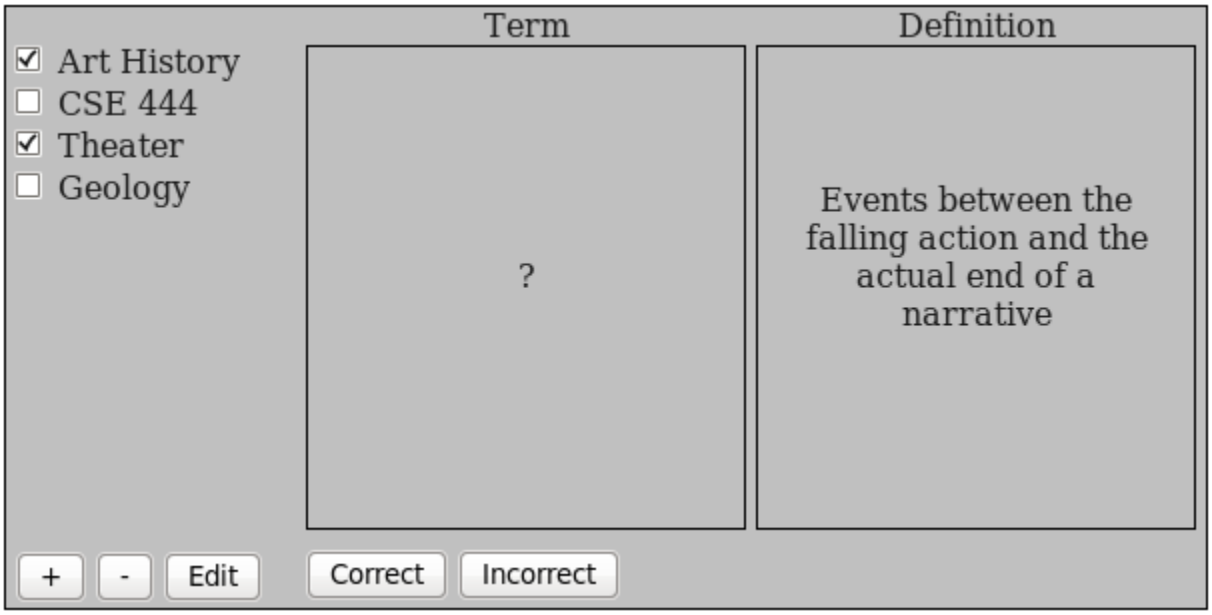
John Chilton (jchilton) and Hilary Worden (hkworden)
CSE 403
Assignment 1

Product

The product would be a desktop application for using flash cards. Many people find flash cards to be a useful method for learning a variety of things, but they have drawbacks: it is not easy (or cost-efficient) to sort cards based on how well you know them (so you often spend more time than you need on some facts, but less than you need on others), and the information you can study is limited to what you can fit in a fairly small two-dimensional space.
Rather than just being a digital representation of a box of paper cards though, the program would keep record of how often a user knew the answer, and would use that data to show cards more or less frequently based on how well the user already knows each one. Cards that the user consistently knows would be showed rarely, while cards that the user never knows would come up often. Additionally, the program would have a text-to-speech feature, so that a card could include audio questions or answers; this could be helpful to students of foreign languages. The program would also allow users to import cards from a CSV file, let the user assign tags to cards (so that users could focus on a particular class or topic), and possibly use a web service through which users could share cards with each other.



Target Customers

The program would mainly be useful to students, although anyone interested in learning information that can be represented in question-answer form could use it as well.

Alternatives

There are a large number of flash card programs available for various platforms. However, few offer text-to-speech (the only one we were able to find was an Android app, and it does not appear to allow use of different languages on a card-by-card basis). Additionally, many flash

card programs simply iterate through the set of cards, rather than keeping track of how often the user knows the answer and using that information to show more difficult cards more frequently.

Software Architecture

Implementation

The program would probably be written in Java, and use a model-view-controller design. SQLite (with JDBC) would be used to store cards and guess records, with a class that manages database access to keep the model cleanly separated from the controller. Swing would likely be used to implement the GUI, and the FreeTTS library would be used to implement the text-to-speech feature.

Interesting Technical Matters

The algorithm for deciding what card to show next could potentially be very complex. A simple system (such as counting correct/incorrect guesses for each card, and only showing cards with a correctness percentage below a certain value) could probably be implemented fairly easily. However, the system could also be quite complicated: for instance, the system might keep an entire table of guesses, tracking what day each guess was made and whether it was correct or not. Then, it could base its decision on only the last week of data, or could consider whether the user's correctness percentage for the card was declining.

Using the FreeTTS library might also present technical challenges, especially since the modules for different languages might come from different sources (such as the MBROLA project).

Challenges and Risks

FreeTTS, and especially support for other languages, might take time to set up. It would therefore be good to start at least experimenting with FreeTTS early on, or even to develop the FreeTTS-related components as early as possible.