IF100 – Spring 2020-2021
Take-Home Exam 4
Due May 31th, 2021, Monday, 23:59 (Sharp Deadline)

Introduction

The aim of this take-home exam is to practice on loops (for/while statements), dictionaries, functions and file operations. The use of loops and file operations are necessary due to the nature of the problem; that is, you cannot finish this take-home exam without using them. You may implement the solution without using a dictionary, but that would be less efficient and much harder for you to code. Besides, beware that if we use a larger file that contains millions of lines in auto grading, your program may fail due to inefficiency if you use lists instead of a dictionary. Maybe, your code may not work on GradeChecker if your code is not efficient as supposed to be.

Description

Netflix thinks that users are not happy with the Netflix recommendations. In order to overcome this problem and increase customer satisfaction, Netflix decided to hire you as a Data Scientist to implement a Python program that will be used as a movie recommendation engine. Together with this description document, you will also be given a ***sample*** input file that includes the movie names, movie genres and user ratings. Users will want to watch a movie of a specific genre, and your program will keep suggesting the best movies with the highest average rating for that genre.

You can find the details about the Input File, User Inputs, Process and Output in the following sections.

Input File

You will be given only one input file sample. The name of the input file is `all_ratings.txt`.

Each line of the input file will contain movie information as in the following format;

userId rating title genres

Ex: 580 4.0 Jurassic Park (1993) Action|Adventure|Sci-Fi|Thriller

- There are 4 pieces of information on each line for a specific movie (userId, rating, title and genres for the movie). **Each piece of information is separated with a tab ("\t") character**, and there will be a pipe ("|") character between the movie genres.
- The example given above represents that the user with ID 580 has rated "Jurassic Park (1993)" as 4.0, and the movie has Action, Adventure, Sci-Fi, and Thriller as genres.
- The rating is a float value between 0.0 and 5.0.
- You can assume the file information will be given in the correct format. So, you don't have to perform any format check.
- You cannot make assumptions about the length of the movie titles and number of movie genres for a movie or the length of the movie_id.
- Movie genre is case insensitive, which means "Adventure" and "adVentuRe" are considered the same genres.
- You can assume that there will be no empty lines in the file. However, you cannot make any assumptions on the number of lines of this file. Keep this in mind while you're preprocessing the content of the file.

For a sample input file, you may check out the `all_ratings.txt` provided with this assignment.

Please note that *we may use some other file (in the same structure and rules surely) while grading* your take-home exams. The content of the file will most probably change for grading purposes, but the name of the file will remain the same.

User Input, Process and Output

The inputs of the program are explained below. It is extremely important to follow this order with the same format since we automatically process your programs. Also, prompts of the input statements to be used have to be exactly the same as the prompts of the "Sample Runs". ***Thus, your work will be graded as 0 unless the order is entirely correct***.

Your program will take *genre* information from the user as the first input.

- *genre* is case insensitive, which means "AcTion" and "actiON" are considered the same.
- You may assume that the entered genre will exist in the file. In other words, the user will enter this input in the correct way.

After the user enters the movie genre, your program will suggest a movie in this genre. However, you will suggest the best movie in this genre based on average movie ratings. You should also print the average rating value (with four decimal places) of the suggested movie as well. You may assume that each movie has a different average rating.

Afterwards, the user will respond whether he/she prefers to watch the movie by entering either "yes" or "no" (this is case insensitive as well, so "YeS", "No" will also be acceptable). If the user enters "yes", then you are done. Just print "`Enjoy the movie!`", and terminate the program. If the user enters "no", then you should suggest the next best movie in this genre until finding a movie that user may enter "yes". During this process, if the user enters anything rather than "yes", "yEs", "No", "no" etc, then you should warn the user and force him/her to enter a valid response.

If the user keeps saying "no" as input and there isn't any movie left in this genre to suggest, then you should print "`No movies left in this genre!`" and terminate the program.

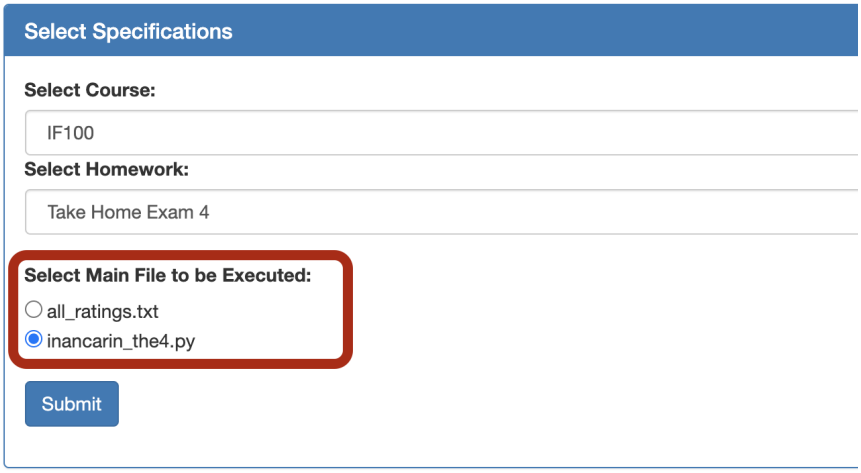
Please see the "Sample Runs" section for some examples.

Important Remarks on Working with Files

Please do NOT use the following two lines to upload the txt file to Colab. Instead use Colab's interface to upload your files as we showed in the lectures (if you use the following two lines, then GradeChecker will not work with your solution):

```
from google.colab import files
uploaded = files.upload()
```

Another note on GradeChecker is that GradeChecker does not have the txt file related to the take-home exams already. So, while uploading your code to GradeChecker, you should upload the required text file together with your .py file (2 files in total for this take-home exam: your python (.py) file and `all_ratings.txt`) and then select your .py file as the main file to be executed before pressing Submit button (see the image given below). You may hold *ctrl* button to select multiple files in the file browser (*command* button in macs)



Sample Runs

Below, we provide some sample runs of the program that you will develop. The *italic* and **bold** phrases are inputs taken from the user. You have to display the required information in the same order and with the same words and characters as below.

Sample Run 1:

```
Enter a genre: acTioN

Would you like to watch Blade Runner (1982) with rating 4.3472: yes

Enjoy the movie!
```

Sample Run 2:

```
Enter a genre: DraMa

Would you like to watch Godfather, The (1972) with rating 4.4298: no

Would you like to watch Shawshank Redemption, The (1994) with rating
4.3929: no

Would you like to watch American History X (1998) with rating 4.3919:
no

Would you like to watch Godfather: Part II, The (1974) with rating
4.3333: yes

Enjoy the movie!
```

Sample Run 3:

Enter a genre: ***Sci-fi***

Would you like to watch Blade Runner (1982) with rating 4.3472: ***no***

Would you like to watch Star Wars: Episode IV - A New Hope (1977) with rating 4.2632: ***noooo***

You entered an incorrect input, please enter yes or no: ***nein***

You entered an incorrect input, please enter yes or no: ***nope***

You entered an incorrect input, please enter yes or no: ***no***

Would you like to watch Star Wars: Episode V - The Empire Strikes Back (1980) with rating 4.2589: ***no***

Would you like to watch Eternal Sunshine of the Spotless Mind (2004) with rating 4.2051: ***yes***

Enjoy the movie!

Sample Run 4:

Enter a genre: ***AdVENture***

Would you like to watch Lord of the Rings: The Return of the King, The (2003) with rating 4.3091: ***yesIwant***

You entered an incorrect input, please enter yes or no: ***IreallyWant***

You entered an incorrect input, please enter yes or no: ***YES!***

You entered an incorrect input, please enter yes or no: ***YES***

Enjoy the movie!

Sample Run 5:

Enter a genre: ***musical***

Would you like to watch Willy Wonka & the Chocolate Factory (1971) with rating 4.1429: ***no***

Would you like to watch Lion King, The (1994) with rating 4.0204: ***no***

Would you like to watch Beauty and the Beast (1991) with rating 3.6531: ***no***

Would you like to watch Aladdin (1992) with rating 3.5741: ***no***

No movies left in this genre!

How to get help?

You can use GradeChecker (https://learnt.sabanciuniv.edu/GradeChecker/) to check your expected grade. Just a reminder, you will see a character ¶ which refers to a newline in your expected output.

What and where to submit?

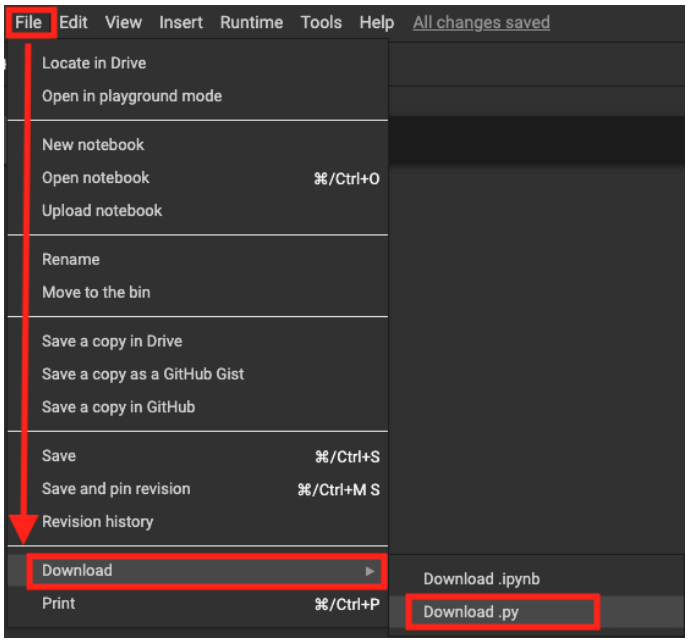
You should prepare (or at least test) your program using Python 3.x.x. We will use Python 3.x.x while testing your take-home exam.

It'd be a good idea to write your name and lastname in the program (as a comment line of course). Do not use any Turkish characters anywhere in your code (not even in comment parts). If your name and last name is "İnanç Arın", and if you want to write it as comment; then you must type it as follows:

 # Inanc Arin

Submission guidelines are below. Since the grading process will be automatic, students are expected to strictly follow these guidelines. If you do not follow these guidelines, your grade will be 0.

- Download your code as *py* file with "File" ->"*Download*" -> "*Download .py*" as below:



- Name your *py* file that contains your program as follows:

  "username_the4.py"

  For example: if your SUCourse+ username is **"duygukaltop"**, then the name of the *py* file should be: **duygukaltop_the4*.py*** (please only use lowercase letters).

- Please make sure that this file is the latest version of your take-home exam program.

- You should also upload the txt file (`all_ratings.txt`) with your python file to GradeChecker, but uploading the txt file is not necessary for SUCourse+. You may or may not include the `all_ratings.txt` file in your SUCourse+ submission.

- Submit your work **through SUCourse+ only**! You can use the GradeChecker only to see if your program can produce the correct outputs both in the correct order and in the correct format. It will not be considered as the official submission. You must submit your work to SUCourse+.

- If you would like to resubmit your work, you should first remove the existing file(s). This step is very important. If you do not delete the old file(s), we will receive both files and the old one may be graded.

General Take-Home Exam Rules

- Successful submission is one of the requirements of the take-home exam. If, for some reason, you cannot successfully submit your take-home exam and we cannot grade it, your grade will be 0.

- There is NO late submission. You need to submit your take-home exam before the deadline. Please be careful that SUCourse+ time and your computer time may have 1-2 minutes differences. You need to take this time difference into consideration.

- Do NOT submit your take-home exam via email or in hardcopy! SUCourse+ is the only way that you can submit your take-home exam.

- If your code does not work because of a syntax error, then we cannot grade it; and thus, your grade will be 0.

- Please submit your **own** work only. It is really easy to find "similar" programs!

- Plagiarism will not be tolerated. Please check our plagiarism policy given in the syllabus of the course.
- ***You <u>must</u> use Google Colab while developing your solutions. Otherwise, you might be asked to have an interview with the instructors or we might not help when you object.***

Good luck!
Ethem Tunal Hamzaoğlu
& IF100 Instructors