Placement script for Pitzer First Year Seminar

```
This code was written by Brian Simpkins in consultation with Professor Sarah
Gilman during the Fall semester of 2022. It is intended to accept a .csv file
listing the course preferences of the incoming class, and outputs a corresponding
placement. See instructions below for using the program, as well as a description
of the codebase and the algorithms behind it.
```

Instructions for use

Initial Placement

To build an initial FYS placement, first you should verify that your input data contains the required columns. Below is a list of the columns the software will look for. If they have been renamed for you, either change the column name back, or modify the code to accept the new name. The order doesn't matter.

- CX ID -> A unique student ID
- Completed Form -> Whether the student has any preferences. Value must be either "Yes" or "No".
- Preference [1, 2, 3, 4, 5] -> The given class preference for each student
- Gender
- IPEDS Classification
- First Gen
- Major Interest by Division

Once you have verified that your input file looks good, open the "FYS_Placement" .exe file. It should upon a screen with a dialogue box for "Input form filepath". This filepath should point to the input csv file. It can either be absolute (C:\Users\John\Documents\FYS_Data.csv), or it can be relative to the location of the executable (.\data\FYS_Data.csv). Then, give a file path for the output file. Then, press the "Initial Placement" button, and wait for the "Output saved!" screen. The file which contains the assignments is called "FYS_StudentAssignments.csv", and the file containing demographic statistics about the placement is called "FYS_AssignmentStats.csv".

Updating Placement

To update a given placement, you will need to verify that your input contains everything required. It needs everything required by the "Initial Placement", plus two additional columns - "Assignment" and "Keep Placement". The former contains the class name which the student was assigned to, and the latter contains an "x" if the student should stay in their assigned class. If anything else is present, the student will be reshuffled.

Once you have verified your input, open the "FYS_Placement" .exe file. It should upon a screen with a dialogue box for "Input form filepath". This filepath should point to the input csv file. It can either be absolute (C:\Users\John\Documents\FYS_StudentAssignments.csv), or it can be relative to the location of the executable (.\data\FYS_StudentAssignments.csv). Then, give a file path for the output file. Then, press the "Update Placement" button, and wait for the "Output saved!" screen. The file which contains the assignments is called "FYS_StudentAssignments.csv", and the file containing demographic statistics about the placement is called "FYS_AssignmentStats.csv".

Description of the software

Code and Libraries

The Python code which places students into their respective First Year Seminars is organized as follows:

- FYS_Placement.py contains code which builds the Graphical User Interface (GUI) that enables the user to interact with the placement software. The GUI is based on the PyQt5 package, and is largely straightforward. When the program is executed, a page is built with dialogue boxes for a filepath and two buttons - one for updating the placement, and the other for creating a new placement. When either button is pressed, the code checks to make sure that the entered filepath exists, and then it begins the placement process. When the process has completed, a box appears and says that the output is saved. There is one unusual aspect of this code: in order to allow the program to operate without stalling the computer, the placement program exists in a separate "thread". Understanding what this means is not entirely important, but note that the "place_students" function is called in the "PlacerThread" class for this reason. When the thread finishes (meaning that the placement has completed), the output is saved to the provided filepath.

- placement.py contains the code which actually updates and creates the placements. When it is first called, it loads in the .csv file from the provided filepath and performs some initial data "tidying". When the function called "place_students" is run, 100 possible placements are made and are then evaluated for their quality. When the function "save_assignments" is called, the best placement (of the 100) is saved as a csv, and some basic demographic statistics are calculated and saved to the same filepath. The packages "numpy", "pandas", and "unicodedata" are used for data cleaning and manipulation. The "math", "random", and "collections" packages are used for the placement algorithm itself.

Backing Algorithm

The algorithm which places students into their classes is optimized for simplicity, and to minimize the "preference depth" required for the placement. In other words, if it is possible to place students such that none recieve their 4th or 5th choice, we want this algorithm to find it. The basic design of the algorithm was based on R code developed by Professor Sarah Gilman, though it has evolved since.

First, the classes are sorted by their "popularity". The class with the fewest students who named it as one of their choices is filled first. Therefore, the final class to be filled is the "most popular" one. When a class is filled, it takes students who preferred it, starting with those who selected it as their first choice, and proceeding forward as such. If there are more students in a "preference level" than there are spaces in the class (all classes should have the same number of students, plus or minus one), then the students in the given level are randomized. Of this randomization, the students who do not match the majority race of the class are chosen, and then those who do not match the majority gender. To be more specific, the order in which the students are selected goes "minority race + minority gender, minority race + majority gender, majority race + minority gender, majority race + majority gender". If any of the classes cannot be filled (because no remaining students named it as a preference), then the class is swapped up in the "popularity" ordering, and the entire process is restarted.

Executable Generation

In order to generate the executable, which allows a user to double-click a file instead of downloading python and running it from the command line, I used a package called "pyinstaller". Executables are good in that they bundle together all of the code, libraries, and interpreters (the thing that runs Python) into one small file, so that the user doesn't need to install all of those dependancies in order to run the program. However, a downside of executables is that they only work for the microarchitecture of the computer that they were generated on. So if I generate an executable on an x86 64-bit Windows computer, it won't run on an ARM-based MacOS computer. Therefore, I have included executables for some popular microarchitectures. If the executable stops working, you either might be missing **Microsoft Visual C++ Redistributable** (which you need), or you may need to generate a new executable.

Potential Future Issues

I can see two potential problems arising with the software in the future.

- The first potential issue is that the executable cannot be run, either because the host computer doesn't have **Microsoft Visual C++ Redistributable** installed, or because there is no executable for the particular computer's microarchitecture. My recommendation is that you read about "PyInstaller" online - their documentation should explain everything and help you get the program up and running.
- The second issue is that the program is completely unable to build a valid placement, because some classes can't be filled no matter what. My suggestion here is that you build a sufficient placement on your own, and modify the preference .csv to take advantage of the "Update" feature. First, put the assignment in the .csv in a column called "Assignment", and then add a "Keep Placement" column to your .csv file. Put 'x's in this column for the students who can remain in their class, and use the "Update Placement" button to reshuffle the remaining students. Hopefully, it will improve upon your placement.