CISC121 Winter 2023 - Assignment 2

Due: Wednesday February 1st 11:59pm (EDT)

Notes:
- Your assignment should follow the python style guide, and any other style/commenting expectations as discussed in lecture
- By submitting the assignment, you agree that you have followed the Queen's Academic Integrity policy, and that your assignment was completed independently.

Q1: Working with Modules and Functions (10 marks)

Create a directory within PyCharm for this assignment, then create python files called **functions.py** and **a2_q1.py**

In the functions.py file, write a function called `all_odd_or_even` to meet the following specifications:
- accept any number of arguments
- return True if
  - it receives at least one argument, AND
  - all the arguments are integers, AND
  - the arguments are either all odd OR all even
- return False in all other situations (including the situation in which it is called with no arguments or invalid arguments)

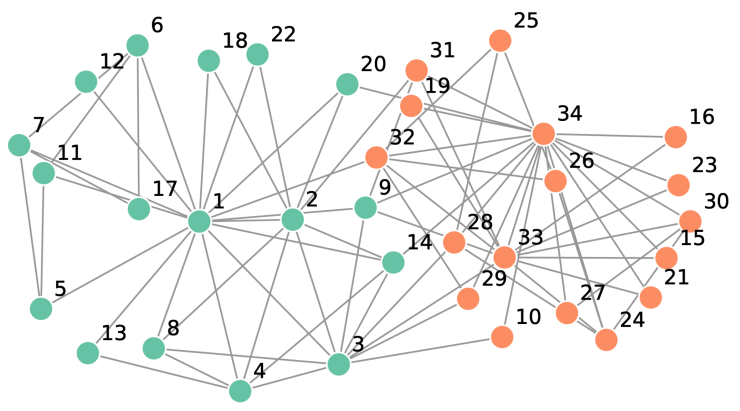
Import this function, and this function only, into your main program in **a2_q1.py**.

In this main program, you are going to ask the user to input all of the arguments into this function by doing the following:
- Begin by introducing what you are doing to the user and ask if they would like to participate in this activity. **Only proceed if the user answers yes.**
- Ask the user for an integer to add to this function.
- If an integer is provided:
  - Save the integer so you can send it to the function later
  - Ask if they would like to add another integer. **Only proceed if they say yes.**
- If an integer is not provided:
  - Quit, and give the user an error message.
- When the user is done, call the function on the integers provided by the user.

Q2: Social Networks using Dictionaries (15 marks)

The study of complex networks is an emerging field in theoretical computer science. Modeling complex networks, and specifically social networks provides a unique challenge of understanding the structure of networks, and understanding social connections.

Friendship data is the key construct in creating a social network model. In this question, we will use dictionaries to store information about a local social network. In a social network, each person is represented by a *node* and if two people are friends they have an *edge* connecting them. The number of edges that connect to a node is called the *degree* of the node. In a Social Network, the *degree* is simply the number of friends a person has. An example of a social network is in the image below:



Citation: (CC-BY 4.0) Wikipedia on Zachary Karate Club (by Paresnah)

Scenario

The input file **friendship.txt**, has the data of a group of friends. Each line shows the names of two persons who are friends with each other. Your program should work for any input files that fit the file format with a different set of data.

Example:

Tanvir	Annika
Annika	John
Nicole	Annika
John	Tanvir

Note: Each pair will only be represented once in the file. So, "John   Annika" and "Annika John" represent the same friendship and both will not appear in the file. This is called a *symmetric* relation (you will learn more about this in CISC102 and CISC203).

Your Task

1. Write a Python function `friends_to_dictionary()` that takes no input, and converts the file called **friendship.txt** into a dictionary, and returns said dictionary.
2. Write a Python function called `all_my_friends('friend')`, that takes as input a friend in the list (you may assume the friend is in the dictionary without testing in the function), and returns a *list* of all friends who are friends with the given name.
3. Write a Python function called `friendship_degree(dictionary)` that takes as input the dictionary created above, and writes the number of friends each person has (in any order) along with their names. You may use the previous function if you choose. Sample output for this function:

```
John has 2 friends:  (Robin, Anna)
Annika has 3 friends: (......)
Nicole has 1 friend: (......)
Tanvir has 2 friends: (......)
```

Now, use all of your functions in a main program called **a2_q2.py** (don't forget to import your functions file). You should have this main program initiate the dictionary you create, and then eventually output the friendship degree as described in 3.

To Submit:

Submit the following files as a **SINGLE zip file** named **a2_999999999.zip**:

1. **functions.py**: A file containing all of your functions for both questions.
2. **a2_q1.py:** the main program for the first question
3. **a2_q2.py:** the main program for the second question
4. **friendship.txt**: a copy of the friendship file we have shared with you.
5. **testing.txt:** Include all tests you run, and the output. You do not need to show the long list of call and response for the user input in Question 1. Simply state what the resulting set of input was, and then the corresponding output. Whenever your tests fail, you may include notes to the TAs. It is encouraged to discuss *why* you've chosen each test case, but not required. It is good to have 3-5 tests for each function. You should be testing your code much more often than this.