MTH 4300
Spring 2025
E. Fink

Homework 9

Create a program that automatically reads test questions from .txt files, and creates from them a test that can be taken and automatically graded.

There will be three types of questions, each stored in their own format.

- *Short answer questions* will simply display a question, and wait for the user to type in an answer. They will be stored in files with the format of `q1_sa.txt`: these files will have two lines, with the first line containing the text of the question, and the second line containing the correct answer.
- *Fill in the blank questions* will display a question with a number of blanks, and then prompt the user to enter an answer for each blank. These questions will be stored in files with the format of `q2_fitb.txt`: the first line of these files will contain the text of the question, and the remaining lines will contain the correct answers for each blank.
- *Multiple choice questions* will display a question, along with choices numbered 1, 2, ...; the user will be prompted to answer the number of the correct choice. These questions will be stored in files with the format of `q3_mc.txt`: the first line of these files will contain the text of the question, the second line will the number of the line below which contains the correct choice, and the remaining lines will contain the various answer choices to be displayed.

As part of your solution, you will create five classes: `Question`, `ShortAnsQ`, `FillInQ`, `MultChoiceQ`, and `Exam`; the middle three classes will inherit from `Question`.

The base class `Question` will contain two `protected` member variables:

- `string question_text`
- `string correct_answer`

and support the following two `protected` member functions:

- `virtual void display()`, which prints out the question;
- `virtual string accept_response()`, which runs code that waits for and accepts the testtaker's response, and returns that response;

and the following `public` member function:

- `bool execute()`, which calls the previous two member functions, and returns whether or not the latter return value is equal to `correct_answer`;
- a default constructor, which sets `question_text` and `correct_answer` to be empty strings.

The first class which inherits (publicly) from `Question` is `ShortAnsQ`. This class should simply have one `public` method (outside those inherited):

- the constructor `ShortAnsQ(string f)`, which opens a file whose file name is the argument `f`, and sets `question_text` and `correct_answer` accordingly (based on the file format described above). For this constructor, review input filestreams, as well as the function `getline()`.

After the first two classes are implemented, try running `test1.cpp`. For this, you will want to execute

```
g++ question.cpp shortansq.cpp test1.cpp
```

or

```
clang -std=c++17 -stdlib=libc++ question.cpp shortansq.cpp test1.cpp
```

and after that, run `./a` or `./a.out`.

The second class which inherits from `Question` (publicly) is `FillInQ`. This class will have a new `private` member:

- `int num_blanks`, the number of blanks to be filled in

and it will implement two methods:

- a constructor `FillInQ(string f)`, which opens a file whose file name is the argument `f`, and sets `question_text` accordingly. As for the `correct_answer`, my suggestion is that you loop through the file, and concatenate the individual correct answers into one long string. (Alternatively, you could have this class hide the `correct_answer` member variable.) The loop should also set `num_blanks` appropriately.
- a `protected` member function `virtual string accept_response()` which hides the version implemented in `Question`. This method should print out requests to fill in each blank, using the `num_blanks` member, and should take the users inputs and concatenate them in the same way that the answers were concatenated in the constructor; this concatenation is then returned.

Now, try running `Test2.cpp`.

The last class which inherits from `Question` (publicly) is `MultChoiceQ`. This class will have a new `private` member:

- `vector<string> choices`, a list of answer choices

and it will implement two methods:

- a constructor `MultChoiceQ(string f)`, which opens a file whose file name is the argument `f`, and sets `question_text` accordingly. The second line in the file will be set to be the `correct_answer`, which will be an integer, corresponding to which line in the file contains the actual correct answer. The subsequent lines contain the answers to be displayed; these will be stored in `choices`.
- a `protected` member function `virtual void display()` which hides the version implemented in `Question`. This method should, in addition to printing out the question contents, print out the answer choices each on their own line, preceded by `1)`, `2)`, etc.

Now, try running `Test3.cpp`.

Now, create an `Exam` class. This class will contain one `private` member variable:

- `vector<Question*> question_list`, a list of pointers to the questions in the exam;

and support the following two `public` member functions:

- `void add_q(Question*)`, which simply adds a reference to a question object to `question_list`;
- `void run()`, which asks the `Questions` in `question_list`, and prints out the percentage that were answered correctly.

Now, in `main.cpp`, create an exam containing all 5 questions contained in the various .txt files.

Please fill out class declarations in the appropriate .h files, and the class implementations in the appropriate .cpp files. For this problem, all constructors should be implemented in .cpp files, without initializer lists.