**READ ME**

# Designer

Letao Li  
Tianyi Hu  
Haojia Li

# Student Query Tool

This program can read student records from a file and see some detail of it. Authorized user can access the whole data frame or part of it with certain criteria by simply inputting number of the function. Besides, the whole program is able to iterate, which allows user to generate several summary reports from the same data file in one single process.

## Overview

SQT aim to help professors and teachers in reading data and producing some shot reports quickly. Besides, as a final group project of Programming with Python courses, SQT is showcase of what we have learnt in the Fall semester.

The functions of this program are:

1. Display all student records;
2. Display students whose last name begins with a certain string (case insensitive);
3. Display all records for students whose graduating year is a certain year;
4. Display a summary report of number and percent of students in each program, for students grading on a certain year;
5. Display a summary report of number and percent of students in each program, for students grading after a certain year.

## Package

Pandas is used in this program.

## Usage

This program is to let the user enter queries to see a detail or summary report.

1. Pressing '1' to see all student records. Simply through Pandas' function of reading txt, we could simply access to data in file, and show them by printing it.

if inpt == '1':   
 print(df)

1. Pressing '2' to find correct name with given start chars of last name. Certain start chars are decided and inputted by the user. A 'for' loop is established here to iterate each last name in the 'Lastname' column. After matching them with the user's target chars, the program will collect them and show them to the user. If the user input nothing, no data will be returned. And the user will be asked whether she would like to run the program again.

*# function for finding correct name with given start chars of lastname*

**def** Lastname(lastname):

df\_ln = df[df['Last'].str.lower().str.startswith(lastname)].reset\_index().drop(columns=['index'])

df\_ln.index = df\_ln.index + 1

retur**n** df\_ln

1. Pressing '3' to filter all records by given graduate year. User will be asked to input the year them want to search. Then the program will search, match and pick each record in 'GradYear' column, and put them in a data frame. It will be printed and showed to the user. If the user does not enter a number, no data will be returned. And the user will be asked whether she would like to keep using the system.

# input 3 for filter all records by given gradyear  
# function for filter all records by given gradyear  
def Gradyear(year):  
 result = df[df['GradYear'] == int(year)].reset\_index().drop(columns = 'index')  
 result.index = result.index + 1  
 return result

1. Pressing '4' to make a short summary report on a certain graduate year. Like function 3, program will ask the user for the year they want. Beside gathering data as function 3 above, the program counts the number and the percentage of students group by program as well. Show them in good format is necessary too. If the user does not enter a number, no data will be returned. And the user will be asked whether she would like to keep using the system.

# input 4 for summary report on a certain gradyear  
# function for summary report on a certain gradyear  
def Grad\_on\_year(year):  
 df\_grad\_on\_year = df\_2.copy()  
 df\_grad\_on\_year = df\_grad\_on\_year[df\_grad\_on\_year['GradYear'] == int(year)][['DegreeProgram', 'Num of student', 'Percentage']].groupby('DegreeProgram').sum()  
 # convert percentage from float to percent  
 df\_grad\_on\_year['Percentage'] = df\_grad\_on\_year['Percentage'].apply(lambda x: format(x, '.2%'))  
 # reset index  
 df\_grad\_on\_year = df\_grad\_on\_year.reset\_index()  
 # set index start from 1  
 df\_grad\_on\_year.index = df\_grad\_on\_year.index + 1  
 return df\_grad\_on\_year

1. Pressing '5' to summary report after a certain graduate year. More datas are included comparing with function 4. The filtering criteria becomes the records whose 'GradYear' larger than that the user inputs. Similar to function '4', records will be sum up by the group of programs. The format showing to the user will be desirable too. If the user does not enter a number, no data will be returned. And the user will be asked whether she would like to keep using the system.

# input 5 for summary report after a certain gradyear  
# function for summary report after a certain gradyear  
def Grad\_after\_year(year):  
 df\_grad\_after\_year = df\_2.copy()  
 df\_grad\_after\_year = df\_grad\_after\_year[df\_grad\_after\_year['GradYear'] > int(year)][['DegreeProgram', 'Num of student', 'Percentage']].groupby('DegreeProgram').sum()  
 # convert percentage from float to percent  
 df\_grad\_after\_year['Percentage'] = df\_grad\_after\_year['Percentage'].apply(lambda x: format(x, '.2%'))  
 # reset index  
 df\_grad\_after\_year = df\_grad\_after\_year.reset\_index()  
 # set index start from 1  
 df\_grad\_after\_year.index = df\_grad\_after\_year.index + 1  
 return df\_grad\_after\_year

1. Whenever each function finishes, the system will ask the user if she would like to keep using it. Inputting 'yes' or 'y' the system will run again, while inputting others the system will quit.

# ask if the user would like to keep using the system  
decision = input('Do you like to try again? Enter Y/Yes to try again, enter something else to stop: ').lower()  
if decision == 'y' or decision == 'yes':  
 continue  
else:  
 print('Thanks for using the system, bye bye.')  
t = False