CPSC 1301, Computer Science I Programming Exercise

Weeks 11 and 12, HistoryTest.py

1 Introduction

This activity consists of two programming parts. The following exercises are open book and open note. You are free to use any written documentation you wish. However, these are individual exercises, and you cannot consult with each other in writing your programs. This exercises consists of two parts and three files. Part 1 is the current week's exercise. Part 2 is next week's exercise. Name this week's file `test_maketest_lastname.py`. Name next week's file `test_givetest_lastname.py`. Name your "main" module `HistoryTest_lastname.py`.

2 Exercise requirements

This exercise consists of building a history test. To initialize a session, you will ask the users the number of problems they wish to answer. You will echo the number of problems to the user.

When the user begins the test, the application will present the specified number of problems of the specified type. The problems will generate random questions from the test bank. The application will keep score, and if the user misses a question the application will display the correct answer to the problem. At the conclusion of the session the application will report the number of correctly answered questions and give a numeric score.

The test is initialized by reading a series of questions from a CSV file. The file format has five columns. Column 1 is a question. Column 2 is the correct answer. The remaining three columns are distractors, incorrect answers. Ordinarily the data would be read from a database, but reading from a CSV file simplifies the application.

Please note the following: (1) The number of questions on the test cannot exceed the number of questions in the database. (2) The test questions *must* be randomized. (3) The suggested answers *must also* be randomized.

2.1 Part 1 - test_maketest.py

This module consists of the following functions.

hello() This function takes no arguments, returns no value, and prints the name of the module — it's just a sanity check.

initialize_test_bank(filename) This function takes a filename as a string (or variable), and returns a list of lines, each line of type string and representing one line in the input file. The output of this function will be used as the input for `initialize_test`.

initialize_test(line_list This function takes a list of lines (as strings) produced by `initialize_test`, obtains user input of an integer specifying the number of questions the user wants to answer, and returns a randomized list of lines taken from the test bank. Note that the output list must be the same length that the user specified, and must consisst of randomly drawn questions from the test bank. The output will be used as input for `structure_test`.

structure_test(test) This function will take as input the randomized list produced by `initialize_test`, and produce as output a list of three element tuples. The first element will be the question. The second answer will be the correct answer. The third element will be a *randomized* list of the correct answer and the three distractors. It will look like this:

```
#note that the third element is randomized
(question, correct, (wrong4, correct, wrong1, wrong2))
```

(optional) randomize_answers(answers) This is a helper function that takes a tuple of answers (third element of the above tuple) and randomizes the list, returning the list. This is called within `sturcture_test` as a convenience.

pres_test.csv This is the CSV file containing the questions and answers.

2.2 Part 2 - test_givetest.py

hello() This function takes no arguments, returns no value, and prints the name of the module — it's just a sanity check.

give_test(atest) This function takes as input the output of `structure_test` and presents the test to the user question by question. It keeps track of the correct answers and tells the student the correct answer for any question the student misses. It returns a tuple consisting of the number of questions answered correctly and the total number of questions.

ask_question(n, question) This function has the responsibility for presenting each question to the student. the arguments are the number of the question (e.g., 1, 2, 3, . . .), and the question typle. It prints the question, the four choices, prompts the student for an answer, and checks the answer. If the user's answer is correct, it return a 1. If the answer is not correct, it reports the correct answer to the user and returns a 0.

calc_grade(correct, total) The arguments are the number answered correct and the total number of questions. It calculates the numeric and the latter grade and reports to the user. It does not return a value.

calc_letter_grade(n) The input is a double representing the numeric grade. The return value is the letter grade.

3 Main Program

Note that the main program imports the two modules on lines 8 and 9: `test_maketest` and `test_givetest`. Line 19 initializes a variable with the name of the CSVfile containing the test. Line 20 initializes the test bank, passing the file name as a parameter and returning the test bank. Line 21 initializes the test, passingin the test bank and returning the test. Line 22 formats the questions and answers for the test. Line 23 delivers the test to the user, and returns a tuple consisting of the number of questions correctly answered and the total number of questions. Line 24 calculates the numeric and letter grades and reports to the user.

```
1  #!python
2  # Name: HistoryTest.py
3  # Author: Charles Carter
4  # Date; June 22, 2021
5  # Purpose: exercises modules test_maketest.py, test_givetest.py, and gives a history test
6
7  # import statements here (if any)
8  import test_maketest as mt
9  import test_givetest as gt
10
11
12 # define methods here (if any)
13 def hello():
14     print("Hello from 'HistoryTest.py'")
```

```

#main function executes the defined functions
if __name__ == '__main__':
    hello()
    csv_file = "pres-test.csv"
    test_bank = mt.initialize_test_bank(csv_file)
    test = mt.initialize_test(test_bank)
    mytest = mt.structure_test(test)
    (correct, total) = gt.give_test(mytest)
    gt.calc_grade(correct, total)
```

4 Sample output

```
This is a history test. How many questions to you want to answer? Enter a number from 1 to 21
3
You are about to take the test. There are 3 questions. To begin, type any key.
<Enter>

Question 1. Who said: Ask not what your country can do for you - but what you can do for your country?
1 Theodore Roosevelt
2 Ulysses Grant
3 Franklin Roosevelt
4 John Kennedy
What is your answer: 4
Correct

Question 2. Which president was never impeached?
1 Donald Trump
2 Richard Nixon
3 Andrew Johnson
4 Bill Clinton
What is your answer: 1
Incorrect. The correct answer is Richard Nixon

Question 3. Who was president when the British burned the White House?
1 Andrew Jackson
2 James Madison
3 Joe Biden
4 James Monroe
What is your answer: 2
Correct

You made a 66.67 which gives you a D
```