Capital Quiz

Stephen Gerkin
October 29, 2019
CIST 2362 CRN 22282

Program Description

A game is to be created that will quiz a user on United States state capitals. This will be implemented using a map containing a key/value pair with the key being the state and the value being the capital. The quiz will display the state to the user and then ask them to enter the name of the capital. The quiz will also be modular and allow the option to quiz based on country capital names.

Program Requirements

- Implement the question/answer information using a map container from the standard template library.
- Display the number of correct and incorrect answers.
- Display the time spent in the quiz.
- The user should be able to exit the program at any time.
- Question order should be randomly generated.

Input Validation

- Capitals can be entered without concern to case.
- Menu selections should only allow for specified entry.

Programming Strategy:

A QuizGame class will be created that quizzes a user on the selected information. This information will be provided using a QuestionMap class containing the question information and map. A wrapper for these to allow selection between state capital and country capital selections will provide the entry point for starting a selected quiz.

QuestionMap Class Documentation

Creates a map of key/value pairs with the Key being a question field to ask and the Value being the correct answer for the question.
It can be constructed from a properly formatted file or from a map object.

Member Variables

Date Type	Name	Description
std::map< std::string, std::string >	questionAnswerMap	For storing key/val entries for question/answer information.
std::string	questionTypeName	The name of the question type.
std::string	answerTypeName	The name of the answer type.

Constructors and Destructors

QuestionMap::QuestionMap (std::ifstream & inputFile)

Main constructor using a file to construct a set of questions and information about the questions.

See convertFile documentation for information on how the file should be structured.

Parameters

inputFIle	The file from which we are to retrieve information

QuestionMap::QuestionMap (std::map< std::string, std::string > map, std::string questionTypeName, std::string answerTypeName)[explicit]

Additional constructor allowing the creation of a question set from a map created by means other than reading a file.

Parameters

map	The map of question to answer key/val pairs.
questionTypeName	The field name for the questions (ie "State").
answerTypeName	The field name for the answers (ie "Capital").

Member Functions

bool QuestionMap::checkGuess (const std::string & questionPart, const std::string & guess)

Checks a given question against a guess within the map to verify the correct answer.

Allows for leading/trailing whitespace and different case entry.

Parameters

questionPart	The question we want to check.
guess	The guess we want to check.

Returns

True if the guess is equal to the val in the map given the question.

void QuestionMap::convertFile (std::ifstream & file) `[private]`

Converts a CSV file into a **QuestionMap** object.

Each key/val must be on a separate line and separated by only a comma. The first line of the file should be the question field type name and answer field type name.

Parameters

file	The file we wish to convert to a **QuestionMap**.

std::string QuestionMap::getAnswer (const std::string & question) const

Gets a value from the map containing the question information given a question.

Parameters

question	The question we want the answer for.

Returns

The answer to the question.

const std::string & QuestionMap::getAnswerTypeName () const

Getter for answer type name.

std::deque< std::string > QuestionMap::getListOfQuestionParts ()

Gets a double-ended queue of question pieces only.

Returns

A double-ended queue consisting of the map keys.

const std::string & QuestionMap::getQuestionTypeName () const

Getter for question type name.

std::pair< std::string, std::string > QuestionMap::parseLineIntoPair (const std::string & line) `[private]`

Parses an individual line of a file into a pair of entries.

Parameters

line	The line we wish to parse into two values.

Returns

A pair containing the two pieces of the line.

QuizGame Class Documentation

QuizGame takes a QuestionMap and quizzes the users on the information.

It keeps track of correct and incorrect questions in a separate double-ended queue. This may eventually be extended to allow the user to view a list of questions they got correct and/or incorrect. Statistics about the current quiz are displayed each time a question is asked. If the user wants to quit at anytime, they can type in the QUIT_STR constant and a **QuitException** will be thrown to signal the user's desire to quit the quiz.

Member Variables

Date Type	Name	Description
QuestionMap	questions	For storing the questions to ask the user.
const std::string	QUIT_STR	The string the user should type to quit.
std::deque<std::string>	questionsToAsk	For storing the queue of questions to ask the user.
std::deque<std::string>	answeredCorrectlyList	For storing the correctly answered questions.
std::deque<std::string>	notAnsweredCorrectlyList	For storing incorrectly answered questions.

Constructor & Destructor Documentation

QuizGame::QuizGame (QuestionMap questions) `[explicit]`

The constructor initializes the **QuestionMap** we are working with and starts the quiz.

Parameters

questions	The questions we are to use for the game.

Member Function Documentation

void QuizGame::answeredCorrectly (const std::string & question) `[private]`

Handles correctly answered questions.

Parameters

question	The question the user answered correctly.

void QuizGame::askQuestion () `[private]`

Asks the next question in the queue and moves it to the appropriate correct or incorrect queue after answered.

void QuizGame::displayStartMsg () `[private]`

Displays the starting message information for the quiz.

bool QuizGame::getYesNo () `[private]`

Gets yes or not input from the user.

Returns

True if yes.

std::string QuizGame::guess () `[private]`

Gets a guess from the user.

Returns

The user input.

void QuizGame::notAnsweredCorrectly (const std::string & question) `[private]`

Handles incorrectly answered questions.

Parameters

question	The question the user answered incorrectly.

void QuizGame::prepareQuestions () `[private]`

Gets our list of questions and shuffles them using a Mersenne twister algorithm.

void QuizGame::quit () `[private]`

Throws an exception to signal the user wishes to quit the quiz.

void QuizGame::show (const std::string & msg) const `[private]`

Shows a given string to the user.

Parameters

msg	

void QuizGame::showCurrentPlayTime () `[private]`

Displays the current time played.

The time played may be off by at most 1 second any time this function is called but this is not cumulative as the amount of time played is recalculated every time this function is called.

void QuizGame::showStatistics () `[private]`

Shows the current game statistics including time played and number of questions answered correctly/incorrectly.

void QuizGame::start () `[private]`

Notes the beginning time of the quiz and asks questions until there are no more questions to ask.

Program Demonstration Functions

void countryCapitalQuiz ()

Initializes a **QuizGame** with country capital quiz questions and starts it.

void determineQuizType ()

Gets the type of quiz the user wants to attempt and starts that quiz.

void displayMenu ()

Displays a menu of options to the user.

int getMenuEntry ()

Gets a menu selection from the user as an integer value.

Returns

The selection as an integer.

bool getYesNo ()

Gets a yes or no entry input from the user.

Returns

True if yes.

void initAndGo ()

Main entry point for demonstrating the **QuizGame** class.

Contains methods for demonstrating the **QuizGame** class.

Asks the user to select a quiz type.

void invalidSelection ()

Displays information if a selection is invalid.

void startQuiz (const QuestionMap & questions)

Starts a quiz with the given questions.

Parameters

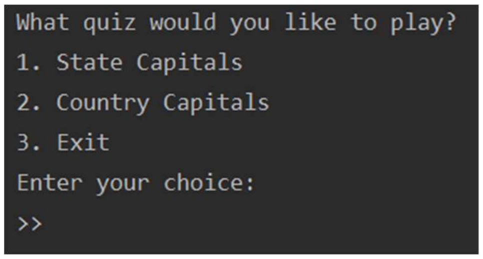
questions	The questions we want to start a quiz with.

void stateCapitalQuiz ()

Initializes a **QuizGame** with state capital quiz questions and starts it.

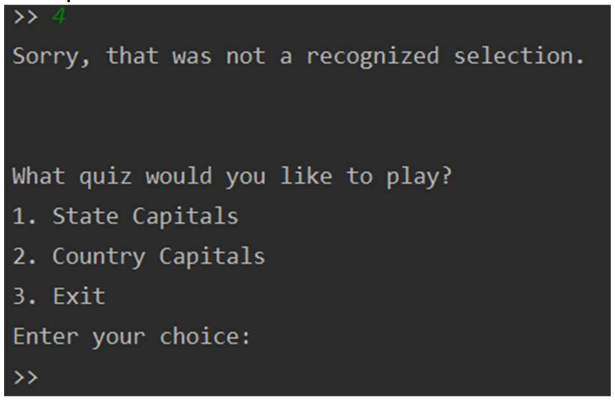
Input and Output Demonstration

Start Screen:



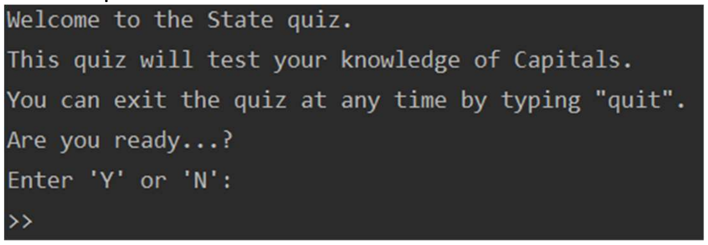
```
What quiz would you like to play?
1. State Capitals
2. Country Capitals
3. Exit
Enter your choice:
>>
```

Bad quiz selection:



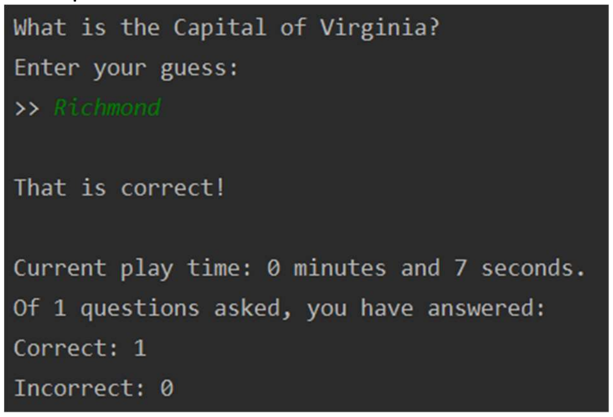
```
>> 4
Sorry, that was not a recognized selection.


What quiz would you like to play?
1. State Capitals
2. Country Capitals
3. Exit
Enter your choice:
>>
```

State Capital Quiz:



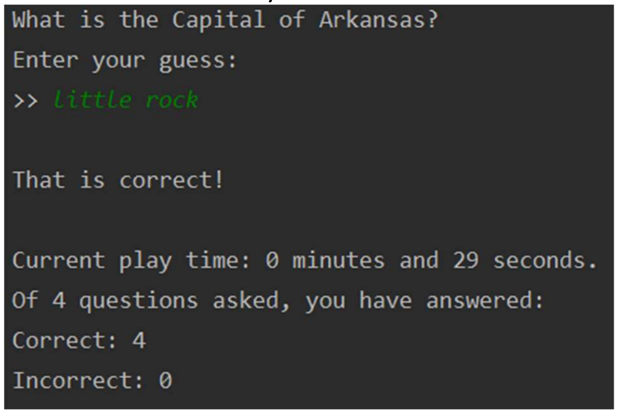
```
Welcome to the State quiz.
This quiz will test your knowledge of Capitals.
You can exit the quiz at any time by typing "quit".
Are you ready...?
Enter 'Y' or 'N':
>>
```

First question:



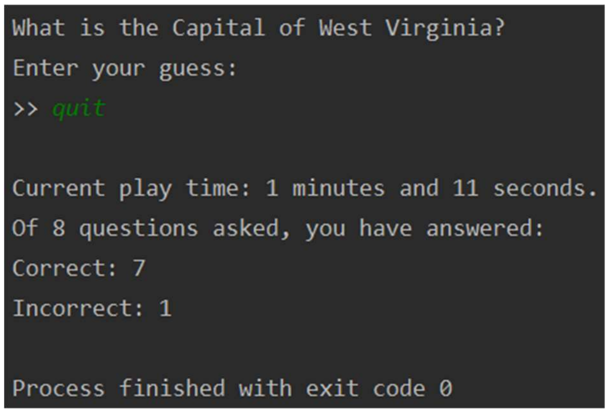
```
What is the Capital of Virginia?
Enter your guess:
>> Richmond

That is correct!

Current play time: 0 minutes and 7 seconds.
Of 1 questions asked, you have answered:
Correct: 1
Incorrect: 0
```

Allow case insensitivity:



```
What is the Capital of Arkansas?
Enter your guess:
>> little rock

That is correct!

Current play time: 0 minutes and 29 seconds.
Of 4 questions asked, you have answered:
Correct: 4
Incorrect: 0
```

Quit:



```
What is the Capital of West Virginia?
Enter your guess:
>> quit

Current play time: 1 minutes and 11 seconds.
Of 8 questions asked, you have answered:
Correct: 7
Incorrect: 1

Process finished with exit code 0
```