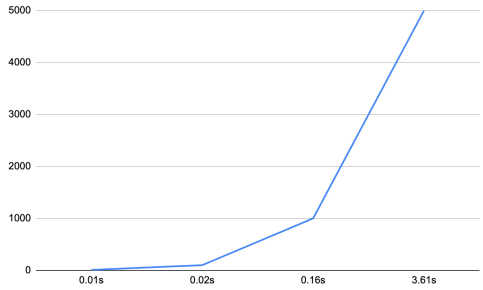
CSC 440 Assignment 1
Marriage.py's benchmark and Pandemic

Benchmarks:

0.01s	10
0.02s	100
0.16s	1000
3.61s	5000



The graph obtained using the inputs size and my running time of my Gale Shapley algorithm python implementation meet my expectations as it has a curve approximately between the curve of n and n^2

Gale Shapley algorithm attempts combinations of a knight and a lady based on each knight and lady ranking list of each opposite gender among all knights and ladies. That number of attempted combinations depends on the number of rejected knights after every knight's first proposal

The number of combinations attempted determines the running time of the Gale Shapley algorithm. The more attempts, the longer it takes for the algorithm to terminate.
For n knights and n ladies:

- In the best case with the quickest running time, each knight first proposal to a lady leads to an engagement without any lady rejecting, thus for n knights there is 1 proposal, thus for n knight there is n * 1 = n attempts made. (our graph curve should be over n curve)
- In the worst case with the longest running time, every knight only their last proposal to a lady leads to an engagement without any rejection from the n ladies, n knights get rejected n - 1 times, thus n knights make n proposals in total. Thus for n knight there is n * n = n^2 attempts made

In the best case, my function doing Gale Shapley algorithm (makeCouples) is called twice but only iterates over n knights only during the first call, on that second call (first recursive call), it reaches the base case, terminates with 'returns'. With a total of n iterations.

In the worst case, that function is called n + 1 times but it only iterates over n knights during the first n calls, on the last call it reaches the base case, terminates with 'return'. With a total of n * n = n^2 iterations (our graph curve should be under n^2 curve)

The running time for reading the file and making sure the input file meets with the expectations influences our results for Gale Shapley as I use the time command. In my program I check for the file's validity as the algorithm function runs.
Then after, I check the validity of the names on the files that weren't iterated through.
Leading to a n * n = n^2 iterations for each each execution of my program which is closer to a n^2 graph curve
My running time also depends on my type of processor and caching strategy. We abstract all this to $O(n^2)$ which is about the running time, keeping just the higher degree and omitting constants

By Yann Youbi
Professor: Dr. Daniels

Pandemic: At least 2 infected students are required to infect at least one healthy student as long as they are both adjacent to the healthy one

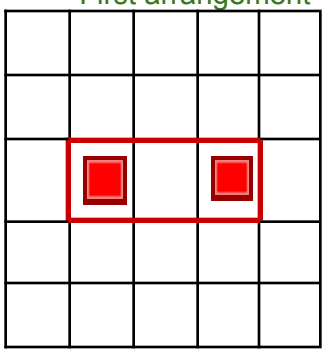
We need the infected students to be arranged in such a way they can infect as many healthy students as possible.

There are multiple ways to arrange 2 infected students in order to infect a healthy one

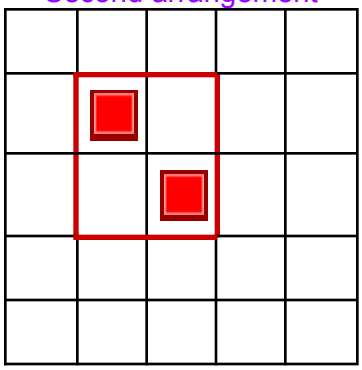
On this first arrangement only 1 healthy student is adjacent to the 2 infected ones

On the second arrangement 2 healthy students are adjacents to the 2 infected ones
The infected students at the extremity define the corners of the infection zone
So the second arrangement infects more healthy students than the first one.
With that second arrangement we have
- $2^2 = 4$ infected in total from 2 infected, after 1 day
- $3^2 = 9$ infected from 3 infected after 2 days,
- $4^2 = 16$ infected from 4 infected after 3 days,

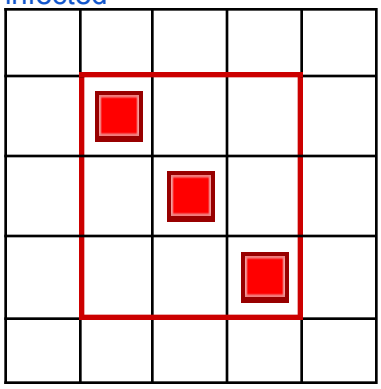
First arrangement



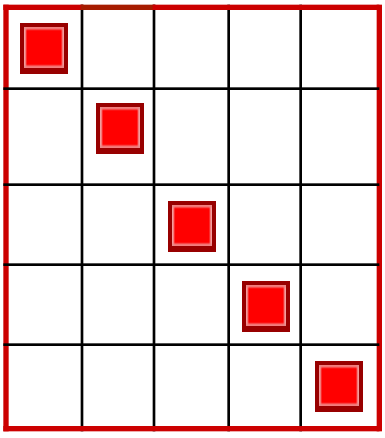
Second arrangement



Second arrangement with 3 infected



- $5^2 = 25$ infected from 5 infected after 4 days



By Yann Youbi
Professor: Dr. Daniels

1. What is the maximum number of initially infected students such that, regardless of how they are placed in the classroom, at least one initially healthy student always remains healthy?

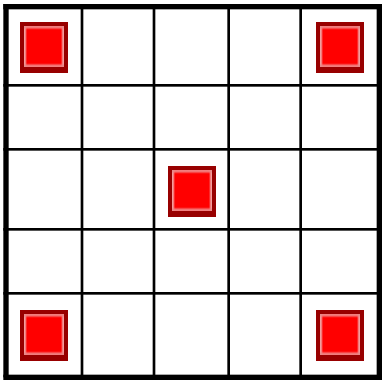
4 infected students

2. What is the minimum number of initially infected students such that there is some arrangement of many initially infected students that will result in every student eventually becoming infected?

5 infected students.

3. Can you arrange this minimum number of infected students in such a way that the infection never spreads to any healthy student?

Yes by having the n infected students separated by 2 or more healthy students, or by aligning the n infected students adjacent to each other in a same column or row. That way 2 infected ones can't be adjacent to a healthy student as they are all on the same side of the healthy



4. How would your answers change if there were n2 students in an n x n grid?

- 1. n - 1
- 2. n
- 3. Yes by having the n infected students separated by 2 or more healthy students

5. Does it matter if n is even or odd?

No, as we've seen for n = 2 and n = 3. Whether n is even or odd, the perimeter of the grid remains even (see answer to question 6)

6. What *geometric* property about the set of infected students never changes as the days pass and the infection spreads?

As the days pass and the infection spreads, the total perimeter of all infection zones remains the same; the infection zones always stop spreading when forming a rectangle or square of infected student(s). Let's say the perimeter of an infected zone composed of 1 infected student is 4. From our first and second arrangements with 2 infected students on day 1 the total perimeter of the infected zones is 4 + 4 = 8. On day 2, for the first there are 3 infected students forming one infection zone, a rectangle of perimeter 8, and for the second there are 4 infected students forming one infection zone, a square of perimeter 8.

By Yann Youbi
Professor: Dr. Daniels