Homework 0

1) i) Stable marriage problem is a matching problem which requires two different sites (site 1 and site 2) with disjoint sets of size n. Each site has their own preference list that contains all the members of the other site. The matching is called "stable" if there are no such pair both prefer having different partners than their current partners. Otherwise it is called a blocking pairs.

   Input: A set of one site of size n such that S1 = {s0, s1, …., sn}.
   A set of another site of size n such that S2 = {s'0, s'1, ….., s'n}.
   Preference list of each member from both sites. (Member which has a smaller index has a stronger preference).
   Ex: Preference of s'1 = [s4, s1, s2….] s.t s4>s1>s2….

   Output: List of the stable matching of each members of site1 and site2.
   Ex: Result:
   s'0 = {s4} s0 = {s'5}
   s'1 = {s3} s1 = {s'3}
   ……

   ii) Suppose there is a 2x2 basketball tournament and there are n tall players and n short players. Each tall player should be matched with one short player in order to be an individual team. Besides, each player has their own preference list. "Stable Marriage Problem" algorithm could be used to solve this matching problem.

2) i) Initially assign every person as a free (unengaged) person.
   While there are some man (m) who is free:
   w = the most preferable woman in the m's list that he has not proposed yet
   if w == free:
   marry w with m.
   else:
   if w prefers m to her current partner:
   set w's current partner (m') as free
   marry m with w
   else:
   w rejects m
   m is still free

   Output = stable matching of all pairs.

   ii) Since each man proposes to a specific woman only once (because if a man marry a woman and then divorce, he do not have to propose to that woman

again because this will lead a worse matching), totally a single man can make at most n proposal. For this reason, if we assume that there are n men and n women, there could be at most n^2 proposals. Each proposal could be done by constant time, by reaching the preference list by the index the people. For example, we could hold a list for each man, and when we pick a free man from the list, we could increment the value of that mans' corresponding index and we can use that value when choosing the women from his preference list. By the similar idea, we can check whether a woman prefer a man to his current partner in constant time (by looking to the list's index). So, the complexity of this problem is $O(n^2)$.

Bonus:

```
free = [0,1,2,3]
man_pref_list = [[3,2,1,0],[1,0,3,2],[2,3,1,0],[2,1,3,0]]  #men 0 1 2 3 #women 0 1
2 3
wom_pref_list = [[1,2,3,0],[2,3,0,1],[1,3,2,0],[3,2,1,0]]
next = [0,0,0,0]
matches = [0,0,0,0]
ranking = [[1,4,3,2],[2,1,4,3],[1,4,2,3],[1,2,3,4]] #ranks start from 1 2 3 4 (from
least to most)


while(len(free) != 0):
    select_man = free[len(free) - 1]
    prefer_woman = man_pref_list[select_man][next[select_man]]

    if(matches[prefer_woman] == 0):
        matches[prefer_woman] = select_man
        free.pop()

    elif (ranking[prefer_woman][select_man] <
ranking[prefer_woman][matches[prefer_woman]]):
        left_man = matches[prefer_woman]
        free.pop()
        matches[prefer_woman] = select_man
        free.append(left_man)

    next[select_man] += 1
```

Efe Şencan 25083