Practice 1: Patata brava

Programming 2

Academic Year 2023-2024

This course's practice involves developing a faculty profile management system. The necessary concepts for this practice are covered in *Unit 1* and *Unit 2* of the theory.

Delivery Conditions

- The deadline for this assignment is **Friday, March 15th**, by **23:59**.
- You must submit a single file named `prac1.cc` containing the code for all the functions.

Code of Honor

If any form of copying (total or partial) is detected in your assignment, you will receive a **0** grade, and the Polytechnic School will be notified to take disciplinary action

It is acceptable to discuss possible solutions to assignments with your classmates
It is acceptable to enrol in an academy if it helps you study and complete the assignments

It is unacceptable to copy code from other classmates or ask ChatGPT to do the assignment for you
It is unacceptable to enrol in an academy to have the assignments completed for you

Seek help from your teacher if needed
Do not copy

General Rules

- You must submit the assignment exclusively through the practice server of the Department of Software and Computing Systems (DLSI). You can access it in two ways:
  - Main page of DLSI (`https://www.dlsi.ua.es`), option "HOMEWORK UPLOAD"
  - Directly at the address `https://pracdlsi.dlsi.ua.es`
- Things to consider when submitting:
  - The username and password for submitting assignments are the same as those used in UACloud
  - You can submit the assignment multiple times, but only the last submission will be evaluated
  - Submissions via other means, such as email or UACloud, will not be accepted
  - Late submissions will not be accepted
- Your assignment must compile without errors using the C++ compiler available in the Linux distribution of the practice labs

- If your assignment cannot be compiled, your grade will be 0
- 70% of the assignment grade will depend on automatic correction, so it is essential to strictly adhere to the specified texts and output formats in this assignment. The remaining 30% will depend on manual code review by your teacher, so you must also adhere to the course's style guide. Additionally, updating your code on *GitHub* every week will be taken into account
- At the beginning of all submitted source files, you must include a comment with your identification number (NIF or equivalent) and your name. For example:

prac1.cc

```
// ID 12345678X GARCIA GARCIA, JUAN MANUEL
...
```

- The calculation of the assignment grade and its relevance to the final grade of the course are detailed in the presentation slides of the course (*Unit 0*).

1 Practice Description

Patabrava.com was a website where class notes and epic phrases from university teachers across the country were collected. It was created over twenty years ago and had one million users before its disappearance in 2019.

In this practice, you will pay tribute to the legendary website by creating a program to manage teacher profiles. You will be able to include various details for each teacher, such as their rating, nickname, or most epic phrases.

2 Implementation Details

Several files necessary for the correct completion of the assignment will be published on the course's Moodle:

- `prac1.cc`. This file contains a program skeleton on which to base your assignment. Download it and add your code to it. This file contains the following information:
  - The necessary structures (`struct`) for the assignment
  - An enumerated type `Error` that contains all possible error classes that may occur in this assignment (e.g., `ERR_OPTION`)
  - A function `error` that displays the corresponding error message on the screen based on the parameter passed to it. For example, when the function receives the parameter `ERR_OPTION`, it will display the message `ERROR: wrong menu option`
  - A function `showMenu` that displays the program menu on the screen
  - A `main` function that implements the management of the main menu and calls the corresponding functions depending on the option chosen by the user
- `autocorrector-prac1.tgz`. It contains the files of the autocorrector to test the assignment with some input tests. The automatic correction of the assignment will be performed with a similar program, using these tests and others defined by the teachers
- `prac1`. Executable file of the assignment (compiled for 64-bit Linux machines) developed by the teachers, so you can test it with the inputs you want and see the expected correct output

- In the assignment correction, only data of the correct type will be provided, although the values may be incorrect. For example, if the rating of a teacher is requested, it will always be tested with an integer value, which could be `-1237` or `0`, but never with a value of another type (character, string, floating-point number, etc.)
- To avoid problems with character encoding, accents, diaeresis, or the letter ñ will not be used in any of the texts entered to evaluate this assignment (teacher's name, nickname, epic phrases, etc.)
- The rest of the decisions in the implementation of the assignment are up to you, but keep in mind that the source code will be reviewed by your teacher following the style guide published on the course's Moodle. Part of the assignment grade depends on this review

3 Program Workflow

The program you will develop will allow you to manage teacher profiles in your degree program. To better manage these profiles, and in anticipation that you will continue using this program throughout your academic career, you can group them by academic year.

For each teacher, you will be able to store different information, such as their name, nickname, the subject in which you had them (for simplicity, we will assume that you will not encounter the same professor in any other subject), a list of their most epic phrases (which you can rate from 1 to 10), and also a rating for the teacher, in this case from 1 to 5.

4 Components

The program to be developed must be able to manage three fundamental elements: *courses*, which represent each of the academic years in which the degree program is divided, *teachers*, who are each of the subjects about which you want to store information and are grouped into courses, and *epic phrases*, which are those brilliant speeches that your professors sometimes deliver in class and are associated with a specific teacher. The following sections describe the structure of each of these components in detail.

4.1 Epic Phrase

Structures of type `Phrase` store information related to epic phrases uttered by teachers. Each epic phrase will contain the text of the phrase (`text`), the date it was uttered (`date`), and a rating field (`rating`) represented by a number ranging from 1 to 10. This information will be stored in a structure with the following format:

```
struct Phrase {
    string text;
    Date date;
    int rating;
};
```

The `Date` data type is in turn a structure containing the `day`,`month`, and year `year` of the date the phrase was uttered. This structure is defined as follows:

```
struct Date {
    int day;
    int month;
    int year;
};
```

For example, the text of the phrase could be *Imaginemos una vaca esferica* (field `text`), uttered on the *10* (`day` field of `date`) of *2* (`month` field of `date`) *2021* (`year` field of `date`) with a rating of *8* (field `rating`).

4.2 Teacher

A teacher will have associated a set of epic phrases (`listPhrases`) with the structure mentioned earlier. Additionally, their `name`, `nickname` if any, the `subject` they have taught, and a `rating` of their performance from 1 to 5 will be stored. To store all this information, a structure of type `Teacher` will be used with the following structure:

```
struct Teacher {
    string name;
    string nickname;
    char subject[MAXSUBJECT];
    int rating;
    vector<Phrase> listPhrases;
};
```

The constant `MAXSUBJECT` will have the value `50`, so the name of the `subject` will have a maximum length of 49 characters (note that we also need to store the `'\0'`).

For example, we can have a professor named *Eduardo Smith*, nicknamed *Cocobongo*, teaching the subject *Metafisica II*, with a rating of *4*, and who has three epic phrases, each with the information described in Section **??**.

4.3 Academic Year

An academic year is stored in a structure `AcademicYear` and groups all the teachers you had in a specific year, for example, during 2023 - 2024. It will contain an identifier of the starting year (`id`) and a list of teachers (`listTeachers`) with all their information. Academic years will be stored in a structure with the following format:

```
struct AcademicYear{
    int id;
    vector<Teacher> listTeachers;
};
```

We can have a course with the identifier *2023* (`id`) with three teachers stored in the `listTeachers` vector, each with the information described in the previous section.

- For simplicity, a teacher can only be assigned to one academic year. Also, remember that they can only teach one subject.

5 Menu

Upon executing the program, the main menu of the program will be displayed on the screen, waiting for the user to choose an option:

Terminal

```
1- Add academic year
2- Delete academic year
3- Add teacher
4- Delete teacher
5- Show teacher
6- Add phrase
7- Summary
q- Quit
Option:
```

The valid options that the user can input are numbers from `1` to `7` and the letter `q`. If the chosen option is not one of these (for example, `9`, `x`, or `;`), the error `ERR_OPTION` will be emitted by calling the `error` function with that parameter. When the user chooses a correct option, the code associated with that option should be executed. Upon finishing it, the main menu will be displayed again, prompting for another option, until the user decides to exit the program using the `q` option.

6 Options

The following sections describe the functionality that each of the seven options displayed in the main menu of the program should have.

6.1 Add academic year

This option allows the user to input a new academic year, which will be stored in the `id` field of the `AcademicYear` structure, using the `addAcademicYear` function that you have to implement. This function is activated when the user chooses option `1` from the main menu. The program will prompt for the academic year with the following message:

Terminal

```
Enter academic year:
```

If an empty string is entered, the `ERR_EMPTY` error will be emitted, and it will return to the main menu. If that academic year already exists, the `ERR_DUPLICATED` error will be emitted, and the user will be prompted again to input the year with the same message. No further checks need to be performed on the input year. If everything is correct, the entered value will be stored in the course identifier, and it will return to the main menu.

6.2 Delete academic year

This option allows the user to delete a course along with all the teachers associated with it. To do this, you will need to implement the `deleteAcademicYear` function. This function is activated when the user chooses option `2` from the menu. First, the user will be prompted to input the academic year they wish to delete with the following message:

Terminal

```
Enter academic year:
```

If the user inputs an empty string, the `ERR_EMPTY` error will be emitted, and it will return to the main menu. If an existing course is entered, it will be deleted from the program along with all the teachers associated with that year. If the entered course does not exist, the `ERR_NOT_EXIST` error will be emitted, and the user will be prompted again to input the value showing the same message as above. No further checks need to be performed on the input year.

6.3 Add teacher

This option allows the creation of a new `Teacher`, using the function `addTeacher` that you will have to implement. This function is activated when the user chooses option `3` from the main menu. First, the user will be prompted to input the academic year to which the teacher is assigned with the following message:

Terminal

```
Enter academic year:
```

If the entered name is an empty string, the ERR_EMPTY error will be emitted, and it will return to the main menu. If the entered academic year does not exist, the ERR_NOT_EXIST error will be displayed, and the user will be prompted again to input the year showing the same message as above. If it is correct, the new teacher should be stored in the listTeachers vector of the corresponding AcademicYear structure once all their data has been filled in, as described below.

Secondly, the user will be prompted to input the name of the teacher with the following message:

```
Terminal
  Enter teacher name:
```

If the entered name is an empty string, the ERR_EMPTY error will be emitted, and it will return to the main menu. If a teacher with that name already exists, the ERR_DUPLICATED error will be displayed, and the user will be prompted again to input the name showing the same message as above. If the name is correct, it will be stored in the name field of the Teacher structure, and the user will be prompted to input the nickname with the following message:

```
Terminal
  Enter nickname:
```

The entered value will be stored in the nickname field of the Teacher structure. This information is optional, so the user can input an empty string (in which case an empty string will be stored in nickname). There can be more than one teacher with the same nickname, so it is not necessary to check for duplicates in this field.

Next, the user will be prompted to input the name of the subject taught by the teacher with the following message:

```
Terminal
  Enter subject:
```

This information will be stored in the subject field, allowing a maximum of 49 characters. If the user inputs a string longer than this, the program should truncate it to fit into the field. If the string is empty, the ERR_EMPTY error will be displayed, and it will return to the main menu.

Finally, the user will be prompted to input the rating of the teacher, which will be an integer between 1 and 5, and will be stored in the rating field. For this, the following message will be displayed:

```
Terminal
  Enter rating:
```

This field is not mandatory to fill, and it can be left empty, in which case the value 0 will be stored in rating. If a value is entered, it should be checked to be between 1 and 5. In any other case, the ERR_RATING message will be displayed, and the value will be prompted again showing the previous message.

6.4 Delete teacher

This option allows the deletion of an existing teacher, using the deleteTeacher function that you need to implement. This function is activated when the user chooses option 4 from the main menu. First, the user will be prompted to input the name of the teacher they wish to delete with the following message:

```
Terminal
Enter teacher name:
```

If the user inputs an empty string, the `ERR_EMPTY` error will be emitted, and it will return to the main menu. If a teacher with that name exists, it will be deleted from the `listTeachers` of the associated academic year along with all its information. Otherwise, the `ERR_NOT_EXIST` error will be emitted, and the user will be prompted again to input the name showing the same message as above.

6.5 Show teacher

This option allows displaying all the information stored for a teacher by implementing the `showTeacher` function. This function is activated when the user chooses option `5` from the main menu. First, the user will be prompted to input the name of the teacher for whom they want to display data with the following message:

```
Terminal
Enter teacher name:
```

If the user inputs an empty string, the `ERR_EMPTY` error will be emitted, and it will return to the main menu. If a teacher with that name does not exist, the `ERR_NOT_EXIST` error will be emitted, and the user will be prompted again to input the name showing the same message as above. If the entered name is correct, all the teacher's information will be displayed with the following format:

```
Terminal
Academic year: 2023
Name: Eduardo Smith
Nickname: Cocobongo
Subject: Metafisica II
Rating: ****
Phrase 1 (2021-2-10) [8]: Imaginemos una vaca esferica
Phrase 2 [7]: El mejor ejemplo de nihilistas lo podemos encontrar en un gimnasio
Phrase 3 (2021-2-14): Tolkien es como Rambo, pero en profesor de Oxford
Phrase 4: Hasta yo tengo mi parte malvada. Voy a Ikea y me llevo el lapiz
```

The teacher's rating will be displayed using asterisks. For example, a rating of `4` will imply displaying four asterisks (`****`). If the `nickname` or `rating` field was left empty when entering the teacher's data, that line of information will not be displayed in the output.

The epic phrases will be displayed one below the other, numbered starting from `1`, followed by the date on which they were said in parentheses, formatted as `year-month-day`, then their `rating` within brackets, followed by a colon and the text of the phrase. As seen in the second phrase of the previous example (`Phrase 2`), if the `date` field were empty (`day`, `month`, and `year` are `0`), the parentheses will not be displayed. As shown in the third phrase (`Phrase 3`), if `rating` were empty, the brackets would not be displayed. In the case of the fourth phrase (`Phrase 4`), neither the date nor the rating will be shown.

Once the teacher's information is displayed on the screen, it will return to the main menu.

6.6 Add phrase

This option allows creating and adding a new epic phrase to a teacher by implementing the `addPhrase` function. This function is activated when the user chooses option `6` from the menu. First, the user will be prompted to input the name of the teacher for whom they want to add a phrase with the following message:

```
Terminal
  Enter teacher name:
```

If the user inputs an empty string, the `ERR_EMPTY` error will be emitted, and it will return to the main menu. If there is no teacher with that name, the `ERR_NOT_EXIST` error will be emitted, and the user will be prompted again to input the name showing the same message as above. If the entered name is correct, the user will be asked to enter the text of the phrase with the following message:

```
Terminal
  Enter phrase:
```

If the string is empty, the `ERR_EMPTY` error will be emitted, and it will return to the main menu. If the phrase has content, it will be stored in the `text` field of the `Phrase` structure, and the user will be prompted to enter the date it was uttered in the *year-month-day* format with the following message:

```
Terminal
  Enter date (year-month-day):
```

To simplify the code, we will assume that the entered date always has the correct format and it is not necessary to check it. This field is not mandatory, and the user could input an empty string, in which case the `day`, `month`, and `year` fields of `Date` will be initialised to `0`.

Finally, the user will be prompted to enter the rating of the phrase with the following message:

```
Terminal
  Enter rating:
```

This field is not mandatory, and it can be left empty, in which case the value `0` will be stored in the `rating` field of the `Phrase` structure. If a value is entered, it must be checked to be between `1` and `10`. If it is not correct, the `ERR_RATING` message will be displayed, and the user will be prompted again to input the value showing the same message.

After entering all this information into a `Phrase` structure, it will be added to the list of phrases (`listPhrases`) of the corresponding teacher, and it will return to the main menu.

6.7 Summary

This option will display a summary of all the stored phrases in the program. To show this information, you have to implement the `summary` function. It is activated when the user chooses option `7` from the main menu. The summary will display the phrases ordered by academic year, from most recent to oldest, with the following information:

- Name of the teacher who said it
- Rating given to the phrase
- Text of the phrase

All the information of a phrase will be displayed on the same line, separating the fields by a hyphen. If the rating is not present, that information will not be shown in the summary, as can be seen in the third and fourth phrases of `Eduardo Smith` in the example of output below:

Terminal

```
Academic year: 2023
Eduardo Smith - 8 - Imaginemos una vaca esférica
Eduardo Smith - 7 - El mejor ejemplo de nihilistas lo podemos encontrar en un gimnasio
Eduardo Smith - Tolkien es como Rambo, pero en profesor de Oxford
Eduardo Smith - Hasta yo tengo mi parte malvada. Voy a Ikea y me llevo el lapiz
Juan Pereza - 5 - Donde estaba yo el dia que dios repartio la paciencia
Academic year: 2022
Maria Grande - 9 - Tranquilo, aprobaras, pero no hoy. Sigue intentandolo
```

- For an academic year, teachers and phrases are displayed in the same order as they were entered into the system
- If an academic year has no teachers assigned, it will not be displayed