VIT-AP UNIVERSITY

AMARAVATI

ANDHRA PRADESH, INDIA

2021

FALL SEM 2021-22

MAT1003- Discrete Mathematical Structures

FINAL PROJECT REPORT

EVALUATED BY:

Prof. Tanuj Kumar

PROJECT REPORT ON

“CATCH A MATCH”

Submitted by:-

N.Pavan Kumar(20BCI7063)

Dev Kapadia(20BCD7051)

Shubhankar Gadad(20BCE7592)

Salome Rao(20BCE7051)

Ashutosh Patel(20BCE7053)

Venkata Bhaskara(20BCI7150)

Under the Guidance of

Prof. Tanuj Kumar

Department of Computer science and Engineering

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude and appreciation to all those who gave us the possibility to complete this report. Special thanks is towards our Professor Tanuj Kumar who gave us a golden opportunity to this wonderful project on the topic Graph theory which has also helped all of us to do a lot of research work and we came to know about so many new things.

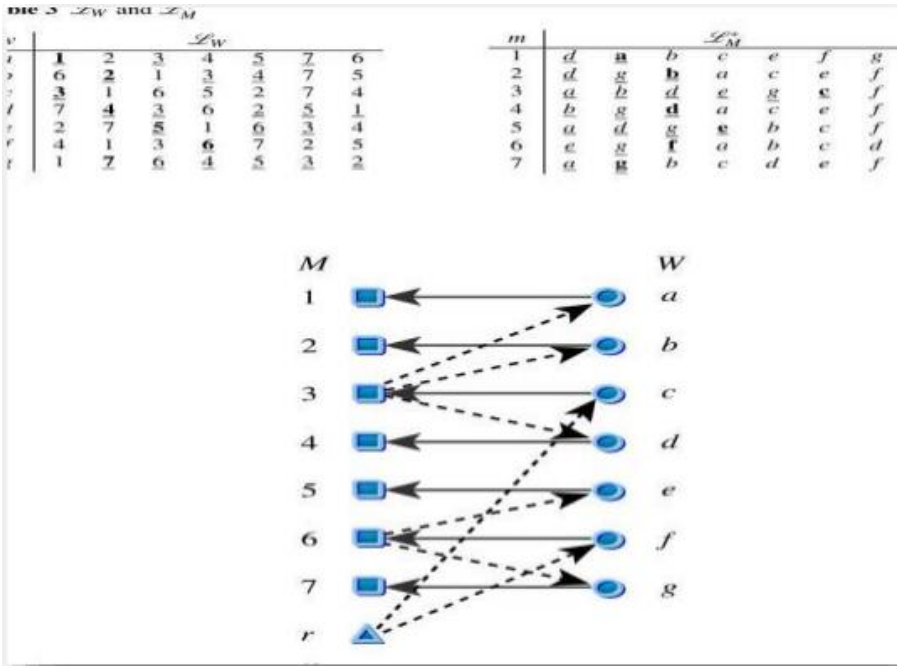
Secondly, we would also like to thank Professor Sandipan Maiti who helped us to work the code.

Finally we thank all our friends who worked with a good team spirit which helped to accomplish this project.

ABSTRACT

The main objective of our project “Catch a Match” is to provide excellent matchmaking experience by exploring the opportunities and resources to meet a true potential partner. Through this we try to provide a fast searching of numerous suitable profiles. He/she can create his/her profile and can communicate with the existing members of site. Since it can be accessed through the internet it is possible for the users to connect with people around the globe.

To find a stable match, our project uses “Gale Shapely algorithm”. The idea is to iterate through all free men while there is any free man available. Every free man goes to all women in his preference list according to the order and vice versa. For every woman he goes to, he checks if the woman is free, if yes, they both become engaged. If the woman is not free, then the woman chooses either says no to him or dumps her current match according to her preference list. So, a match done once can be broken if a woman gets better option.



It establishes a system by which everyone can find the person they most prefer from among those who prefer them. The men and women each rank their preferences.

CONTENTS

INTRODUCTION

Everyone is a part of today's fast-paced world, and we can see the major impact of technology, nearly in every field. During this pandemic, people are maintaining social distance between each other, so everybody prefers to be safe inside their homes. People are trying to connect through the medium of internet.

Our project mainly focuses on providing stable match for a person who is in search of a suitable person with his/her choice of traits. A person can have his/her choice of traits like height, weight, job status, profile pic etc. Therefore, can choose what kind of person is suitable for him / her. He / she can prioritise a particular person from a list of persons from opposite gender. Thereby they will their perfect match.

What is a Stable Match Problem?

The Stable Match Problem states that given N men and N women, where each person has ranked all members of the opposite gender in order of preference, match the men and women together such that there are no two people of opposite gender who would both rather have each other than their current partners. If there are no such people, all the matches are "stable".

A matching is a bijection from the elements of one set to the elements of the other set.

A matching is not stable if:

There is an element A of the first matched set which prefers some given element B of the second matched set over the element to which A is already matched, and

B also prefers A over the element to which B is already matched.

In other words, a matching is stable when there does not exist any match (A, B) which both prefer each other to their current partner under the matching.

Solution to Unstable Match- Gale Shapely Algorithm

In 1962, David Gale and Lloyd Shapley proved that, for any equal number of men and women, it is always possible to solve the SMP and make all matches stable. They presented an algorithm to do so.

The Gale–Shapley algorithm (also known as the deferred acceptance algorithm) involves several "rounds" (or "iterations")

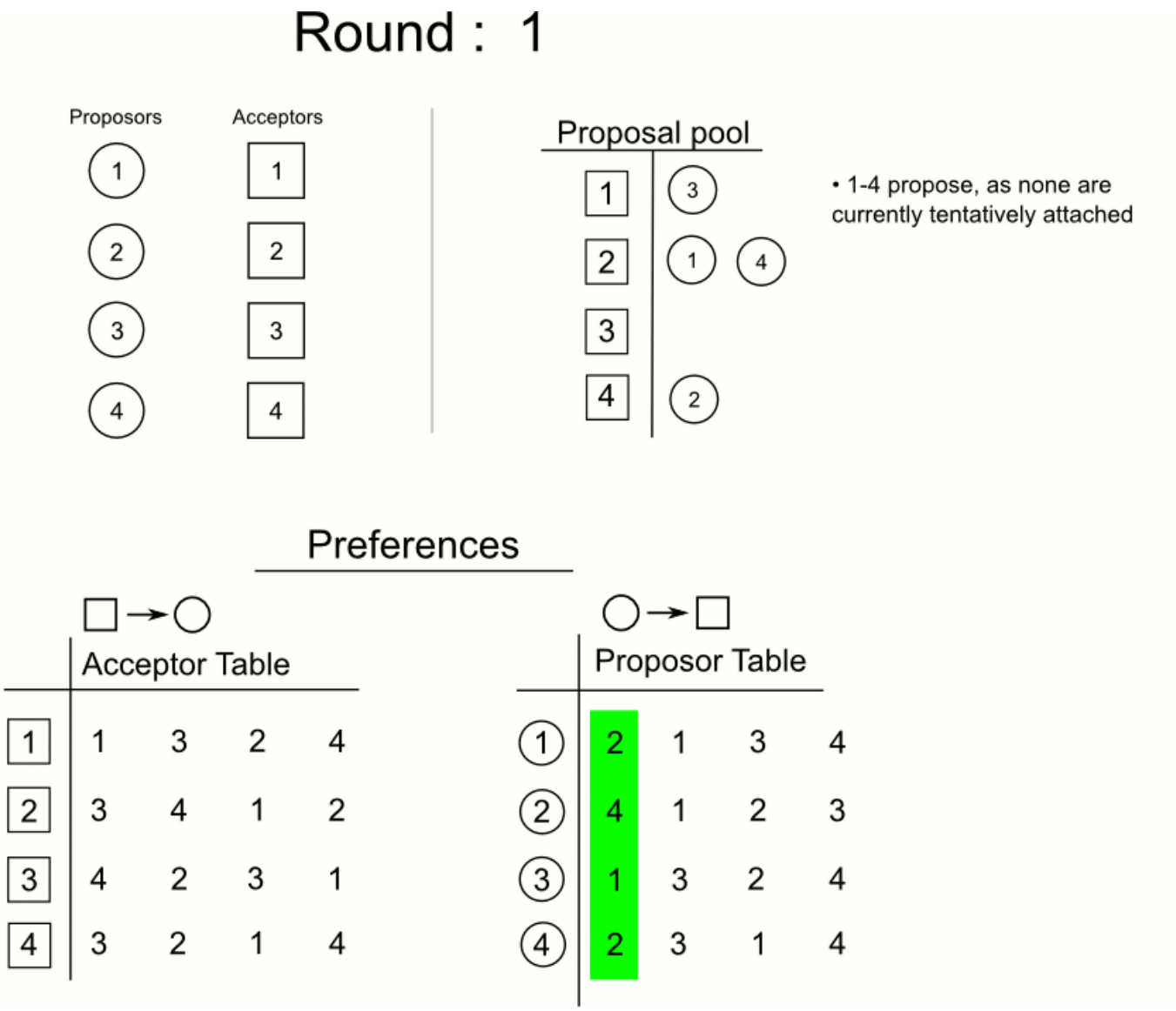
In the first round, first a) each unengaged man proposes to the woman he prefers most, and then b) each woman replies "maybe" to her suitor she most prefers and "no" to all other suitors. She is then provisionally "engaged" to the suitor she most prefers so far, and that suitor is likewise provisionally engaged to her.

In each subsequent round, first a) each unengaged man proposes to the most-preferred woman to whom he has not yet proposed (regardless of whether the woman is already engaged), and then b) each woman replies "maybe" if she is currently not engaged or if she prefers this man over her current provisional partner (in this case, she rejects her current provisional partner who becomes unengaged). The provisional nature of engagements preserves the right of an already-engaged woman to "trade up" (and, in the process, to "jilt" her until-then partner).

This process is repeated until everyone is engaged.

This algorithm is guaranteed to produce a stable marriage for all participants in time $O(n^2)$ where n is the number of men or women.

Among all possible different stable matchings, it always yields the one that is best for all men among all stable matchings, and worst for all women. It is a truthful mechanism from the point of view of men (the proposing side). i.e, no man can get a better matching for himself by misrepresenting his preferences.

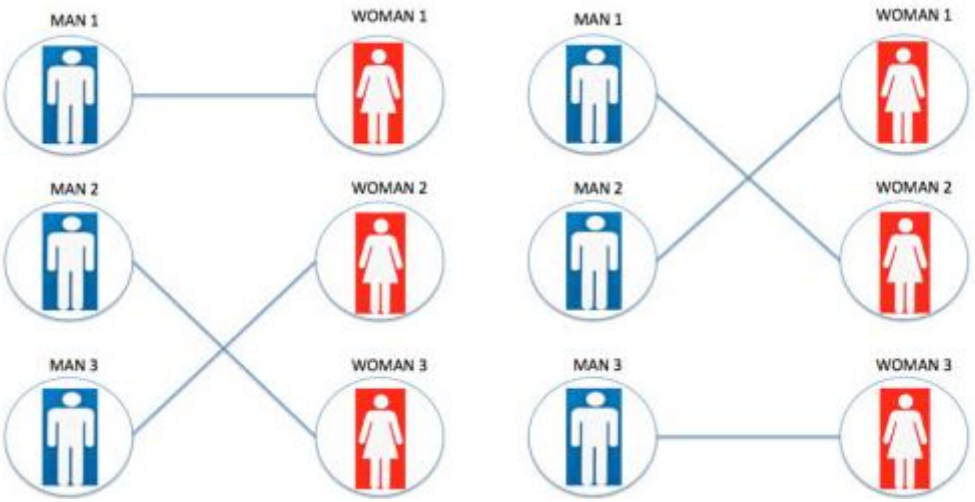


Moreover, the GS algorithm is even group-strategy proof for men, i.e., no coalition of men can coordinate a misrepresentation of their preferences such that all men in the coalition are strictly better-off. However, it is possible for some coalition to misrepresent their preferences such that some men are better-off, and the other men retain the same partner. The GS algorithm is non-truthful for the women (the reviewing side): each woman may be able to misrepresent her preferences and get a better match.

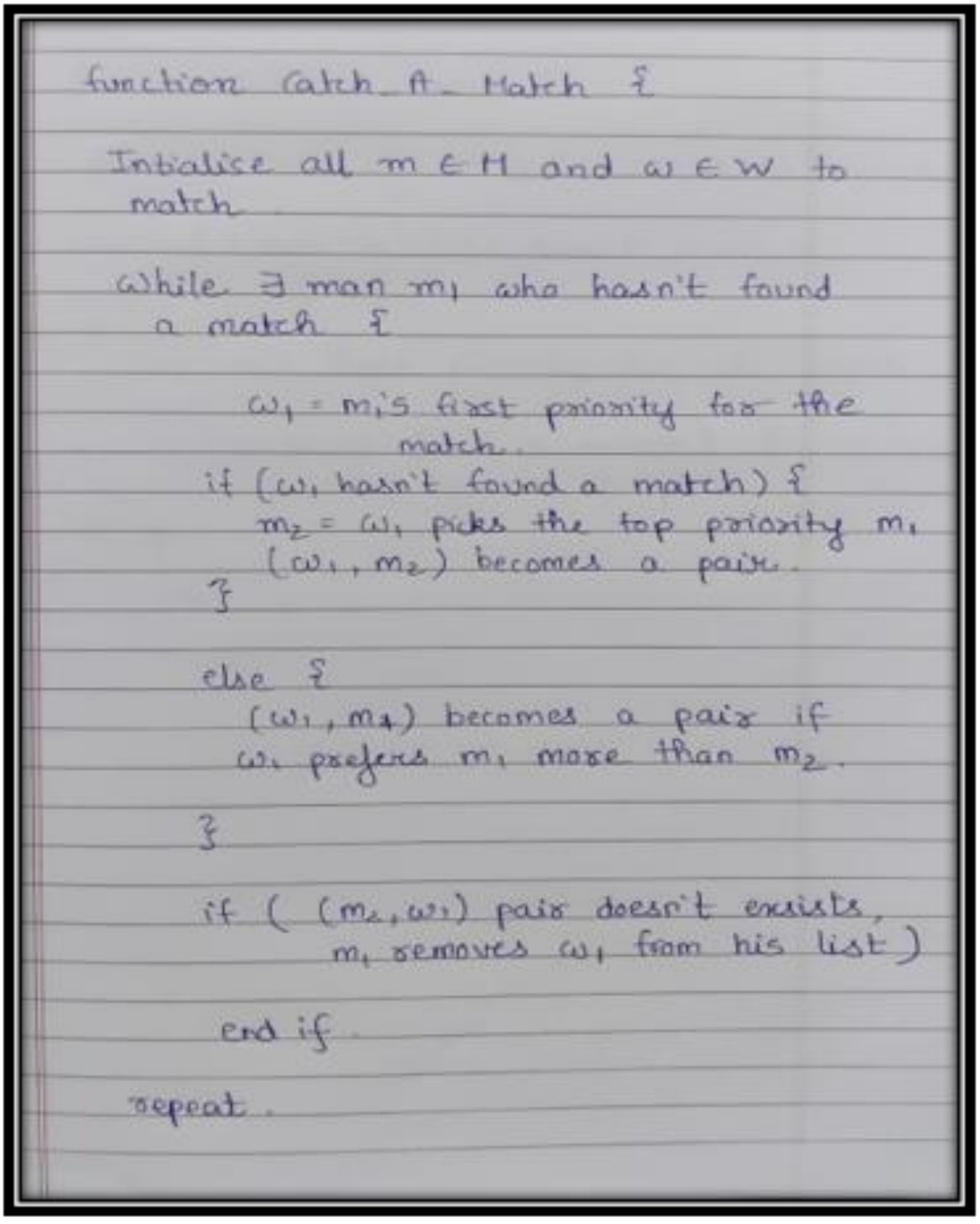
Terms to Know

- A matching is perfect if it has a size of |V| / 2, where V denotes the number of vertices.
- The weight of matching M is the sum of the weights on the edges in M.
- A minimum weight matching for Graph M is a perfect matching for M with minimum weight.
- Given a matching M, x and y (x – Boy; y – Girl) form a rogue couple, if they prefer each other over their mates in M.
- A matching is stable if there are no Rogue Couples.

Note: V and M are the notations used in the Adjacency Matrix of Graph input.



ALGORITHM



function Catch_A_Match {

Initialise all $m \in M$ and $w \in W$ to match.

While $\exists$ man m_1 who hasn't found a match {

w_1 = m_1's first priority for the match.

if (w_1 hasn't found a match) {

m_2 = w_1 picks the top priority m_1

(w_1, m_2) becomes a pair.

}

else {

(w_1, m_1) becomes a pair if w_1 prefers m_1 more than m_2.

}

if ((m_2, w_1) pair doesn't exists, m_1 removes w_1 from his list)

end if.

repeat.

Forecasting our Idea into other fields

The Match-making concept not only being limited to matching couples, but it can also be used in other fields like:

- Assigning new doctors to the Hospitals

The NRMP – where the hospitals offered positions to students – was also criticized for systematically favouring hospitals over students. Indeed, as Gale and Shapley had shown theoretically, the proposing side of the market (in this case, the hospitals) is systematically favoured. In 1995, Roth was asked to help design an improved algorithm that would eliminate these problems.

- Matching students and high- schools

The Gale-Shapley algorithm proved to be useful in other applications, such as high-school choice. Up until 2003, applicants to New York City public high schools were asked to rank their five most preferred choices, after which these preference lists were sent to the schools. The schools then decided which students to admit, reject, or place on waiting lists. The process was repeated in two more rounds, and students who had not been assigned to any school after the third round were allocated through an administrative process.

- Matching kidneys and patients

This problem was studied by Shapley and his colleagues, again in the abstract and based on the notion of stability. The proposed algorithm – the so-called top trading cycle – is in fact very simple. It is based on an initial allocation of objects and subsequent swapping. A challenge in the case of human organs is that some kidney-patient pairs may not be compatible and that complex multilateral swaps may be quite time consuming.

Again, a combination of theory and experimental work has been used to compare different versions of top trading. As a result, increasingly complex chains of kidney donations are now adopted in several U.S. states.

- Matching players in the Tournaments of games

Players in a match can also be matched by Gale Shapely Algorithm. Based upon the positions a player has gained he/ she can be matched with the next player in the order.

Working Principle

- We have n boys and n girls, each boy and girl have their own ranked preference list.
- The boy would choose the first prioritised girl from the list who are not paired. (Initially all the girls are not paired), so the boy selects the first girl from the priority list.
- The girls having atleast one boy picks a favourite (higher priority) and rejects the lower priority boy.
- The boy who got rejected by a particular girl, removes the girl's name from the priority list.
- This process is continued until all the boys and girls are paired.

CODE

```
%pylab inline import pandas as
pd import numpy as np from
collections import Counter from
copy import copy
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
C:\Users\DEV KAPADIA\anaconda3\lib\site-packages\IPython\core\magics\
pylab.py:159: UserWarning: pylab import has clobbered these variables:
['copy']
`%matplotlib` prevents importing * from pylab and numpy
  warn("pylab import has clobbered these variables: %s"  % clobbered +
```

```
man_list = ['Bingley', 'Darcy', 'Collins', 'Wickham']
women_list = ['Elizabeth', 'Jane', 'Lydia', 'Charlotte']
```

```
women_df = pd.DataFrame({'Elizabeth': [4,2,1,3], 'Jane': [1,4,2,3],
'Lydia':[1,4,2,3], 'Charlotte':[1,2,3,4]})
women_df.index = man_list women_df
```

	Elizabeth	Jane	Lydia	Charlotte
Bingley	4	1	1	1
Darcy	2	4	4	2
Collins	1	2	2	3
Wickham	3	3	3	4

```
man_df = pd.DataFrame({'Elizabeth': [2,1,3,4], 'Jane': [1,2,4,3],
'Lydia':[2,1,3,4], 'Charlotte':[3,2,1,4]})
man_df.index = man_list man_df
```

	Elizabeth	Jane	Lydia	Charlotte
Bingley	2	1	2	3
Darcy	1	2	1	2
Collins	3	4	3	1
Wickham	4	3	4	4

```
# dict to control which women each man can make proposals
women_available = {man:women_list for man in man_list}
# waiting list of men that were able to create pair on each iteration
waiting_list = []
# dict to store created pairs
proposals = {}
# variable to count number of iterations
count = 0
```

```
# while not all men have pairs while
len(waiting_list)<len(man_list):
    # man makes proposals
    for man in man_list:         if
man not in waiting_list:
```

```
    for man in man_list:
        if man not in waiting_list:
            # each man make proposal to the top women from it's list
            women = women_available[man]
            best_choice = man_df.loc[man]
[man_df.loc[man].index.isin(women)].idxmin()
            proposals[(man, best_choice)]=(man_df.loc[man]
[best_choice],
                                                   women_df.loc[man]
[best_choice])
    # if women have more than one proposals
    # she will choose the best option
    overlays = Counter([key[1] for key in proposals.keys()])
    # cycle to choose the best options
    for women in overlays.keys():
        if overlays[women]>1:
            # pairs to drop from proposals
            pairs_to_drop = sorted({pair: proposals[pair] for pair in

                    if women in pair}.items(),
                   key=lambda x: x[1][1]
                  )[1:]
             # if man was rejected by woman
             # there is no pint for him to make proposal
             # second time to the same woman
             for p_to_drop in pairs_to_drop:
                 del proposals[p_to_drop[0]]
                 _women = copy(women_available[p_to_drop[0][0]])
                 _women.remove(p_to_drop[0][1])
                 women_available[p_to_drop[0][0]] = _women
    # man who successfully created pairs must be added to the waiting
 list
    waiting_list = [man[0] for man in proposals.keys()]
    # update counter
    count+=1

 proposals
```


```
    # man who successfully created pairs must be added to the waiting
 list
    waiting_list = [man[0] for man in proposals.keys()]
    # update counter
    count+=1

 proposals
```


```
{('Bingley', 'Jane'): (1, 1),
 ('Darcy', 'Elizabeth'): (1, 2),
 ('Collins', 'Charlotte'): (1, 3),
 ('Wickham', 'Lydia'): (4, 3)}
```

REFERENCES:

1. https://www.techiedelight.com/bipartite-graph/
2. https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15251-f10/Site/Materials/Lectures/Lecture21/lecture21.pdf
3. https://medium.com/@UofCalifornia/how-a-matchmaking-algorithm-saved-lives-2a65ac448698
4. https://youtu.be/Qcv1IqHWAzg
5. https://youtu.be/0m_YW1zVs-Q
6. https://en.wikipedia.org/wiki/Stable_marriage_problem#:~:text=In%20other%20words%2C%20a%20matching,has%20been%20stated%20as%20follows%3A&text=When%20there%20are%20no%20such,of%20marriages%20is%20deemed%20stable.
7. https://www.geeksforgeeks.org/stable-marriage-problem/
8. https://www.cs.princeton.edu/~wayne/kleinberg-tardos/pdf/01StableMatching.pdf