Background: it started as a favor for a friend that wanted to create an app for matching IDC students for dating based on their character as part of his masters in psychology, but he didn’t know how to code or what algorithm to use. Based on questionnaires he gave to students, he had on each student a list of integers representing qualities, and weights (percentages) representing the importance they give to each quality **in other people**. From there I had to continue on my own. What I did was to define a score of a couple, called it “power”, kind of chances for success. I calculated the power by multiplying each grade one person gave the other person based on the scores and weights. Now, when each couple (did it for all possible couples) have a score (power), I had to take the k best couples out of all the possible couples. I wanted to take the k couples with the largest overall power (like you did with the people that needed a ride), optimizing for the whole group. I did it with a recursive method: for each couple, I enter the function one time with this couple and one without and always choose the option that gave me the greater score, eventually returning the list of couples that gave me the largest overall score (k possible couples that brought the highest sum of scores) .

*Pseudocode only for this part:*

function (list\_of\_couples\_waiting, result):

# Base case:

if size of list\_of\_couples == 0:

Return result

else:

current\_couple = list\_of\_couples\_waiting.get(0)

list\_of\_couples\_waiting.remove(current\_couple)

result\_with\_couple = function(list\_of\_couples\_waiting, result\_with\_couple)

result\_without\_couple = function(list\_of\_couples\_waiting, result\_without\_couple)

if result\_with\_couple.score > result\_without\_couple.score:

return result\_with\_couple

else:

return result\_without\_couple

As in dynamic programming, eventually if we do it for every couple in the list we’ve started with, we get the optimal list.

Of course in real life even if you have high chances it doesn’t always work so I build a function that, based on the power of a couple, chooses if this couple is happy or not and that way I simulated real life- if a couple wasn’t happy after entering this function I deleted this couple and continue to iterate until all students are either in happy relationships or run out of people to match with.

function2 (waiting\_couples, happy\_couples):

If waiting\_couples.size == 0:

Return happy\_couples

Else:

For couple in waiting\_couples:

waiting.couples.remove(couple)

If couple.isHappy():

happy.couples.add(couple)

#Match again the remaining people and re-enter function2

Return function2(waiting\_couples, happy\_couples)

# The method that checks if a couple is happy based on their score (power)

**boolean** isHappy(){

**double** random = Math.*random*();

**if** (random <= **power**){

**happy** = **true**;

}

**else**{

**happy** = **false**;

}

**return happy**;

}

I wrote it in Java for some good reason I can’t remember but I mainly code in Python. Try to run it for yourself :)