DeutschLerner Project Goals

Corbin Foucart

1 Interface & GUI

Rough Ideas:

1. 2-3 pages: menu and quiz page

1.1 Menu

The idea here is that the user has the ability to access all of the features of the application from this menu. Perhaps some basic stuff about recently taken quizzes and stats. The menu should be a landing page that allows the user to move to other pages. Page Structure:

1. Menu
   (a) Take quiz page
      i. Right now let's focus on building something that just tests the user on all of the questions in the database. It might be better for applying the memory algorithms.
      ii. This would be some type of gui page that simply has a text field for the question and allows the user text boxes to type the answers. For genders, this might include two text boxes. For verbs, I can imagine this consisting of a single text field for regular verbs and several text fields for irregular verbs.
   (b) Database editing

1.2 Quiz

2 Underlying Structure

The idea is that all data can be appropriately saved to comma separated value files and later read in by the program.

2.1 Question structure

There should ideally be several types of question: nouns [article + noun + plural], adjectives [word], regular verbs [word], irregular verbs [3 verb forms], verb / preposition combos [infinitive + preposition]. Delimiters for each value are as follows:

1. Noun - N : "der Stock, Stöcke", use "-" for when no plural exists
2. Regular Verb - RV : "rudern"
3. Irregular Verb - IV
4. Adjective - A
5. Preposition - P

6. Preposition Verb combo - VP
7. Phrase - P

This will be achieved by keeping a comma separated value sheet. Each line will represent a single question:

[question type delimiter], [question grouping number], [question string], [answer string]
The question string will always be displayed. The answer string will be parsed and will quiz the user appropriately.

3 Probabilistic Question Choosing

There are two relevant arrays, a questions array and a completed array. At the start of the quiz, all questions are in the questions array. Even past questions with scores above the cuttoff threshold will be asked once. Once a question exceeds some criterion, it moves into the completed array, and can not be asked again during the course of the quiz.

Within the question array, each question is assigned a score based on how well the user has answered the question in the past. A question starts at score 0 and gets closer to 1 (infinity?) as the user answers the question correctly. Each score maps to a weighted porability for the question, so that questions with scores closer to 0 are more likely to be asked.

The individual unweighted probabilities (don't sum to 1) are calculated:

$$w_i = 1 - s_i$$

Then we have that:

$$p_i = \frac{w_i}{\sum_j w_j}$$

Now the individual probabilities sum to 1. We can create an array corresponding to the question array, with the cumulative sum. Each entry corresponds to the 'cutoff' for each question. Now a random number is generated between 0 and 1, inclusive. The array is searched for the element of the list containing a number greater than the random number generated. The element preceeding that one corresponds to the correct question to choose.

4 Building Goals

1. build csv writer
2. build csv reader
3. build csv merger
4. combine the three into a console driven database addition program
5. write a parser that can parse things into the question text fields
6. write a console based program that just displays the stuff without user input.
7. write the quizzing program with the math integration. Should be totally a console program at this point.
8. design and write the gui. Integrate it into the project
9. color the gui to look cool.