CS 420 Advanced Programming Languages
Fall Semester, 2022
Assignment 2
San Diego State University -- This page last updated on 8/24/22

Assignment 2
Due Sept 18 at 11:59 pm

Use Rust to implement the following.

1. (20 points) Create a public struct called **StudentGrades**. The struct contains a name and a vector of grades. The **StudentGrades** struct has at least two methods (associated functions): **average** and **grade**. The **average** method returns the average score for that student. The **grade** method returns the student's letter grade based on the scale below. The StudentGrades should not lose any information when calling either **average** or **grade**.

Letter Grade	
A	90-100%
B	80-89%
C	70-79%
D	60-69%
F	0-59%

2. (20 points) Create a public struct called **CourseGrades**. The struct has a type-associated function **from_file**(file_path:String). The argument to the method is a path to a file. The method reads the file. For each line in the file, it creates a **StudentGrades** object. Each line of the file is in the format:

    name, grade1, grade2, ..., gradeN

    That is, each line starts with a name, followed by a comma, which is followed by optional spaces, which are followed by a numerical score. Each score represents a grade event, that is, an assignment or exam. Grades are separated by a comma followed by optional spaces. Here is a sample file:
    roger, 100.0, 80.0, 80
    pete, 70, 75.5,73

The struct has at least two methods: (associated functions): **average** and **student**. With the input N an integer, the average method returns the average score of all students on the N'th grade event. The first grade event is indexed by 1, not 0. **average** returns an Option, as the input N could be out of range. **student** has one parameter: a string which is a student's name. It returns an Option. If the name is found in the course, the StudentGrades is returned. Neither method should be destructive. You are not to use a CSV crate or libraries.

3 Create a public struct called **CourseSchedule**. The struct has a type-associated function **from_file**(file_path:String). The file_path is a path to a .xls file of the SDSU schedule. One is downloadable from the same web page as this document. The struct has at least one method, **courses_at(time: &str, days: &str).** The time is given in 24-hour format—for example, 1800 and 815. Days are limited to M, T, W, TH, MWF, MW, and TTH. The method returns a vector of **Course** instances for courses being taught at the day and time indicated. The Struct **Course** has at least the fields: **catalog_numbe**r, **title**, and **instructor**. All are string types. You can use a CSV library on problem 3 if you like.

4 (15 points) Unit Tests. Write a unit test for the method in each of the three problems above.

Late Policy

An assignment turned in 1-7 days late will lose 5% of the total value of the assignment per day late. The eight-day late penalty will be 40% of the assignment, the ninth-day late penalty will be 60%, and after the ninth day late, the penalty will be 90%. Once a solution to an assignment has been posted or discussed in class, the assignment will no longer be accepted. Late penalties are always rounded up to the next integer value.