi).
* **(ii)** 
  + **Input:**
    - A set of 3 NBA teams: T = {t1, t2, t3}
    - A set of 3 players: P = {p1, p2, p3}
  + **Output:**
    - A stable matching of 3 NBA teams and players. SMt-p
  + **Example of input and output:**
    - **Input:**
      * T = {t1, t2, t3}
      * P = {p1, p2, p3}
      * Pplayer(1) ={t1,t2, t3}, Pplayer(2) ={t2,t1, t3}, Pplayer(3) ={t1,t2, t3}
      * Pteams(1) ={p2,p1, p3}, Pteams(2) ={p1,p2, p3}, Pteams(3) ={p1,p2, p3}
    - **Output:**
      * SMt-p = {(t1, p2), (t2, p1), (t3, p3)}

**Problem 2: (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 n2 times. In big-oh notation the asymptotic time complexity of the algorithm is O(n2).

**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'}