==

Web-scraping + DB + Networking

==

Instructions for Lab assignment

Strictly follow the instructions given below:

1. Solution for each problem is to be typed in a .py file already kept inside the team_name folder.
2. Do not change the structure and the names. The automatic checker will give you a **"Not Submitted"** grade otherwise, if it does not find the names.
3. Remember to follow the exact output formats specified. Extra characters, trailing spaces, etc. will lead to incorrect evaluation by auto-grader.
4. **In all questions, take input and output files as command-line inputs. (if any)**
5. Inside your code, if you create any temporary file, **make sure your code removes it.**
6. Please don't declare any global variable(s) outside the function.
7. You are allowed to create helper functions if needed but they should be invoked by one of the functions mentioned in the skeleton file.
8. Also do not add/remove any input or output arguments in the functions provided in the skeleton file.
9. **NOTE:** Make sure the solution does not have inconsistent indentation!
10. In order to auto-grade your code, we'll just import your function by the name mentioned in the problem from the corresponding .py file and run it.

Submission Guidelines for assignment **(On moodle)**

Submission Structure:

```
team_name.tar.gz
        ├── q1_server.py
        ├── q1_client.py
        ├── FINDPRIME.py
        ├── q2.py
        ├── employeeDB.db
        ├── q3.py
        ├── q4.py
        ├── requirements.txt
        └── readme.md
```

1. Students with the lowest roll number in the team should submit the assignment on moodle.
2. There should be only 1 submission from 1 team.
3. Fill in readme.md in the submission directory, which should contain your team name, team member roll numbers, their contributions, and references (cite where you get code/code snippets from).
4. Rename the directory team_name to the actual team name, E.g. Tensors.
5. **Important:** Compress the directory to <team_name>.tar.gz using :
   a. Go inside the team_name directory (where you have your q1_server.py, q1_client.py, FINDPRIME.py, q2.py, q3.py, q4.py python files and other req. dir and files are):
      cd <team_name>
   b. Create a tar file: **tar -cvzf <team_name>.tar.gz ***

General Instructions

- Make sure you know what you write, you might be asked to explain your code at a later point in time.
- The submission will be graded automatically, so stick to the naming conventions strictly

----

Problem 1

Most of you may have read about RPC (Remote Procedure Call). Now it's time to implement one. Your task is to write 3 python programs as follows:

1. **FINDPRIME.py** : It should contain only one function "find_prime(n, m)" , and should return the count of prime numbers in the given range(both inclusive). And this file should not contain anything else.

2. **q1_server.py** : It should be able to invoke the find_prime function whenever the client requests and returns the output.

3. **q1_client.py** : It should take input n, m such that n < m , and print the output it receives from the server.

For testing, you can create two terminal instances and run the server in one of them and the client in the other. Also note that you should run the server first and then the client.

Example:

```
python q1_server.py

python q1_client.py
```

```
11
20
4
```

what server prints doesn't matter. Output should be seen on the client side only. The above example input range is from 11 to 20, and output is 4, since it contains 4 prime numbers(11,13,17,19).

Note:
1. You should use the **xmlrpc** module for creating server and client scripts.
2. Server should listen on **port number 8080**.

Problem 2

Write a python program to create and manipulate a Database in Sqlite3. The program must take as input the absolute path to a file containing the data for this question. The DB you connect to must be called **employeeDB** (Note that you need to submit the employeeDB.db file which will be created)**.** The table you must create is to be named **employeeInfo** and must have the following columns:

Name(text)	**ID(INT)**	**Salary(INT)**	**City(text)**

Make the connection to the DB, setting up the cursor and creating the table in the **__init__** method of a class called **Employee**. For testing, use **`employeeInfo.csv`** provided,to populate the table in a method called **`populate_table.`**

Finally, write the following methods to handle queries specified below. The **Employee** class you write must have separate methods for each of the following operations:

- **__init__** : connect to the DB, set up the cursor and create the table.
- **populate_table**: populating that table using the given CSV file
- **print_all:** Executing and printing the result of a query that returns the entire contents of the **employeeInfo** table. *Columns are tab separated*.
- **highest_salary**: Executing and printing the result of the following query: find the name of the person or persons having the highest salary. There can be multiple people with the same salary. You must print all of them. *If there are more than one then separate them by a space.*

- **second_highest_salary**: Executing and printing the result of the following query: find the name of the person or persons having the second highest salary. There can be multiple people with the same salary. *You must print all of them. If there are more than one then separate them by a space*
- **same_city**: Executing and printing the result of the following query: find the Ids of the persons who belong to the same city. *Space separated.*

You should call each of these methods or functions from your program, to print the result to the output.

Sample Execution:-
python q2.py employee_Info.csv
Suppose employee_info.csv contains for 3 persons.
output:

```
Name   Id    Salary    City
Amit   102   200000    Mumbai
Sumit  103   200000    Mumbai
Aman   105   250000    Jaipur
Aman
Amit Sumit
102 103
```

Problem 3

You took admission in IITB, and want to know the courses offered and their respective instructors in the CSE department. You come up with an idea to have a database so that you can see the courses offered during a given semester and their respective instructors along with their Email ID at one place.

You need to write a Python program that will create the **CSE_DB** database and the following tables:

1. **CSE_Instructors** : This table contains all the details of the instructors scraped from this url *(Current Faculty part only)***.** The table should have three columns: **Instructor**, **Research_Interests**, and **Email.** All the columns are of type text.

2. **CSE_Courses** : This table contains all the courses scraped from this url. The table should have three columns **Course_Code**, **Course_Name**, and **Instructor**. Here also all the columns are of type text.

After successfully creating both the tables, you have to perform mapping on both the tables and create a **CSE_Mapped** table, keeping the Instructor as the key. But out of the mapped result, you are supposed to print a table with only three columns on the terminal which are: **Course_Code, Instructor, Email.** If the value is absent for any specific cell, then fill it up with a NA.

The table structure is as follows:

Course_Code (Text)	Instructor (Text)	Email (Text)

Note :

1. Keep this in mind while scraping from urls.
    a. You have to fetch the **Current Faculty** part only from the faculty page. **Hint**: Fetch the **#current** part first (can be stored in local html file) and then proceed with the normal web-scraping procedure.
    b. Remove "(Department Head)" suffix from the Prof. Umesh Bellur name.
    c. Remove 'Prof. ' prefix from the Instructor column because later this column will act as a key while mapping the tables.
2. Follow the instructions mentioned in the **q3.py** file.
3. The output should be printed in the terminal itself, no need to create any new file.
4. You may use BeautifulSoup to scrape the courses from the url.
5. You may use tabulate or PrettyTable, or any other of your choice to print the courses in tabular format, but the output should be in tabular format.
6. Submit only **q3.py**, please remove the **.db** file, and all other files(if any) you have created.
7. For creating **requirements.txt** file, cd to the directory where **q3** folder is present
    pipreqs q3/
    [You can install pipreqs using pip: pip install pipreqs]

Sample Output:

Course Code	Instructor	Email
CS 101	Purushottam Kulkarni	puru@ cse.iitb.ac.in
CS 207	Prof. Manoj Prabhakaran	mp@cse.iitb.ac.in
CS 213	Prof. Ajit Diwan	aad@cse.iitb.ac.in

To run your program:

```
>>> python3 q3.py
```

Problem 4

Web scraping using StackAPI.

Write a **Python program** to take multi word **input** through the command line.
Example Input : "python Numpy"

Based on the given input, we expect you to extract questions from StackOverflow using StackAPI which contains all the words in the provided input i.e. **"python**" and "**Numpy**". Questions should be fetched only if there is an **accepted** (green ticked) answer to it, and also questions should have a score(upvotes) **greater than 200**. It should be in **decreasing order** of number of upvotes.

Store the output in a CSV file, which contains the following:
question_id, tags, total_answers, link, accepted_answer.

1. question id

    e.g. 1987694

2. all the tags which are attached to the questions,

    e.g. "['python', 'arrays', 'numpy', 'options', 'output-formatting']"

3. total number of answers

    e.g. 21

4. question link

    e.g.https://stackoverflow.com/questions/1987694/how-to-print-the-full-numpy-array-without-truncation

5. corresponding accepted answer link

    e.g.https://stackoverflow.com/questions/1987694/how-to-print-the-full-numpy-array-without-truncation/#1988024

The CSV file must have the name as tag1_tag2_......_tagn.csv for n-words in your input argument. [**NOTE**: **these tags should be in lowercase**]

Example input: python Numpy

Filename: python_numpy.csv

the tag should contain all the tags in the input separated by "_".

To run your program:

python3 q4.py python Numpy

NOTE:

1. Explore the use of StackAPI, this should be very useful for you. Do check out the

StackAPI Documentation [Refer : StackAPI: A Python wrapper for the Stack Exchange API — StackAPI 0.1.12 documentation]

2. **The header of the csv file will be :**

Question_id, tags, total_answers, link, accepted_answer

3. All the rows of the csv file will be of the format specified below:

1987694, "['python', 'arrays', 'numpy', 'options', 'output- formatting']", 21, https://stackoverflow.com/questions/1987694/how-to-print-the-full-numpy-array-without-truncation,https://stackoverflow.com/questions/1987694/how-to-print-the-full-numpy-array-without-truncation/#1988024

4. Convert the user input tags to lowercase.

5. **Multi Word tags as input should be joined using '-' instead of ' '. There will be ' ' (space) only between each tag.**

6. In case, if you are getting a 'throttle_violation' exception, try this out.

[Refer : 'StackAPI' throttling exception, Throttles]

You are limited to 300 API calls a day without the key. You can provide your key to StackAPI when you set up your SITE:

SITE = StackAPI('stackoverflow', key=APP_KEY, access_token=ACCESS_TOKEN)

This will increase your quota to 10,000 API calls.

6. **For some tags, the number of questions can be very large, limit your results to top 50 according to votes.**

7. Submit only the q4.py file, please remove the csv file from the submission directory.