Assignment 1: SimpleEnroll📚📚

Due Friday, January 26th, 2024 at 11:59 PM

Table of Contents

Overview

It's that time of the quarter again; time to use SimpleEnroll again 🤗 Wootwoot. One thing everyone realizes in their Stanford career at one point is that they have to *eventually* graduate — and so enrolling in classes becomes a strategic endeavor to maximize the XP towards graduation, while also being able to sleep more than 4 hours a night!

In this hopefully short assignment, we're going to use data from the ExploreCourses API to figure out which CS classes on ExploreCourses are offered this year, and which are not! We'll be taking advantage of streams, while also exercising initialization and references in C++. Lets jump in ʕ•́ᴥ•̀ʔっ

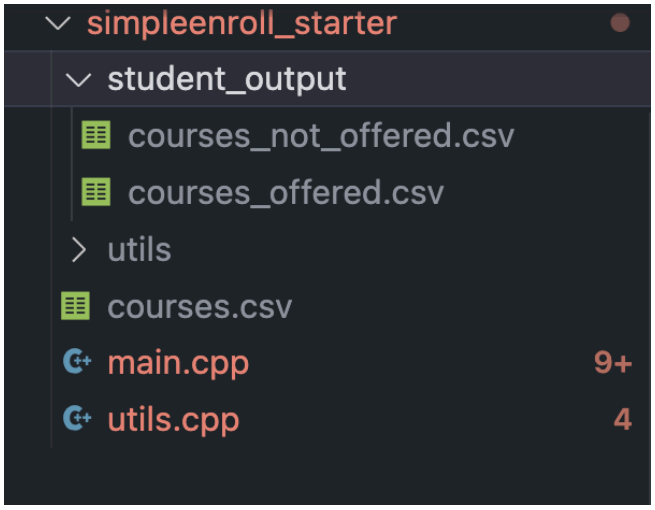
There are really only two files you should care about:

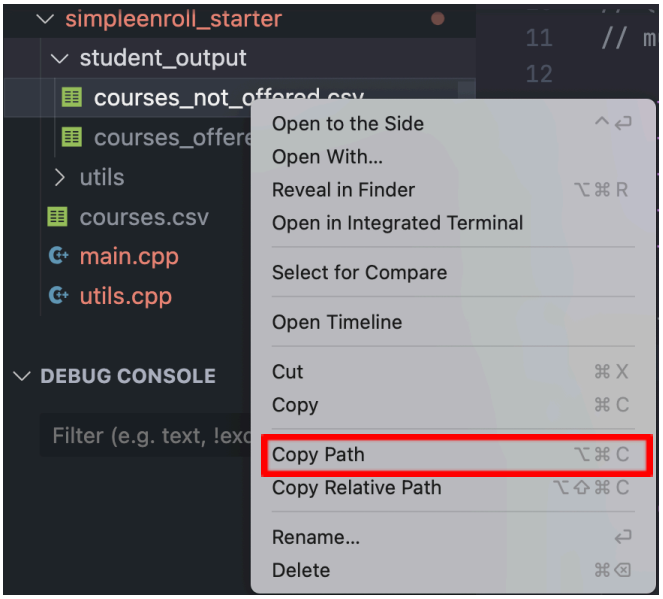
`utils.cpp`
`Main.cpp`

Please do the following to download the starter code here! Please unzip the file.

Part 0: Read the code + add paths + Course struct

1) Take a look at the `main.cpp` function, and take special notice of how `vector_of_courses` is passed into parse_csv, `write_courses_offered`, and `write_courses_not_offered`. Think about what these functions are doing, do you need to change anything in the function definition? Spoiler, you do.
2) In the `main.cpp` and `utils.cpp` you need to *manually* add paths to the code. Here's how you can do this.
    a) In VSCode:





Paste the path of

- courses_offered.csv as a string in the variable `std::string COURSES_OFFERED_CSV_PATH`
- `courses_not_offered.csv` as a string in the variable `std::string COURSES_NOT_OFFERED_CSV_PATH`

3) Write the correct types for the fields in the `Course` struct. Remember what types streams deal with?

Part 1: parse_csv

Check out `courses.csv`, it is a csv file, with three columns: `Title`, `Number of Units`, and `Quarter.` Implement `parse_csv` so that, for *each* line in the csv file, it creates a struct `Course` containing the Title, Number of Units, and Quarter for that line.

A couple of things you need to think about:

1) How are you going to read in `courses.csv`? Muahahaha, perhaps a stream 😏?
2) How will you get each line in the file?

Hints:

1) Make use of the `split` function we provide to get a vector that looks like the following: `[Title, Number of Units, Quarter]`
    a) Check out the implementation – ask us any questions about it – you should be able to reason about it since it's using a `stringstream`

Part 2: write_courses_offered

Ok. Now you have a populated `vector_of_courses` which has all of the records of the `courses.csv` file neatly stored in a `Course` struct! You find yourself interested in ***only*** the courses that are offered, right? In this function, write out to `student_output/courses_offered.csv` all the courses that don't have **"`null`"** or the `quarter` field.

IMPORTANT → When writing out to the file please follow this format:

`<Title>,<Number of Units>,<Quarter>`

There are no spaces between the commas! The autograder will not be happy if this format is not followed!

Additionally, keep track of the classes that are offered perhaps with another vector and delete them from `vector_of_courses`. **This means that after this function runs,** `vector_of_courses` **should ONLY contain classes that are not offered!** For this, be sure to take a look at the PROVIDED function: `delete_elem_from_vector`!

Don't worry, I know you're wondering what this is:

```
auto it = std::find(v.begin(), v.end(), elem);
```

We got you……in like 3 weeks.

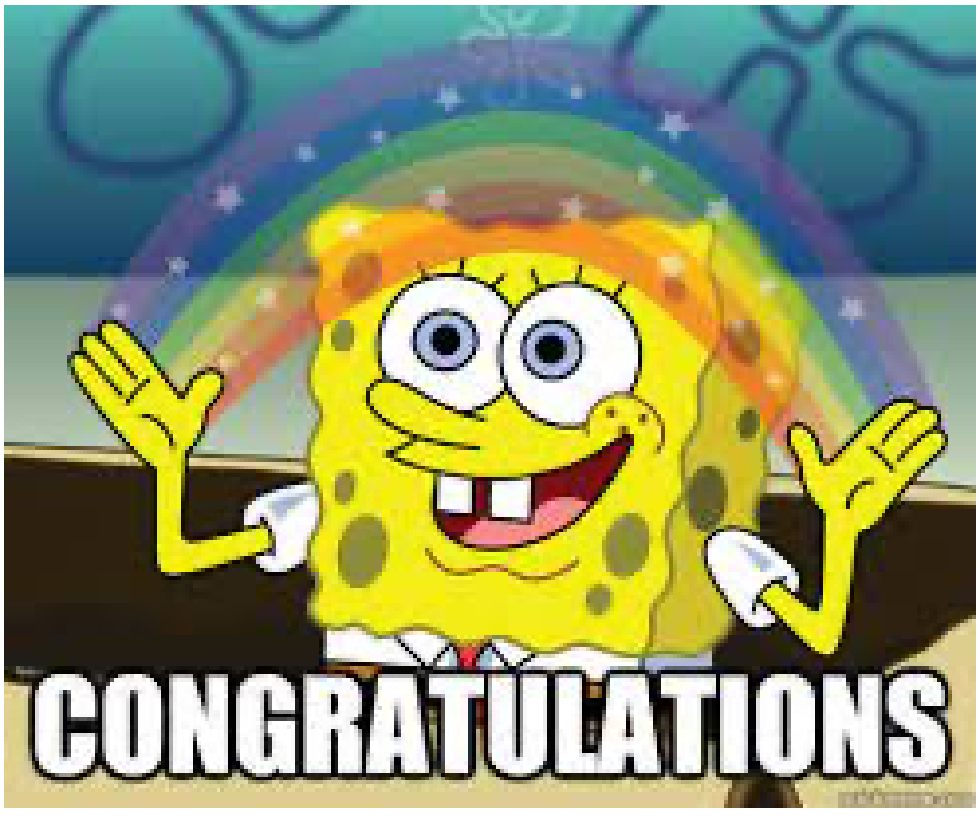
Part 3: write_courses_not_offered

So you're curious about classes that aren't offered…curiosity hurt the cat. In the `write_courses_not_offered` function, write out to `student_output/courses_offered.csv` the classes in `vector_of_courses.` Remember since you **deleted** the classes that are offered in part 2, `vector_of_courses` trivially contains ONLY classes that are not offered – lucky you. So this step should look really similar to part 2 except shorter and a *tiny* bit simpler

Congrats you're done!

The instructions to submit this assignment will be posted on Ed closer to the assignment deadline

Plz leave feedback here!



Shoutout

A special thank you to Jim Sproch for providing the ExploreCourses API. After an unsuccessful attempt to web-scrape the ExploreCourses website, I was directed to Jim Sproch who was kind enough to not only provide us the API but also answer our questions about its use!