Documentation for the Learning App

Harshitha Jeyakumar, Ramya Nataraj, Jessica Ononye, Tatyana Lim, Nivitha Paranthaman

Purposes of Each File:

I. learning_app_math.py

The purpose of the Math.py file is to run the Math component of the Learning App. When a user chooses to practice math, the Math.py stores the functions necessary to start the practice. In the file, there is the MathQ class, which stores all the functionality to create math practice questions. Based on the grade of the user and the amount of questions they want to practice, the MathQ class generates the appropriate amount of questions at the appropriate level of difficulty. It then outputs each question to the user, stores the user's answers, and based on the amount of questions they get right, a score is generated.

II. learning_app_grammar.py

Learning_app_grammar.py is a file that runs the grammar component of the learning app. Once a user gives their name and grade, they can choose grammar to practice with. The app then prompts the user to enter a sentence, with this sentence, this program analyzes for punctuation, capitalization, and word count. Based on the errors made or not made, the program gives them advice on how to improve their grammar and motivates them to keep practicing. These errors are stored in a list which is then outputted at the end of their attempts along with all the errors they had made along the way.

III. learning_app_vocab.py

The learning_app_vocab.py file runs as the vocabulary component of the learning app. Upon entering their name and grade in the learning app's interface, the user can select to practice vocabulary. The contents of the learning_app_vocab.py file are implemented in the LearningApp.py file to initiate the vocabulary practice. The learning_app_vocab.py file contains the Vocab class, which consists of five methods that work together to generate the necessary vocabulary questions and the user's score at the end of the practice. Each question is set up as a multiple-choice question and the difficulty of each question has been adjusted according to the user's grade level.

IV. learning_app_user.py

The purpose of the LearningApp.py file is to create and run the User parts of our app. It will prompt the user for answers to various introductory questions, including asking them for their name, their grade (from Kindergarten to 2nd Grade), and the subject they will like to focus on, from a couple of options, grammar, math, and vocab. In the file the User Class initializes the User with their name, subject chosen, and grade chosen. The grade is also edited through a function to a more compatible format for all classes. Based on the subject choice made by the user, a function will call other functions that will run instances of the other classes for each subject, Math, Grammar, and Vocab.

V. file_learning_app.py

The file_learning_app.py contains a Summary class that generates output files containing statistics and graphs of the user's playthrough of the learning app. Depending on what subject the user chooses to practice, the Summary class has methods to generate a bar graph for the math, vocabulary, or grammar section. There is also a stats method that generates a text summary for the user. The summary varies for different subjects, but some things included in the summary are the number of questions answered by the user, the number of questions that were answered correctly or incorrectly, and their final score. The results_folder method creates a folder in a directory that the user specifies. The folder is populated with a txt file of the summary statistics and a .png file of the bar graph, with file names given by the user.

How to Run the Learning App:

MacOs:
In the command line, type the following command: python3 Learning_App.py
Once you do so, you will be prompted to enter your name, choose the grade level of questions you want to practice in, and choose the subject you want to practice from the options of grammar, vocabulary, and math. The questions will then be generated and the user will be prompted to answer each question.

How to Interpret Results:

Once a user answers questions in 3 attempts, they will be provided with a summary of their results. Each result is formatted as a list, with lists within of the correct/expected answers, answers given by the users as a list, a score that was calculated, and the number of attempts they took. Depending on the subject that was chosen, each of the results is formatted differently. The user will also be prompted to enter a file path, where a folder will be created with additional information about their results through a generated text file with statistical information and a barplot. If the user selects the math subject, the barplot will display the percentage of questions they answered correctly for each attempt. The statistics file will display the number of questions

answered, the number of attempts, the correct and incorrect answers, and the final score. If the user selects the vocabulary subject, the barplot will display the percentage of total questions that are correct and the percentage of total questions that are incorrect after each question. The statistics file will display the number of questions answered, the correct and incorrect answers of the user, and the user's score. If the user selects the grammar subject, the barplot will display the number of each grammatical error type for each sentence they input. The statistics file will display the number of sentences the user inputs, the percentage of each type of grammatical error, and the total errors.

Attribution:

Method/Function	***Primary Author***	***Techniques Demonstrated***
MathQ.__init__	Harshitha Jeyakumar	Optional Parameters
MathQ.math_questions	Harshitha Jeyakumar	List Comprehensions
User.grade_level	Ramya Nataraj	Regular Expressions
math_choosen	Ramya Nataraj	Composition
vocab_choosen	Ramya Nataraj	Composition
Vocab.user_answers	Tatyana Lim	Conditional expressions
Vocab.vocab_generator	Tatyana Lim	Use of json.load()
Grammar.punctuation	Jessica Ononye	Sequence Unpacking
Grammar.error_count	Jessica Ononye	F-strings with expressions
Grammar.word_count	Jessica Ononye	F-strings with expressions
Summary.bar_visual_math	Nivitha Paranthaman	Visualizing data with pyplot
Summary.bar_visual_vocab	Nivitha Paranthaman	Visualizing data with pyplot
Summary.bar_visual_grammar	Nivitha Paranthaman	Visualizing data with pyplot
Summary.results_folder	Nivitha Paranthaman	with statement to print to file