In this workshop, you'll write a program to analyze a file of data on the mating patterns of female horseshoe crabs. Your workshop leader will guide you through the process. Please ***do not*** *work ahead.*

1 Background

Assume you are given the file `crabs.csv` that contains data from a study of 173 nesting horseshoe crabs (J. Brockmann, Ethology, 102: 1–21, 1996). "Each female horseshoe crab in the study had a male crab attached to her in her nest. The study investigated factors that affect whether the female crab had any other males, called satellites, residing nearby her." (A. Agresti. Introduction to Categorical Data. 2nd ed. John Wiley & Sons, 2018.)

The file contains a line of data for each of the 173 horseshoe crabs in the study. Each line of data contains four pieces of data:

- *Satellites*: the number of male partners in addition to the female's primary partner (an int)
- *Width*: the width of the female in centimeters (a float)
- *Dark*: a boolean value indicating whether the female has dark coloring (True or False)
- *GoodSpine*: a boolean value indicating whether the female has good spine condition (True or False)

The values are separated by commas.

For example, the first five lines of the file contain:

```
8,28.3,False,False
0,22.5,True,False
9,26.0,False,True
0,24.8,True,False
4,26.0,True,False
```

Our goal is to compute the average number of satellites for female crabs with dark coloring and female crabs with a good spine.

2 Reading data

1. Write a function, called `get_filename` that asks the user for the name of the file containing the data and returns the filename provided by the user. The function must check if the filename entered by the user ends with `.csv` and re-prompt the user for a filename until they enter a filename ending with `.csv`.

2. Write a function called `get_lines` that takes a filename and returns a list of strings, where each string contains one line of the file. If the file does not exist, the function should return `None`.

3. Write a function called `parse_data` that takes a list of strings and returns a list of sublists, where each sublist contains each of the four values contained in a line of the `crabs.csv` file. The values in the sublists should be properly converted from strings to the appropriate data types (i.e., `int`, `float`, and `bool`). For example

   ```
   parse_data(['8,28.3,False,False\n','0,22.5,True,False\n','9,26.0,False,True\n'])
   ```

   should return
   `[[8,28.3,False,False],[0,22.5,True,False],[9,26.0,False,True]]`.

4. Write a function called `get_data` that asks the user for the name of a file and returns the data as a list of sublists, where each sublist contains each of the four values listed above. If the user enters the name of a file that does not exist, the function should print `Invalid file` and return `None`.

3 Analyzing data

5. Write a function, called `get_filtered_data` that takes an index, a value (of any data type), and a list of sublists of data values. The function should return a list containing the sublists for which the value stored at the provided index in the sublist matches the provided value. For example:

```
get_filtered_data(2, True, [
        [8,28.3,False,False],
        [0,22.5,True,False],
        [9,26.0,False,True]])
```

should return `[[0,22.5,True,False]]`. If you need to use a loop in your function, use a `while` loop.

6. Write a function called get_data_column that takes an index and a list of sublists of data values. The function should return a list containing the value stored at the provided index in each sublist. For example,

   ```
   get_data_column(0, [[8,28.3,False,False],
                        [0,22.5,True,False],
                        [9,26.0,False,True]])
   ```

   should return [8,0,9]. If you need to use a loop in your function, use a while loop.

7. Write a function called average that takes a list of integers or floating point values and computes and returns the average of the values in the list. You can use the built-in function sum in your solution.

8. Write a function called analyze_data that takes a list of sublists of values containing the data from crabs.csv and prints the average number of satellites for crabs with dark coloring and crabs with a good spine.

Lastly, here is a main function that could be used to start the data analysis:

```
def main():
    data = get_data()
    if data is not None:
        analyze_data(data)

main()
```