Function Name: `socialDistancing`

Inputs:

1. (*char*) A NxM char array representing a crowd

Outputs:

1. (*char*) A string you telling how many people in that crowd are violating social distancing

Background:

You're heading to a MATLAB-themed rave but you need to find out whether or not your fellow partygoers are following social distancing rules! You will use MATLAB to figure out whether or not a group is violating social distancing, and if so, how many people in that group are in violation.

Function Description:

Calculate the distances between persons (marked by 'x') in a crowd (empty spaces marked by 'o') and determine if that person is at least 3 feet from everyone else. Your output string will be in the following format if:

- Everyone in the crowd is following social distancing:
  - `'This crowd is following social distancing guidelines'`
- Some (or all) people in the crowd are not following social distancing:
  - `'<number of violators> person(s) are violating social distancing rules here'`

Example:

```
>> concert = ['ooooooooooxoooooooo';
              'oooxooooooooxoooooo';
              'ooooooooxoooooooooo';
              'oooooooooooooooooxo';
              'oooooooxoooxooooooo'];
>> statement = socialDistancing(concert);
     statement → '4 person(s) are violating social distancing rules here'
```

Notes:

- Inputs can be of any size, but they will always have at least 1 person.
- One element in your array corresponds to 1x1 feet
- If a person is right next to someone else, they are 1 foot apart

Hints:

- You can use the distance formula (pythagorean theorem) on pairs of indices.

Function Name: `luigisMansion`

Inputs:

1. (*structure*) An NxM structure array representing a maze

Outputs:

1. (*char*) A string that gives the directions to solve the maze

Topics: (*iteration*), (*structures*)

Background:

You and your brother Mario have received an invitation to a luxurious high-rise hotel and decide to visit for vacation. However, after your first night there, your brother Mario disappears and you've been trapped inside a maze. You've managed to find the blueprint of the maze. Use your coding skills to find a way out!

Function Description:

The structure you are given contains 4 fields: up, right, down, and left. Each of these fields could take one of three values: `'wall'`, `'door'`, or `'exit'`. Door can be passed, while walls cannot. Starting at the top left corner, solve the maze by finding the route to the exit. Each direction you travel should be recorded in the output string separated by spaces.

Example:

`maze =`

up: down: left: right:	'wall' 'door' 'wall' 'wall'	'wall' 'door' 'wall' 'door'	'wall' 'wall' 'door' 'wall'
up: down: left: right:	'door' 'door' 'wall' 'wall'	'door' 'door' 'wall' 'door'	'wall' 'wall' 'door' 'exit'
up: down: left: right:	'door' 'wall' 'wall' 'door'	'door' 'wall' 'door' 'wall'	'wall' 'wall' 'wall' 'wall'

```
route = luigisMansion(maze)
route → 'down down right up right'
```

Notes:

- There will always be exactly one solution to exit the maze

Function Name: `zoomVsBluejeans`

Inputs:

1. (*char)* The name of the first text file
2. *(char)* The name of the second text file

File Outputs:

1. A text file comparing the two input text files

Topics: (*Low Level I/O*)

Background:

It happened. Georgia Tech classes have gone online, and a heated debate has arisen over which online virtual classroom environment is truly the best: Zoom or Bluejeans. Being the refined and intellectual person that you are, you decide to put aside your politics and rise above your emotions to find out the true answer. To do this, you turn to your most trusted and infallible source of knowledge to settle this once and for all: MATLAB.

Function Description:

Given two transcripts, one from a Zoom meeting, and another from a Bluejeans meeting, read through them and create a line by line comparison, which will be written to an output file named `'<filename1>_<filename2>_comparison.txt'`. In order to create this comparison file, you will need to read through both files simultaneously, comparing each line as you go. There will be 3 scenarios you will need to account for as you compare each line:

1. The lines are the same: In this case, you will write to the output file the line number from each file that matched, in the following format: `'File 1 line <lineNumber1> matches File 2 line <lineNumber2>.'`
2. File 1 contains lines that File 2 does not: In this case, you will need to count how many lines have been inserted into that place in File 1, and write that to the output file in the following format: `'File 1 has <num> additional lines.'` Make sure it says `'line'` if there is only 1 additional line!
3. File 2 contains lines that File 1 does not: This is the same as number 2, but vice versa. In this case, write the number of additional lines in File 2 to the output file in the following format: `'File 2 has <num> additional lines.'`

Example:

zoom.txt →

```
1| Hello
2| CS 1371
3| students.
4| How are you doing today?
5| Today we will be learning
6| MATLAB.
7| The test is tomorrow.
8| Let's begin.
```

bluejeans.txt →

```
1| Hello
2| How are you doing today?
3| Today we will be learning
4| MATLAB.
5| Let's begin.
6| We will start with
7| function headers.
```

```
zoomVsBluejeans('zoom.txt', 'bluejeans.txt');
```

zoom_bluejeans_comparison.txt →

```
1| File 1 line 1 matches File 2 line 1.
2| File 1 has 2 additional lines.
3| File 1 line 4 matches File 2 line 2.
4| File 1 line 5 matches File 2 line 3.
5| File 1 line 6 matches File 2 line 4.
6| File 1 has 1 additional line.
7| File 1 line 8 matches File 2 line 5.
8| File 2 has 2 additional lines.
```

Notes:

- There may be extra lines added at the end of one of the files. Make sure you account for this.
- You should NOT have an extra newline at the end of your output file.

Hints:

- You can get a second pointer to a file by simply calling `fopen()` again. Think about how you could use this to search ahead in the files. Just don't forget to `fclose()` both of them!

Function Name: `marchSadness`

Inputs:

1. (*char*) The name of an excel file containing team stats
2. (*char*) The name of an excel file representing a March Madness bracket

File Outputs:

1. The filled out March Madness bracket

Topics: (*cell arrays*), (*high level*)

Background:

The NCAA's basketball national championship tournament, known as March Madness, has been cancelled. You and your friends are missing out on all the fun of making bracket predictions and watching games. Have no fear; MATLAB is here! You and your friends all fill out your brackets and then use MATLAB to simulate your own tournament, called March Sadness.

Function Description:

You are given an Excel file containing the statistics for all the teams competing in your bracket tournament and another Excel file representing the bracket. The statistics file will have a list of school names in the leftmost column and is guaranteed to have the following additional columns **in no particular order**: games played (G), games won (W), points (P), opponent points (OP), free throws (FT), free throw attempts (FTA), and total rebounds (TRB). The format of the bracket will be a single column of team names such that the team in the first row plays the team in the second row, the team in the third row plays the team in the fourth row, and so on. You will use the following formula to determine which team won the game:

$$\frac{1}{8}\left(\frac{P1}{G1}-\frac{P2}{G2}\right)-\frac{1}{10}\left(\frac{OP1}{G1}-\frac{OP2}{G2}\right)+10\left(\frac{W1}{G1}-\frac{W2}{G2}\right)+8\left(\frac{FT1}{FTA1}-\frac{FT2}{FTA2}\right)+\frac{1}{12}\left(\frac{TRB1}{G1}-\frac{TRB2}{G2}\right)$$

A positive result indicates that team one was the winner while a negative result indicates that team two is the winner. There will never be a result of 0.

Play out the entire bracket and write the results to '`<winner_name>.xslx`'.

Notes:

- The number of teams in the bracket will always be a power of 2.
- When creating a new column of the output, always place the winning team at the row index halfway in between itself and its opponent, rounding up when necessary. For example, in round 2 when Princeton (row 2) plays Michigan State (row 4), the winner is placed in row 3.

Example:

Stats.xlsx →

School	G	W	TRB	P	OP	FT	FTA
Georgia Tech	31	17	1121	2126	2059	403	595
Clemson	31	16	1045	2062	1988	341	497
Michigan State	31	22	1258	2353	2007	424	565
Princeton	27	14	869	1931	1866	306	417

Bracket.xlsx →

Princeton
Georgia Tech
Michigan State
Clemson

```
marchSadness('Stats.xlsx', 'Bracket.xlsx')
```

Michigan State.xlsx →

Princeton		
Georgia Tech	Princeton	
Michigan State		Michigan State
Clemson	Michigan State	

Function Name: `puzzled`

Inputs:

1. (*char*) The filename of an image
2. (*double)* An 1xN vector of the correct positions for the pieces of the image

File Outputs:

1. A file containing the correctly reassembled image with `'_solved'` appended after the original file name but before the file extension

Topics: (*images)*

Background:

The quarantine. The boredom. Desperate times call for desperate measures. You finally break out the puzzles. Anyone can do a puzzle on their own. But, because this is the last drill problem you'll see this semester, you turn, once again, to your trusty and beloved MATLAB <3

Function Description:

Given a mixed up image and a vector of the correct positions of the pieces, break up the image and reassemble it according to the new positions. In order to do this, you will need to find the size of the position vector, and break the image into that number of pieces. All pieces should be of equal size. Once you have reassembled the image write it to a new image file with `'_solved'` appended to the original name.

Example:

```
positions = [3 2 4 1];

puzzled('numbers.png', positions)
```

`numbers.png` →

`numbers_solved.png` →

Notes:

- The width of the image will always be divisible by the length of the position vector.
- The number at each index of the position vector represents what number piece should go in that index. For example, if the first number in the position vector is a 3, that means go to the 3rd piece in the scrambled image and put it in the first position.