Texas Instruments - Carnegie Mellon

Human Computer Interaction

Final Project Report

August 2, 1999

Table of Contents

Overview

This report details the process and progress of the Texas Instruments Blue Team as part of the Carnegie Mellon University HCI Master's Project.

At the start of the project, the team was directed by Texas Instruments to find uses for TI-83 Plus graphing calculators in non-math and -science high-school classrooms. We gathered initial information about the activities and needs of high school teachers and students in the humanities through interactions which took place in the context of their daily routines. The analysis of this information steered the team toward the creation of a Flash Card Viewer application for the calculator, and a companion Flash Card Maker application for personal computers running Microsoft Windows. Early prototypes of the two applications were constructed; upon which the team conducted iterative design and user testing with high school students over the span of six weeks during the summer of 1999.

This report details:

I. An overview of work conducted over the last two semesters at Carnegie Mellon, including:

- Background research that guided the design of the Viewer and Maker applications
- Initial stages of design and testing
- User testing procedures, and some results

II. A detailed description of the functionality of the two final applications:

- Flash Card Viewer (calculator)
- Flash Card Maker (PC)
- Suggestions for future work

PART I: Project Summary

Background Research

The focus for this project was to find uses for graphing calculators in non-math and -science areas of high-school curriculum and classrooms. As part of the initial data gathering phase, we conducted observations and talked with high school teachers and students in the context of their normal activities. We called this technique 'contextual interviewing'. Observing teachers in the process of teaching and students while at home studying enabled us to get a firsthand look at the kinds of tasks and sequences they complete on a daily basis. Please see the interim report issued in April of 1999 for a more detailed account of this phase of the project.

Over the course of several months, group members observed high school humanities classes while in session. Observations took place in public and private schools, and in a variety of subjects:

Public High School	Private High School
German (3 visits)	Composition
US History	US History (2 visits)
Biology	European History
Physics II	Drama
Calculus	Calculus

Group members also spoke directly with teachers and students:

5 Teacher Interviews
Government
Composition
Algebra
English
US History

3 Student Interviews

The information gathered in multiple interviews and observations was later examined for patterns and commonalties across separate interviews. The following statements summarize the issues that affected a majority of teachers and students in the humanities:

1. Students are comfortable exploring new uses for their calculators and like using the graphing functions.
2. Students used a variety of methods to complete their homework and to study for tests such as review of class notes, use of flash cards, and use of an electronic dictionary.
3. Non-math and -science teachers are wary of new technologies and do not want it to take up precious class time.
4. Classroom activities included but were not limited to data analysis, test taking, note taking, discussions, and debates.
5. Teachers were concerned about participation during class discussions. The atmosphere of the classes varied widely from enthusiastic chaos to subdued indifference.
6. Teachers used many different methods to transmit information to the class. These included overhead projectors, videotapes, audiotapes, roll-down maps, blackboards, whiteboards, bulletin boards, posters, and handouts.

Based on the preceding six general findings, the group brainstormed design ideas which would determine the direction of future work. The resulting list of ideas was culled to include only those which matched the focus well, had strong support from the primary data collected, and were suitable for implementation on the TI-83 Plus. The four major design ideas settled upon were 1) Adaptive Flash Cards, 2) Data Visualization Tool, 3) Searchable Reference, and 4) ClassNet. (For more details about each of these ideas, see the interim report distributed in April 1999.)

After receiving feedback from Texas Instruments, the group chose to pursue the Adaptive Flash Cards concept. This decision was motivated by several factors:

1. Most importantly, an overall goal of this project was to produce a high-fidelity prototype. This goal seemed most attainable with the Flash Cards concept because it did not require any highly complicated visual presentations of information.
2. There was strong support for the use of flash cards as study aids among high school students, and we wanted to provide them with an unspecialized application that could be useful in conjunction with any of their humanities classes.
3. There were some interesting design issues in translating a very physical form of studying into a virtual form. We wanted to explore the 'adaptive' nature of electronic flash cards, and what kind of advantages they could offer over paper.
4. Finally, it seemed that providing students with an opportunity to carry around fewer items at school implied that there would be one less thing for them to lose.

The group decided very early in the development of the Flash Cards concept that it was necessary to split the functionality we wanted to provide into two applications: the Flash Card Viewer for the calculator, and the Flash Card Maker for the PC. This decision was based on the apparent constraints of the TI-83 platform for its use in non-math and -science classes:

- Text input on the calculator is slow relative to a computer keyboard, and humanities classes involve a lot of text!
- The resolution of the screen limits the detail that can be displayed.
- Text display is approximately twenty words—eight lines of sixteen characters—about two or three words per line, even fewer if the words are long.

These constraints led us to conclude that the creation of electronic flash cards was better suited for a personal computer, and the calculator application should serve as the Viewer for flash cards created on a PC. After the user has finished creating the flash cards he can export the text to a file, which can then be imported into GraphLink and downloaded to the calculator.

Initial Steps

Initial steps guiding the creation of Flash Card applications for the PC and TI-83 Plus included informal field research into the use of flash cards, library research into psychological principles behind flash cards as study aids, and preliminary user testing to clarify early interface design issues.

Field Research

Our first step after deciding to pursue the Adaptive Flash Cards concept was to interview three students about their use of flash cards, and attempt to collect some flash cards that students around Carnegie Mellon University had created for their own use. We found that students both create their own study materials and also get study materials from teachers. If the teacher provides the students with good study materials, then students are less likely to create their own. They use these study aids to test themselves when studying for exams, in order to verify their knowledge of the material.

We asked several CMU students how they use flash cards when they study, and found that most make a mental and physical distinction between cards they know and cards they don't know. After looking at a flash card, students decide how well they know the material on the card. They will either set it aside because they feel they know it well enough that they don't need to return to it, or keep it in the stack of cards they are actively studying. The cards they are less sure of are thus studied more often.

Library Research

In order to assist us in deciding what functionality should be included, group members delved into the extensive psychology literature regarding learning. Several concepts relevant to the use of flash cards were explored in some detail:

1. Order Effects

It is an established principle that the order in which a person learns something affects their ability to recall it later. If a set of information is always studied in the same order, it will be harder to recall it if it is presented in a different order. To defeat this effect it is necessary to frequently change the order in which the material to be learned is presented.

2. Spaced Practice

Another commonly observed phenomenon is that recall for material that seems to be learned can be improved if that material continues to be presented on an irregular basis to the person studying. This supports the idea that even material students are confident they know should be repeated just for practice.

3. Feeling of Knowing

Feeling of knowing is a state that occurs when a person sees a word or concept with which they seem to be unfamiliar. The best way to describe it is, "…the state of believing that a piece of information can be retrieved from memory even though that piece of information currently cannot be recalled," (Miner & Reder, 1994). It is possible for this state to occur when a person views a flash card -- indeed any time information has to be recalled from memory.

4. Judgements of Learning

A 'Judgement of Learning' occurs when people are asked to predict "future memory performance of recently learned material," (Kelemen & Weaver, 1997). This prediction tends to be fairly inaccurate immediately after viewing the material, and increases until after a 5-minute delay the accuracy of the prediction is nearly perfect. This phenomenon has consequences for the use of flash cards, as the use of traditional flash cards involves just this kind of judgement when deciding which cards to keep in the stack and which are known well enough to set aside. However, even immediately following exposure to the material, judgements of learning are better than chance.

Preliminary User Testing

Preliminary user tests were informal. Paper mockups of the interface were created and placed on the calculator screen. The mockups consisted of a sequence of small, rectangular pieces of paper that fit directly on the calculator screen surface. Participants held the calculator in their hand and pressed the calculator buttons while the tester switched paper screens in response to the participants' actions. Only one paper screen was visible to the participant at any given time.

Participants were asked several questions about their interaction with the mockups, and these questions served as general guidelines for each user test. However, the tester also followed up on comments and questions from participants during the test. Issues addressed included affordance, screen placement, word choice and self-report. (For more information on preliminary user testing, see the interim report issued in June 1999.)

Design Decisions

Based on initial user test data as well as previous research, at this stage the design team made decisions regarding the interface of future prototypes:

- Mapping navigational tasks (flip, next card, previous card) onto the arrow keys, without explicit labels, is acceptable for users as long as instructions are provided.
- User comments indicate that as long as the front and back are visually distinctive in some way, labeling them explicitly is unnecessary.
- Having a separate screen for self-report was annoying for users, but putting the self-report in the status line didn't give novices enough information about the purpose of the self-report.
- Self-report is better than chance. Therefore it is an acceptable way to gauge learning, so it was not eliminated from the design.
- Learning support that is not possible using paper flash cards, such as random presentation of cards, should be supported.

Interface Evaluation

We employed two very different techniques for interface evaluation throughout the course of this project: Heuristic Evaluation and the Think-Aloud Usability Study. A discussion of each technique follows.

Heuristic Evaluation

According to Jakob Nielsen's website (Nielsen, 1999), "The goal of heuristic evaluation is to find the usability problems in the design so that they can be attended to as part of an iterative design process. Heuristic evaluation involves having a small set of evaluators examine the interface and judge its compliance with recognized usability principles (the 'heuristics'). " This technique actually uses two techniques in combination as it employs a team of evaluators and follows an established set of design heuristics. The evaluation team can be made up of three or more individuals not involved in the initial software design, but it does not include any end users of the system. The design heuristics can be thought of as general-purpose guidelines – we chose a set developed by Nielsen and Molich (1989) detailed below. For a detailed description of each of the heuristics, see Appendix B.

1. Visibility of system status
2. Match between system and the real world
3. User control and freedom
4. Consistency and standards
5. Error prevention
6. Recognition rather than recall
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. Help users recognize, diagnose, and recover from errors
10. Help and documentation

The above heuristics were applied to the Flash Card Maker application nearly halfway through its development cycle. The resulting user interface problems were corrected in the next version of the Maker.

Think-Aloud Usability Study

This technique involves the observation of real users interacting with an interface, whether it is a prototype or a full release. Users are asked to speak their thoughts aloud – to say everything that enters their minds as they are interacting with the interface. This procedure can reveal places in the interface where users are confused and frustrated, or happy and satisfied.

We embarked upon an iterative process of design and user testing in late June. Think-Aloud Usability Studies with high school students were scheduled each Monday and Thursday for a span of approximately six weeks. Changes suggested by a day's user tests were implemented for the next round of testing. We solicited high school students from several local high schools:

Males	18	(10 public, 8 private)
Females	8	(4 public, 4 private)

Testing took place in a usability lab on the Carnegie Mellon University campus equipped with three video cameras and video mixing equipment which enabled us to capture a picture-in-picture of the calculator screen and the users' physical interactions with the calculator simultaneously. The calculator display was projected onto a wall and the projected image was recorded so that the users' activities were not inhibited by attempts to capture what was displayed directly from the calculator screen itself.

Both the Flash Card Maker and Viewer applications were tested using the Think Aloud Usability Study technique. See Appendix C for an example user test protocol.

Feedback and Conclusion

The combined functionality of the Flash Card Maker and Flash Card Viewer provides the end user with several advantages over traditional paper flash cards.

Easily created, replicated and shared

The flash cards are easily created, replicated, and shared. Students can easily generate their own content from scratch or from pre-existing sources such as teacher handouts or end-of-chapter reviews printed in their textbook. Because the flashcards are digital and the calculators easily networkable, a student who writes a stack may share that stack of cards with any number of his classmates. He may even share his flashcards with other students over the WWW.

By enabling such a means of generating and sharing files, the Flash Card Viewer and Maker are poised to create a user community who regularly author and exchange flashcard content.

Enhanced learning support

The Flash Card Viewer is better able than traditional flashcards to present topics in an order and an amount that improve learning. Furthermore, we know from the educational literature that the very act of writing your own flashcards is an educational experience in and of itself. By allowing for multiple navigational methods within the application, the Flash Card Viewer supports multiple learning styles.

In Conclusion…

The Flash Card Viewer and Flash Card Maker are two useable and useful study tools. They have a clear application in supporting the study habits of high school students. Furthermore, students are already familiar with the concept of paper flashcards and are convinced of their utility. Finally, students like them:

"I think it is good. I will use this to study... Can you have this on the 82 or 85?" – high school student commenting on the Viewer application

"I like it... It looks cool." – high school student commenting on the Maker application

"Can I take this home?" – high school student's closing question

We recommend that TI continue developing these applications beyond the scope of this project. We feel that with another iteration of testing and development, the Flash Card Viewer and Flash Card Maker will become valuable tools for high school students in non-math and –science subjects.

PART II: Detailed Descriptions

Introduction

This section is meant to provide evidence for and explain the reasoning behind many of the interface decisions that lead us to the following designs. Our belief is that with both the working prototypes of the Flash Card Viewer and the Flash Card Maker and this document, whomever picks up this project will be able to produce the next version of this software.

For each design decision, we try to explain the direction we chose in three steps. First, we give a brief description of the feature. Next we explain the evidence for the feature (normally through user testing or another usability technique). Finally we try to point out any pitfalls we worked around during the course of our research so that future authors might avoid some of the mistakes we made.

Following the description of both the Viewer and Maker, we present a section of future work that includes questions and interesting issues which we were unable to address in the scope of this project. Any person continuing this project should give serious attention to the issues and concerns raised in this section, many of which came directly out of user's suggestions and subject observation.

Preceding the description of both the Flash Card Viewer and Maker, are brief user manuals to bring the reader up to speed with the applications' use.

Flash Card Viewer Manual

Starting the Flash Card Viewer

Start the Flash Card application by pressing **PRGM** and selecting **VIEWER**.
You will see the Viewer start menu with the following options:

USE LAST STACK
Retrieves the stack you were studying the last time you ran the application.
Note: This option will not appear when you run the application for the first time.

PICK NEW STACK
Allows you to select a new flash card stack to study.

INSTRUCTIONS
Brings up a short summary of key commands.

QUIT
Exits the flash card viewer

Using the Flash Cards

Once you start the flash card stack, it will be shuffled. That is, the order of the cards will be randomly mixed up. A screen will come up telling you the name of the stack and the number of cards in the stack.

After the cards are shuffled you will see the front side of the first card in the stack. The text of your flash card is at the top. The functions listed at the bottom can be accessed by pressing the blue buttons on the top row of the calculator (see Figure 1):

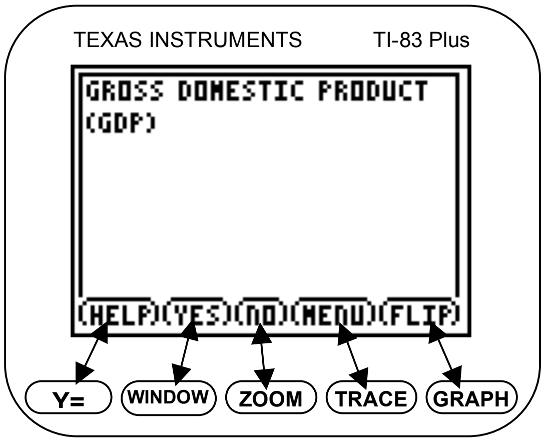


Fig. 1: Flash

Button...	**Means...**
Y=	HELP
WINDOW	YES
ZOOM	NO
TRACE	MENU
GRAPH	FLIP

- **HELP** brings up a short summary of key commands.
- **MENU** brings up a menu similar to the start menu.

The rest of the options are described in the next few sections.

Flipping the Card

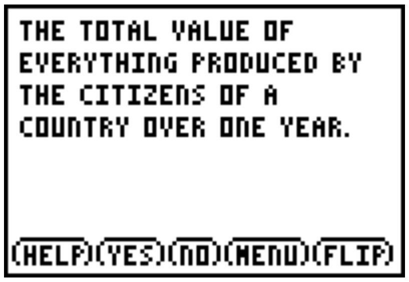
Like paper flash cards, the TI Adaptive Flash Cards have a front and a back. To see the back of the flash card you must "flip" the card over. There are two ways to flip:

- Press the **FLIP (GRAPH)** key on the top row of buttons.

or

- Press the **up arrow** (⇧) or **down arrow** (⇩) on the keypad.

After flipping, you will see the text on the other side of the flash card as in Figure 2.



THE TOTAL VALUE OF EVERYTHING PRODUCED BY THE CITIZENS OF A COUNTRY OVER ONE YEAR.

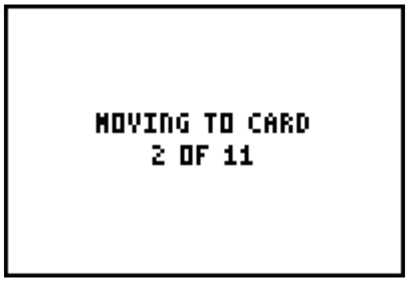
Fig. 2: Back of

Moving Between Cards

- To retrieve the next card in the stack, use the **right arrow** (⇨) on the keypad.

- To return to the previous card in the stack, use the **left arrow** (⇦) on the keypad.

When you move to a different card, the application will display a brief message on the screen, like that in Figure 3.



MOVING TO CARD
2 OF 11

Fig. 3: Between card

Self-test (Yes & No)

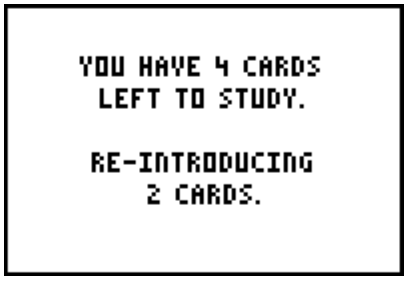
As you go through the stack of cards, you can keep track of when you think you know a card well, and when you think you need more practice on that card.

- Press **YES (WINDOW)** when you are confident that you know the answer for a card
- Press **NO (ZOOM)** when you are not confident that you know the answer for a card

The application will remember the cards you answered NO to, and will show those again the next time you go through the stack.

The application will also remember the cards you answered YES to, and will remove them from the stack. However, the application will occasionally put the YES cards back into the stack so that you don't forget those answers while you are busy looking at the NO cards.

When you reach the end of the stack, the application will let you know how many NO cards you have left, and how many YES cards are going back into the stack. The screen will look like that in Figure 4.



```
YOU HAVE 4 CARDS
 LEFT TO STUDY.

 RE-INTRODUCING
    2 CARDS.
```

Fig. 4: End-of-stack

End of Stack

Every time you finish a stack, you will get a menu of options:

SHUFFLE/REPEAT
    Reorders and restarts the current stack. *Note: this option will not appear if you don't have any NO cards left (i.e. you answered YES to all the cards in the current stack)*.

PICK NEW STACK
    Lets you select a new flash card stack to study

STATISTICS
    Displays a list of which cards you said YES or NO to

INSTRUCTIONS
    Brings up a short summary of key commands

QUIT
    Exits the flash card viewer

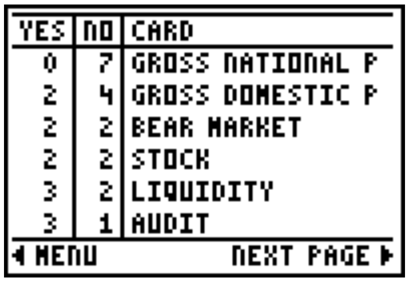
You can also access this menu from any card by pressing the **MENU (TRACE)** button.

Statistics

The application keeps track of which cards you said YES and NO to.

- Select **STATISTICS** from the menu to look at a report of your YES and NO cards.

You will get a list similar to Figure 5. At the top of the list (on the first page of statistics), you will see the cards you answered NO to more often. At the bottom of the list (on the last page of statistics), you will see the cards you answered YES to more often.



YES	NO	CARD
0	7	GROSS NATIONAL P
2	4	GROSS DOMESTIC P
2	2	BEAR MARKET
2	2	STOCK
3	2	LIQUIDITY
3	1	AUDIT

Fig. 5:

Exiting the Flash Card Viewer

There are two ways to exit the application:

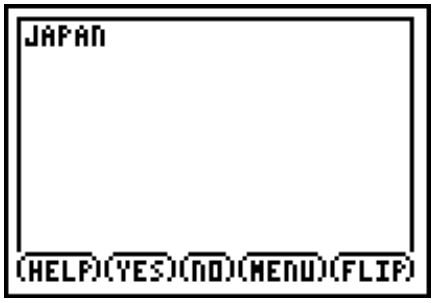
- Select **QUIT** from any menu

  or
- Press the **2nd** key and then the **QUIT** (**MODE**) key.

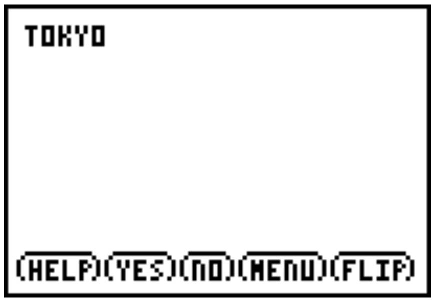
Flash Card Viewer Details

1. Basic Characteristics

FRONT



BACK

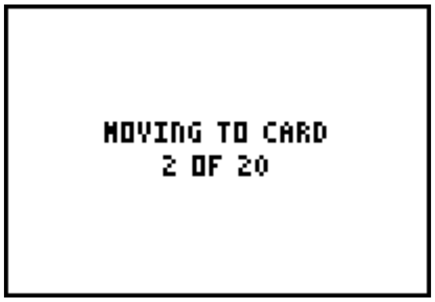


Feature:
Each flash card consists of a front and back. The front of each flash card appears first, and user must press the blue "Graph" button which maps to the Flip label on the screen to see the back of the flash card.

Reasoning:
The basic idea for this feature came directly from paper flash cards. People who are familiar with paper flash cards expect for each card to have a front and a back.

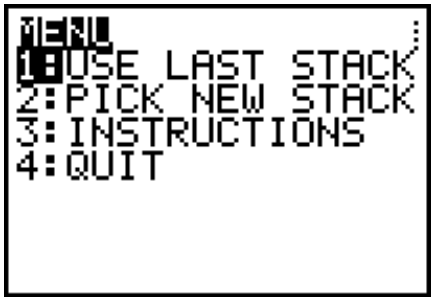
Pitfall:
Some users mistook the front of the next flash card with the back of the current flash card when words in the flash card set were completely unfamiliar. This situation only occurred when the user was not aware that it was possible to flip the flash card, before the Flip label was instated.

Solutions:
We invented two solutions to solve this problem. The fifth empty button was labeled 'Flip' and given the same flip functionality as the up and down arrow keys on the calculator. The insertion of the following screen between cards in user tests with participants also aided in marking the transition from one card to the next.



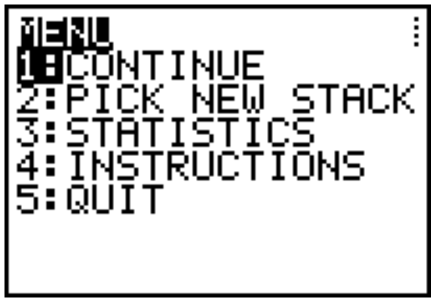
MOVING TO CARD
2 OF 20

2. Navigation, Menus, Multiple Stacks



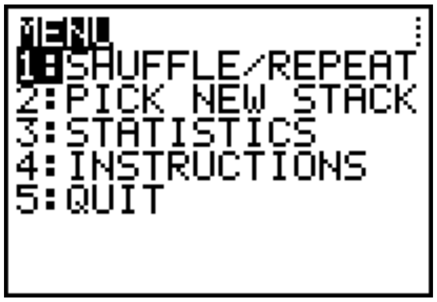
```
MENU
1:USE LAST STACK
2:PICK NEW STACK
3:INSTRUCTIONS
4:QUIT
```

1. Start Menu



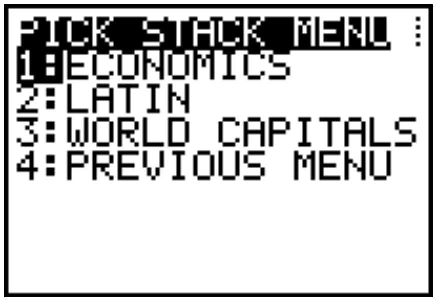
```
MENU
1:CONTINUE
2:PICK NEW STACK
3:STATISTICS
4:INSTRUCTIONS
5:QUIT
```

2. Intermediate Menu, accessed from 'Menu' button



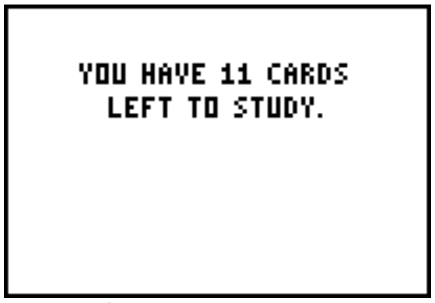
```
MENU
1:SHUFFLE/REPEAT
2:PICK NEW STACK
3:STATISTICS
4:INSTRUCTIONS
5:QUIT
```

3. Final Menu



```
PICK STACK MENU
1:ECONOMICS
2:LATIN
3:WORLD CAPITALS
4:PREVIOUS MENU
```

4. Pick Stack menu



```
YOU HAVE 11 CARDS
LEFT TO STUDY.
```

5. End of stack message

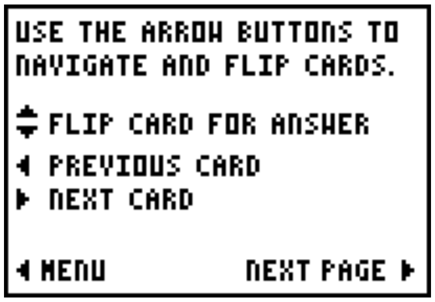
Feature:
The flash card application has four distinct menus available at different times during the use of the application. The 'Start Menu' (1) is the first screen to appear when the user starts the application. The 'Intermediate Menu' (2) can be reached by pressing the blue "TRACE" button

Reasoning:
As more functionality was added to the application, it became necessary to invent more efficient methods for accessing that functionality from various parts of the application.

Pitfall:
From glancing over the menu screens above, it is clear that there is a possibility for users to become confused by the different menus since they all contain similar choices.

Solution:
We attempted to minimize this by keeping the position of an option consistent across all menus. For example, you can see that the 'Pick New Stack' option is always number 2, and 'Quit' is always the last menu option.

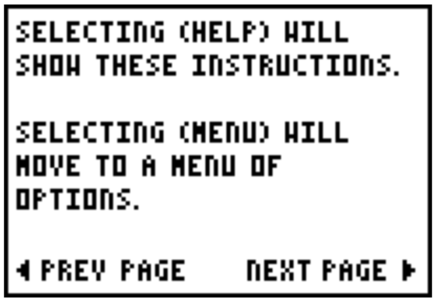
3. Help (Instructions)


USE THE ARROW BUTTONS TO NAVIGATE AND FLIP CARDS.

⇕ FLIP CARD FOR ANSWER
◀ PREVIOUS CARD
▶ NEXT CARD

◀ MENU NEXT PAGE ▶

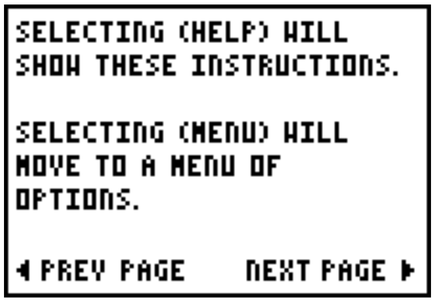
Help Page #1


SELECTING (HELP) WILL SHOW THESE INSTRUCTIONS.

SELECTING (MENU) WILL MOVE TO A MENU OF OPTIONS.

◀ PREV PAGE NEXT PAGE ▶

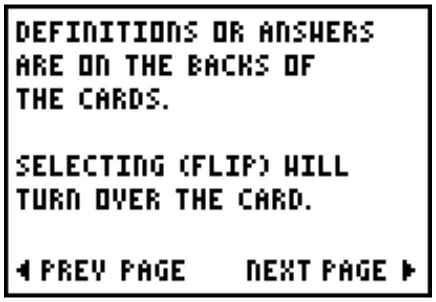
Help Page #2


SELECTING (HELP) WILL SHOW THESE INSTRUCTIONS.

SELECTING (MENU) WILL MOVE TO A MENU OF OPTIONS.

◀ PREV PAGE NEXT PAGE ▶

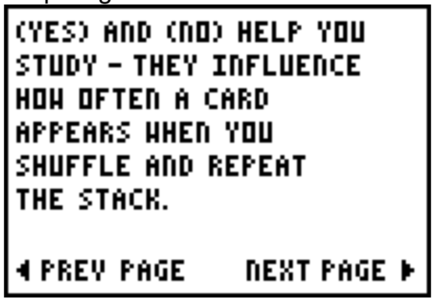
Help Page #3


DEFINITIONS OR ANSWERS ARE ON THE BACKS OF THE CARDS.

SELECTING (FLIP) WILL TURN OVER THE CARD.

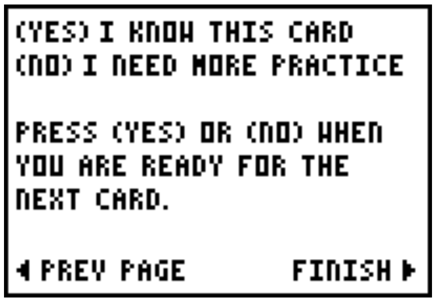
◀ PREV PAGE NEXT PAGE ▶

Help Page #4


(YES) AND (NO) HELP YOU STUDY - THEY INFLUENCE HOW OFTEN A CARD APPEARS WHEN YOU SHUFFLE AND REPEAT THE STACK.

◀ PREV PAGE NEXT PAGE ▶

Help Page #5


(YES) I KNOW THIS CARD
(NO) I NEED MORE PRACTICE

PRESS (YES) OR (NO) WHEN YOU ARE READY FOR THE NEXT CARD.

◀ PREV PAGE FINISH ▶

Help Page #6

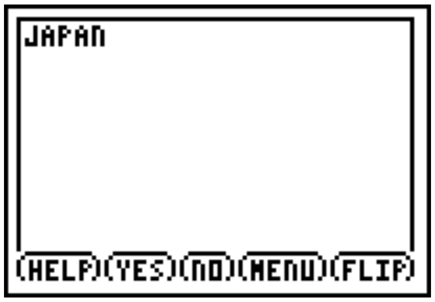
Feature:
The Flash Card Viewer has a sequence of six screens of instructions which users should read before beginning to use the application.

Reasoning:
While our user tests demonstrated that the calculator application is useable without reading the instructions, it is difficult for the user to understand the purpose of the Yes / No buttons without reading the online help. The instructions are available

Pitfall:
Students who don't read the help will miss important functionality of the Flash Card Viewer.

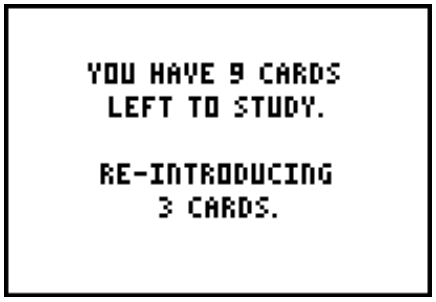
Solution:
We attempted to lessen the impact of not reading the instructions on users by making the help available from every screen and menu in the application.

4. Yes & No / Stats


JAPAN

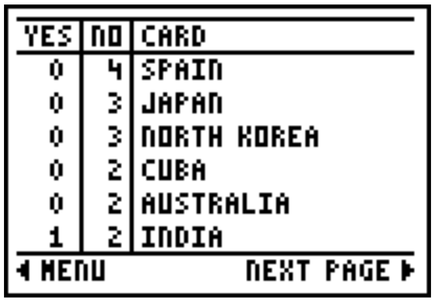
(HELP) (YES) (NO) (MENU) (FLIP)

Front of card


YOU HAVE 9 CARDS
LEFT TO STUDY.

RE-INTRODUCING
3 CARDS.

Re-introduction


YES	NO	CARD
0	4	SPAIN
0	3	JAPAN
0	3	NORTH KOREA
0	2	CUBA
0	2	AUSTRALIA
1	2	INDIA

◀ MENU NEXT PAGE ▶

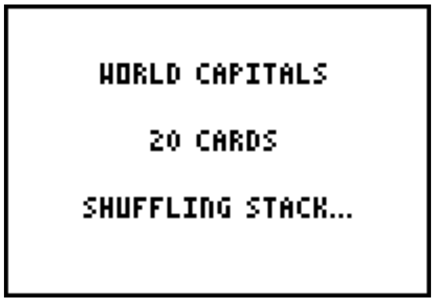
Statistics

Feature:
The main aspect of the Flash Card Viewer that gives it an advantage over paper flash cards is the inclusion of the Yes and No buttons in the interface. These buttons mimic the behavior of users of paper flash cards, in that they provide users with the chance to progressively narrow down their set of flash cards to the ones they are less comfortable with. The calculator records their answers for each card and re-introduces random cards that users have said they feel comfortable with. The Statistics screen lists each card in a flash card set in order, starting with the items users said they were less comfortable with.

Reasoning:
See the section in Part I of this report regarding research into the Psychology Literature for more information.

Pitfall:
Students who don't read the help will not understand the purpose of the Yes and No buttons.

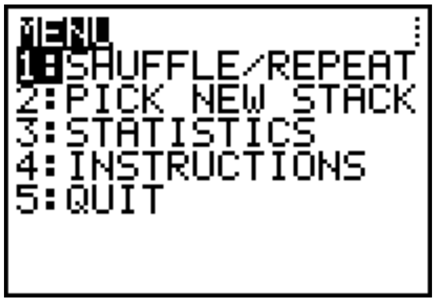
5. Reintroduction of Cards / Shuffle & Repeat


WORLD CAPITALS

20 CARDS

SHUFFLING STACK...

Shuffling…



Shuffle and Repeat

Feature:
The Flash Card Viewer presents a set of flash cards in random order each time the user re-starts the set of cards.

Reasoning:
Presenting the flash cards in a different order each time they are viewed is an effort to avoid learning order effects. See the section in Part I of this report regarding research into the Psychology Literature for more information.

Pitfall:
This feature makes it difficult to look up a specific word in the set of flash cards on the calculator.

Solution:
See the future work section of this document.

Flash Card Maker Manual

Getting Started

1. Double-click on **FlashMkr5.0** to open the Flash Card Maker.

2. Go to **File → New Flashcard Set** to create a new stack.

The stack starts with a blank card. You can immediately start typing text on the front of the flash card.

3. To start typing on the back of the card click in the "Back of Card" window, or press the tab key.

4. To make a new card click on the **NEW** button, or press tab to highlight the **NEW** button and then press Enter.

Making New Cards

You can insert a new card into the stack in one of three ways:

- Click the **NEW** button on the control panel
  or
- Go to **Cards → Insert New Card**
  or
- Press **Ctrl-N**

The application will insert the new card in the next stack position, following the one that is being displayed. If you are looking at the last card in the stack, the new card will be added to end of the stack.

Deleting Cards

Like inserting, you can remove a card from the stack in one of three ways:

- Click the **DEL** button on the control panel
  or
- Go to **Cards → Delete This Card**
  or
- Press **Ctrl-D**

A dialog box will come up asking you to verify that you want to delete the card. Click **Yes** if you do want to delete the card, **No** if you want to cancel the operation. The displayed card will be deleted and the rest of the stack will shift up to fill the hole.

Moving Between Cards

Like the viewer, the Flash Card Maker uses arrow keys to move between cards.

- To see the next card in the stack, use the **right arrow** (⇨) on the control panel, or go to **Cards → Next Card**.

- To return to the previous card in the stack, use the **left arrow** (⇦) on the control panel, or go to **Cards → Previous Card**.

- To jump to the first card in the stack, go to **Cards → First Card**.

- To jump to the last card in the stack, go to **Cards → Last Card**.

Searching for Cards

Search functions have been included in the Flash Card Maker to make it easier to go through a big stack.

- To search for a card containing a particular word,
  Go to **Edit → Find** (or press **Ctrl-F**) and enter the word you are looking for.

The Flash Card Maker will jump to the first card containing the word.

- To search for a card in a list format,
  Go to **Cards → Go to Card #** (or press **Ctrl-G**).

The application will bring up a box containing a scrollable list of cards in the stack.   Double-click on the card you want to see.

Saving and Loading Flash Card Stacks

- To save the card stack under the same file name,
  Go to **File → Save** (or press **Ctrl-S**).

- To save the card stack under a different file name,
  Go to **File → Save As**

Make sure to save the stack as type “Flash Card File” (*.FCF).

- To load an existing flash card stack,
  Go to **File → Open** (or press **Ctrl-O**) and choose from the list of FCF files.

Exporting Cards to Calculator

Once you have created your flash card stack, you can send them to the TI-83 Plus so that they can be used with the Flash Card Viewer. To do this you will need to have the TI- Graph Link ™ for Windows® software, which you can download for free from the Texas Instruments website (http://www.ti.com/calc). You must use the version of Graph Link for the TI-83 Plus.

1. In Flash Card Maker, go to **Link → Send Cards to Calculator**.

2. Enter the file name of the stack and save as type "Text File" (*.TXT).

3. Open Graph Link and go to **Tools → Import → ASCII Program**.

4. Select your stack file (ending in .TXT).

5. Save your stack file as type "TI-83 Plus Program" (*.8xp).

6. *** In the Program window, change the program name to "INITDAT1", "INITDAT2", or "INITDAT3".

7. Click on **Send To RAM** in the Program window to send the file to the TI-83 Plus calculator.

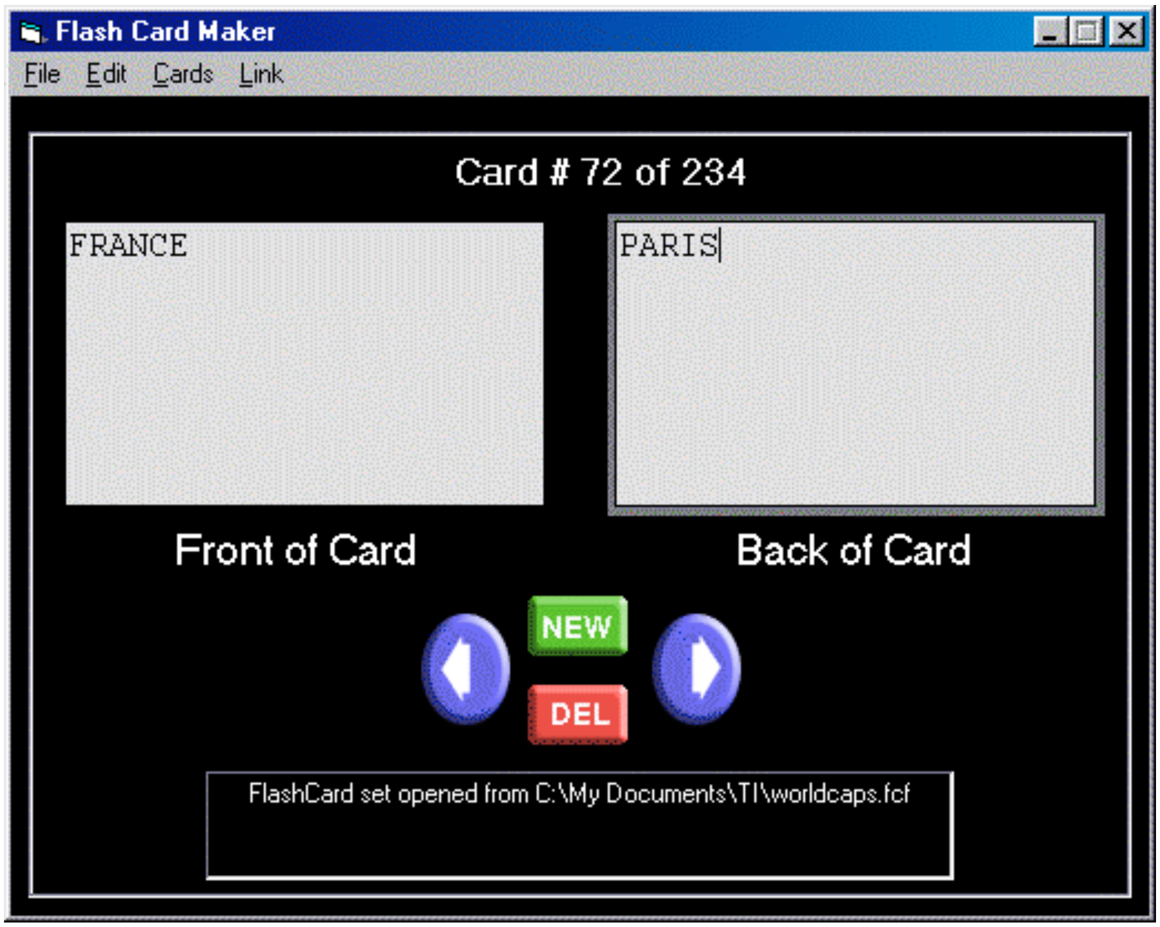
Exiting the Flash Card Maker

- To exit the Maker, go to **File → Quit** or press **Ctrl-Q**.

*** This step will change once dynamic stack naming has been implemented in the application. For example, the user will be able to name it "History"," Biology", or whatever.

Flash Card Maker Details

Two Screens

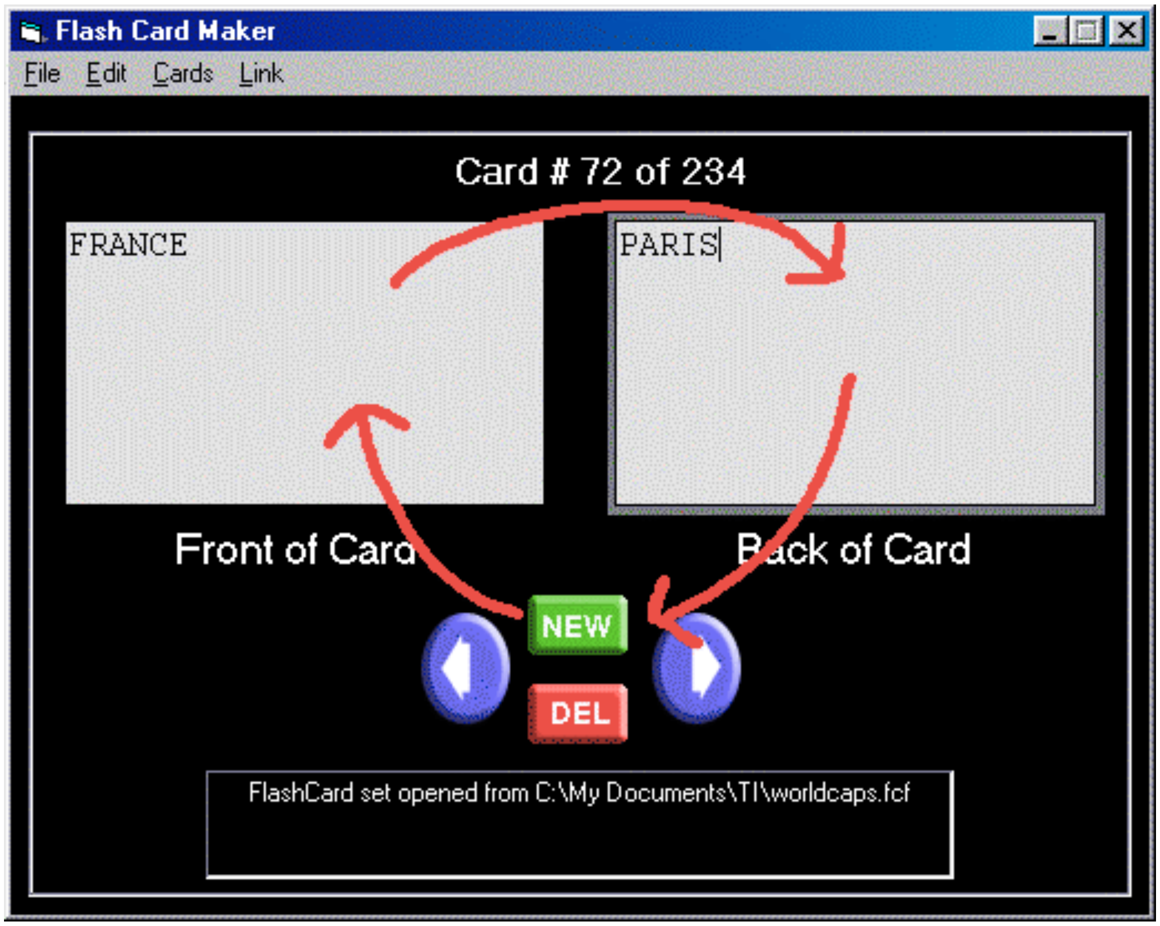


FlashCard set opened from C:\My Documents\TI\worldcaps.fcf

Feature: Make the 'Front of Card' and 'Back of Card' text fields visible simultaneously.

Reasoning: Throughout user testing the calculator application, we found a significant number of students having difficulty distinguishing between the back of one card, and the front of the next. Showing both sides of a single card in the Flash Card Maker eliminate this potential navigational error. “Its easy to move between cards and recognize [the] front and back.” (user 15)

Tab Switching

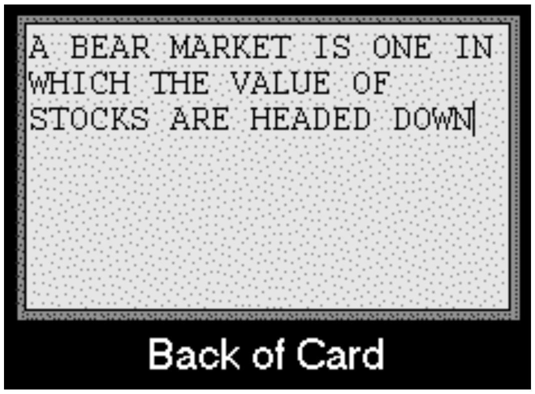


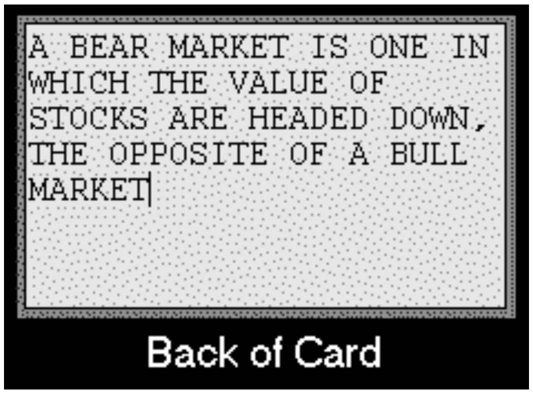
FlashCard set opened from C:\My Documents\TI\worldcaps.fcf

Feature: Tabbing switches focus between ‘Front of Card’, ‘Back of Card’, and the ‘New Card’ button.

Reasoning: With their design heuristic ‘Provide Short Cuts’, Nielsen and Molich (1989) suggest targeting frequently performed tasks involving lengthy sequences of actions. The creation of multiple sequential flashcards in the Flash Card Maker is greatly facilitated if the user need not remove his hands from the keyboard. By tabbing between ‘Front of Card’, ‘Back of Card’, and the ‘New Card’ button, expert users are able to rapidly generate flashcard content. Users also asked for this feature, or alluded to it with comments such as, “I wanted to tab to get to the back of card window.” (user 7)

Cursor Positioning

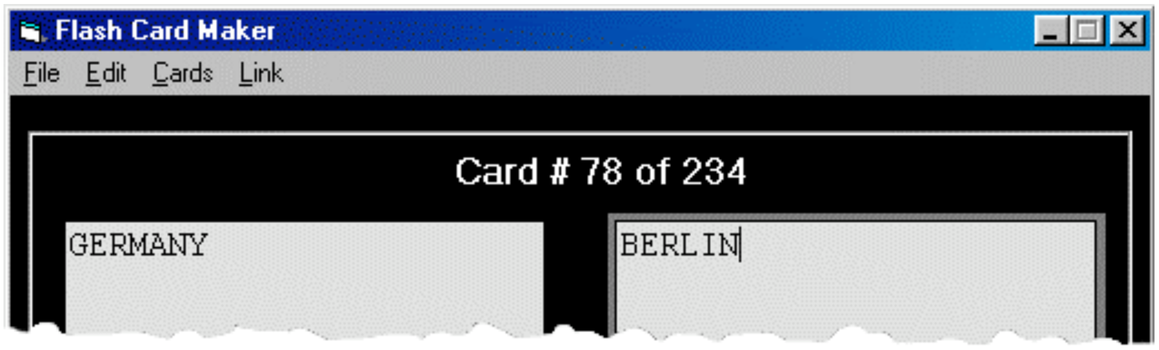

A BEAR MARKET IS ONE IN WHICH THE VALUE OF STOCKS ARE HEADED DOWN


A BEAR MARKET IS ONE IN WHICH THE VALUE OF STOCKS ARE HEADED DOWN, THE OPPOSITE OF A BULL MARKET

Feature: When tabbing between the Front and the Back of a card or navigating from one card to another, the cursor is inserted at the end of the active text box.

Reasoning: A common editing task involves adding additional information to the end of a card. By inserting the cursor at the end of the active text box, we are supporting this task.

Displaying Card Numbers



Card # 78 of 234

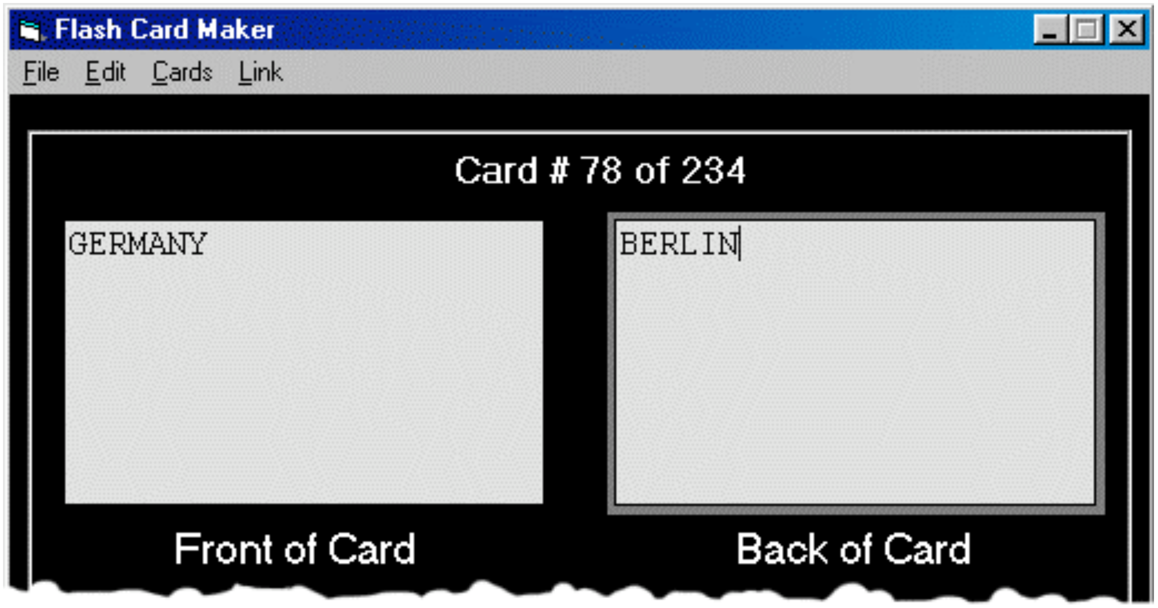
GERMANY

BERLIN

Feature: Display the current card number in the context of the total number of cards.

Reasoning: Throughout user testing the calculator application, we found a significant number of students having difficulty distinguishing between the back of one card, and the front of the next. Without a clear picture of where in the stack he is, a user cannot accurately navigate the flashcards. By using the language "Card # X of Y", we are reinforcing the language used in the calculator application.

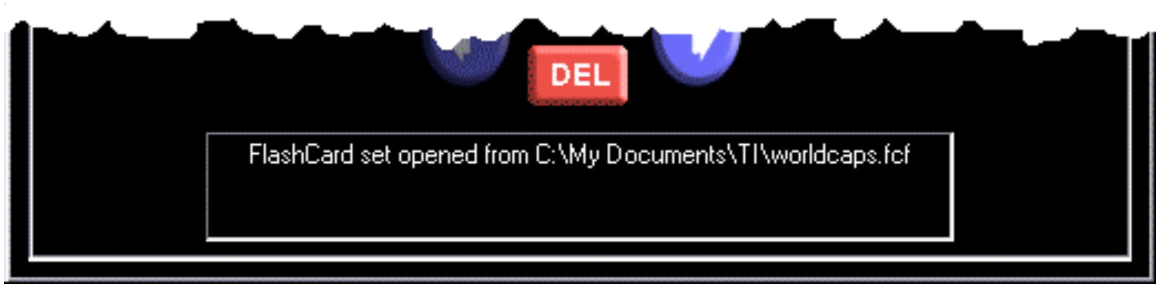
Highlighting of Active Text Field



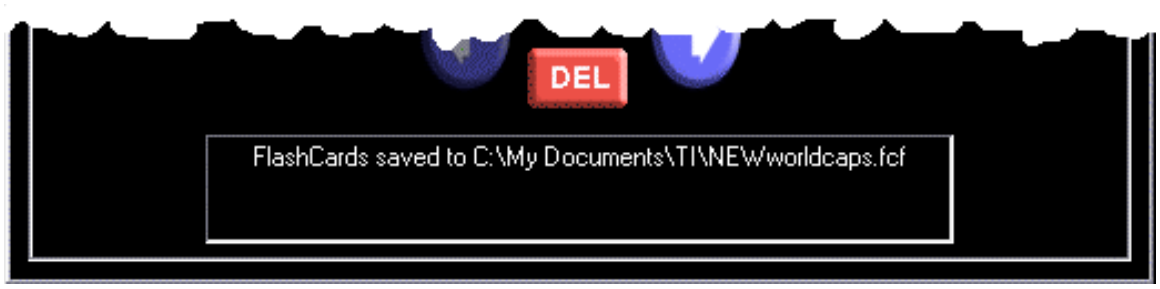
Feature: Give the text field or button with focus extra highlighting beyond a blinking cursor alone. In the above screenshot, we see that the text field on the right has both the focus (notice the cursor) and an extra border.

Reasoning: With multiple text fields and buttons, users often had trouble understanding where the application's focus was. “I couldn't see the cursor in the ‘Front of Card’ to know that I could type in a card right away – I thought I had to click in the box to make it work.” (user 9) This misunderstanding lead to several editing mistakes. By giving the active text field or button an extra level of highlighting, we were able to help users avoid these errors.

Message Box



FlashCard set opened from C:\My Documents\TI\worldcaps.fcf



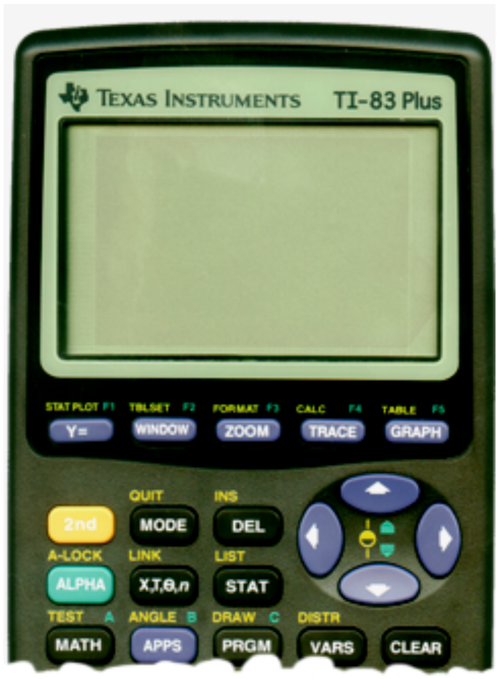
FlashCards saved to C:\My Documents\TI\NEWworldcaps.fcf

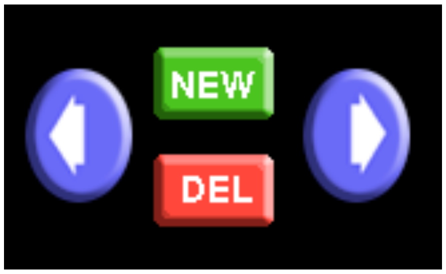
Feature: Provide a message box for communicating system states and errors to the user.

Reasoning: Through user testing, we observed several subjects questioning whether or not a save or deletion actually occurred. A resident message box serves as a common area for such system messages, as well as any instructions or errors we need to convey to the user. A message box provides more detailed feedback than a system beep.

Future Work: The message box could also be used to explain such topics as illegal characters or the maximum length of text fields.

Common Navigation Affordances

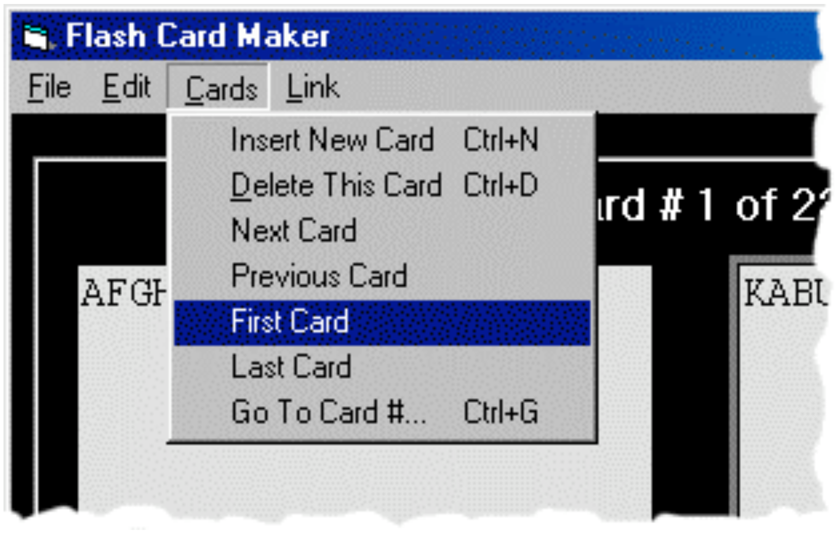




Feature: Navigation buttons mimic the calculator's arrow keys.

Reasoning: Because the calculator app already teaches one method of flashcard navigation, the Flash Card Maker should leverage on this technique. By teaching one common navigation technique, we eliminate unnecessary learning of multiple interfaces.

Rapid Navigation

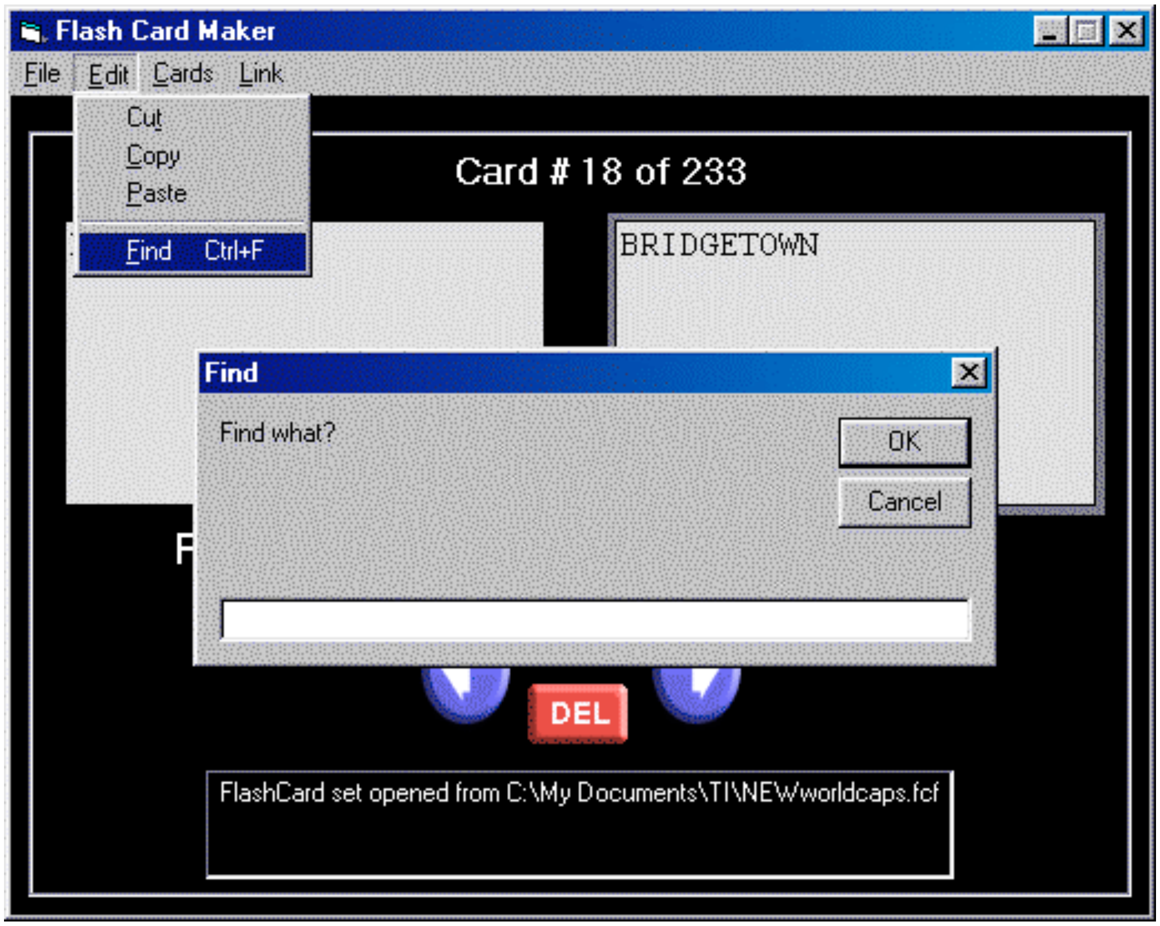


Feature: Provide a means of navigating to the first or last card in one step.

Reasoning: A common editing task is adding additional cards to the end of an existing stack. Through user testing, we noticed a high level of frustration clicking through a stack of cards one by one. Providing a method to jump to the last card in a stack eases this task. Another common task we observed included navigating back to the beginning of a stack, and reviewing the stack's contents; therefore, we added a similar means to jump to the first card. "After making corrections, he used << key to go to beginning of stack so that he could double-check his work." (Notes on subject 18)

Pitfall: Thinking we could make navigating to the first and last card as accessible as navigation to the next or previous card, we testing a version of the Flash Card Maker that included a 'First Card' and 'Last Card' button. Through user testing, we found that the addition of these buttons caused much confusion – users often could not correctly identify their functionality and often clicked the wrong button. We removed the buttons from the screen, and left the functionality to the 'Cards' menu.

Finding



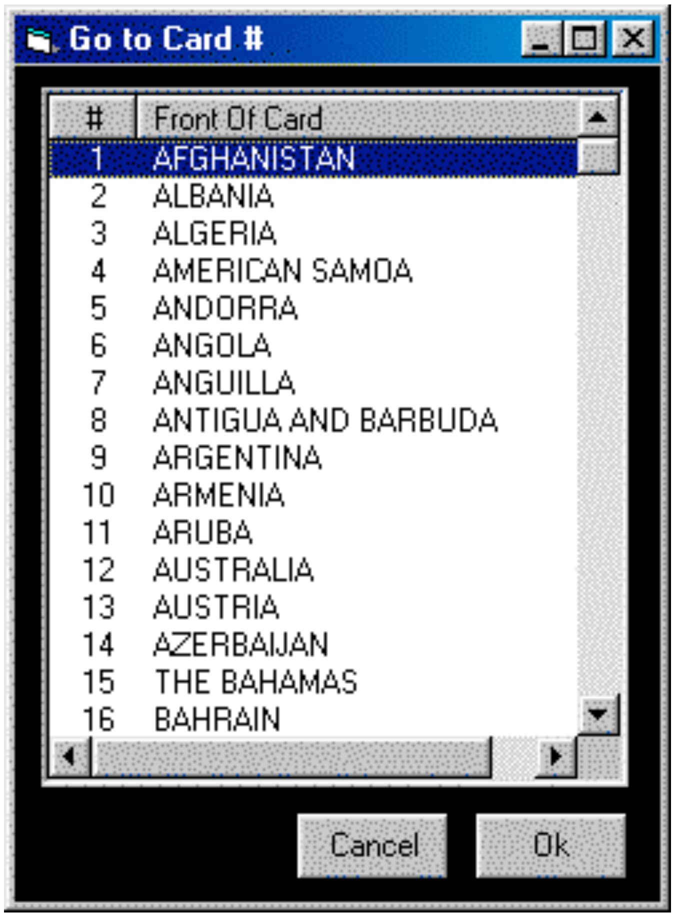
FlashCard set opened from C:\My Documents\TI\NEWworldcaps.fcf

Feature: Search for a specific string within stacks

Reasoning: Asked for specifically by users, the find dialogue box makes several editing tasks easier.

Future Work: Add the ability to replace found text should improve these tasks further.

'Go to Card #…'

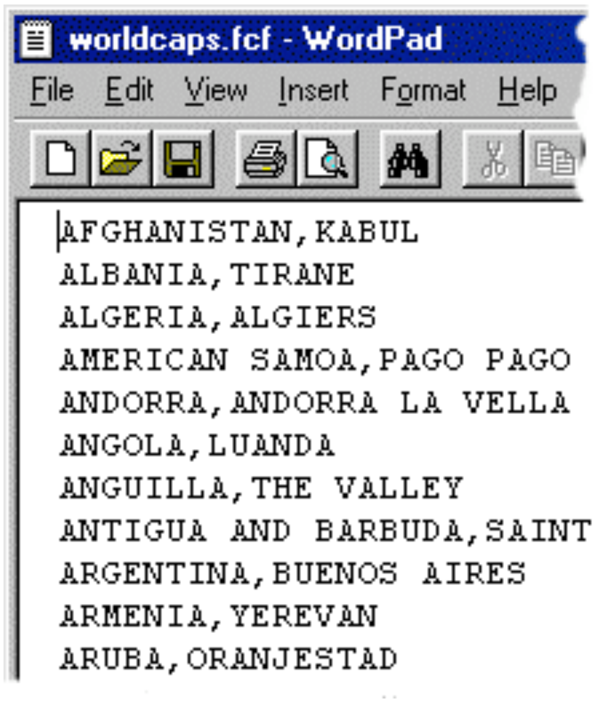


#	Front Of Card
1	AFGHANISTAN
2	ALBANIA
3	ALGERIA
4	AMERICAN SAMOA
5	ANDORRA
6	ANGOLA
7	ANGUILLA
8	ANTIGUA AND BARBUDA
9	ARGENTINA
10	ARMENIA
11	ARUBA
12	AUSTRALIA
13	AUSTRIA
14	AZERBAIJAN
15	THE BAHAMAS
16	BAHRAIN

Feature: Provide a listview of cards with their numbers and text.

Reasoning: Navigating through large stacks frustrated many subjects. Furthermore, searching tasks were hampered by the lack of a holistic view of the stack. A listview to jump to any arbitrary card makes searching and editing tasks significantly easier.

File Format

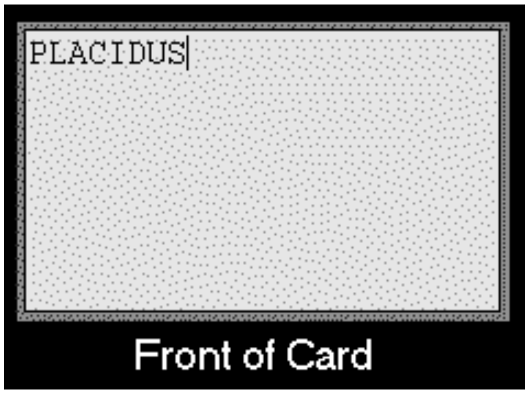


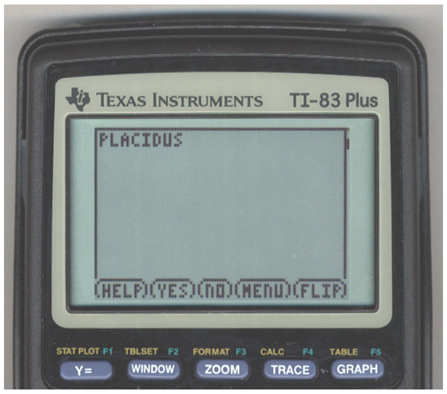
```
AFGHANISTAN,KABUL
ALBANIA,TIRANE
ALGERIA,ALGIERS
AMERICAN SAMOA,PAGO PAGO
ANDORRA,ANDORRA LA VELLA
ANGOLA,LUANDA
ANGUILLA,THE VALLEY
ANTIGUA AND BARBUDA,SAINT
ARGENTINA,BUENOS AIRES
ARMENIA,YEREVAN
ARUBA,ORANJESTAD
```

Feature: Provide a text editable file format for easy importing of pre-existing materials.

Reasoning: Through using a text-editable file format, we are allowing experts to rapidly generate flashcard data from pre-existing sources. Though teacher interviews, we found that teachers often encourage their students to conduct research on the WWW. Being able to transmit this data to the Flash Card Maker seems a necessity. See the report distributed in April 1999 for further details.

Pseudo WYSIWYG Screens

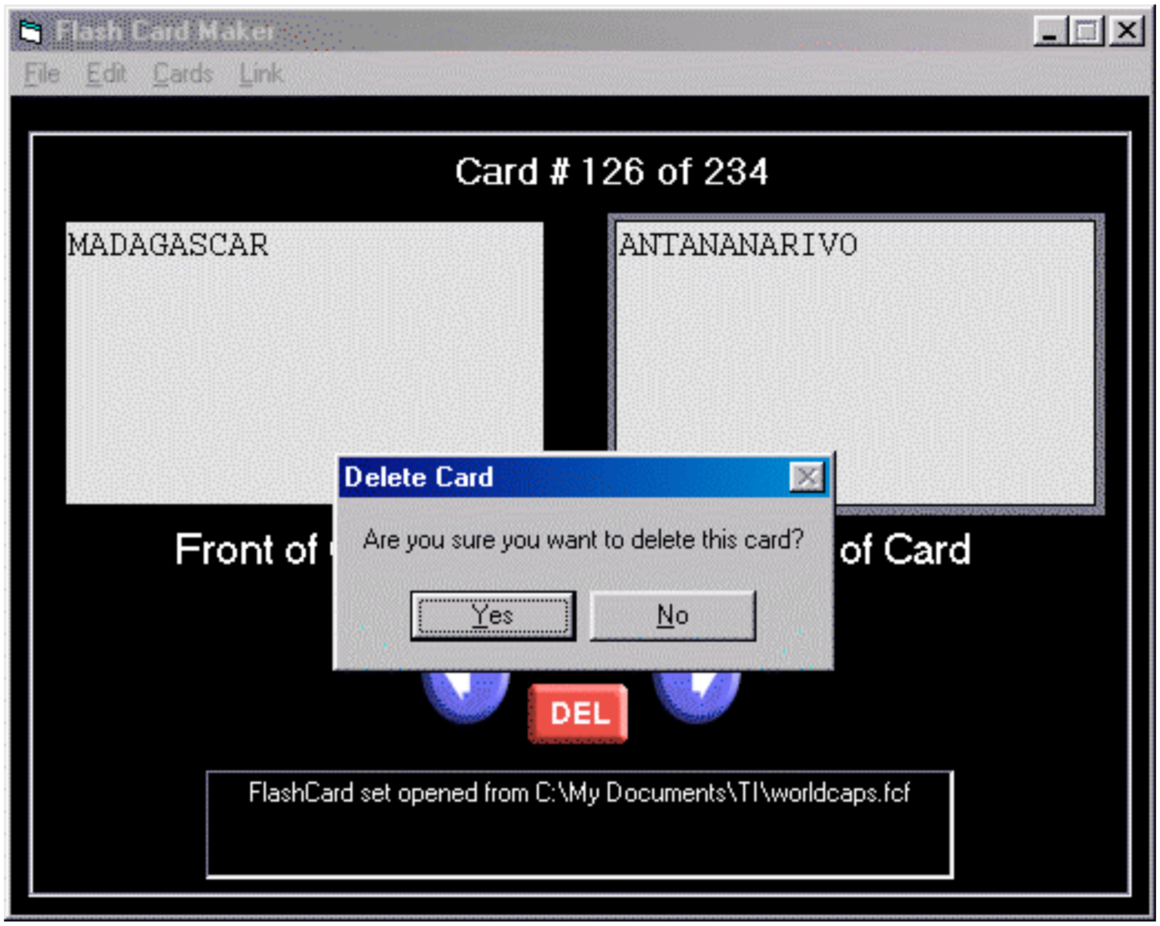



TEXAS INSTRUMENTS TI-83 Plus
PLACIDUS
HELP YES NO MENU FLIP
STAT PLOT F1 TBLSET F2 FORMAT F3 CALC F4 TABLE F5
Y= WINDOW ZOOM TRACE GRAPH

Feature: 'Front of Card' and 'Back of Card' text fields mimic the appearance of the calculator's screen.

Reasoning: Through making the Flash Card Maker's text fields mimic the look of the calculator, the user quickly sees how his cards will display on the TI-83+. Issues of spacing, how many words will fit on a line, and how many lines will fit on the screen are taken care of with minimal effort. User testing and interviews helped us alter the Flash Card Maker's text from the calculator's text in several ways – screen resolution, contrast, and font. The higher resolution of the computer's screen should be used to make reading easier – especially during long editing tasks. Similarly, increasing the contrast beyond the calculator's improves readability. Along the same lines, serif fonts are known to improve readability.

Confirming Deletion



Feature: Ask user to confirm deletion of flashcard.

Reasoning: On several occasions during user testing, subjects were unclear about whether or not they had successfully deleted a card. “He had no problem deleting a card -- went straight to DEL. After clicking OK in the confirm box he said, "Oops -- I hope that deleted." It didn't seem like there was enough feedback for him to know for sure that the card had been deleted.” (Notes on subject 16) On two occasions, a user accidentally clicked the delete button when targeting the new button. Adding a confirmation dialogue box for deleting eliminate both of these errors during further testing.

Future Work

Flash Card Viewer

Categorization

The statistics screen could be made to look for patterns of mistakes, so that it could point out categories to review. For example, a student may be answering all of the questions on Latin history correctly while they miss many of the Latin vocabulary words. The application could point out the difference in these categories so that the student could adjust their study habits.

Grouping Subsets

On occasion, students mentioned that it would be helpful if certain cards always appeared one after another in the stack. We observed many instances of back and forth reviewing between two similar and confusing cards, such as GNP and GDP. Placing the cards next to one another may aid in learning their distinction.

Pictures/Diagrams on Flash Cards

Many of the flashcards students showed to us included hand drawn pictures and diagrams. Supporting this type of learning seems a natural next step.

Search/Lookup

Subjects expressed an interest in jumping to specific cards from the statistics screen so that they could immediately review missed material.

References to Textbook Material

Several students suggested that they would find it helpful if the statistics screen not only told them what they had missed, but also where to look up the material for review. Providing page numbers along with the statistics would not only let the students know what was wrong, but also let them know where to review it.

Flash Card Maker

Check for Illegal Characters

The PC is able to display many characters that cannot be displayed on the calculator. The Flash Card Maker will need to replace these characters and provide feedback to the user.

Provide Feedback on Maximum Text Length

The Flash Card Maker currently limits the length of both the 'Front of Card' and 'Back of Card' text fields so that the user cannot type in more characters than can be displayed on the calculator. The application will need to provide the user feedback on why this limiting is taking place.

Provide one click linking to calculator from PC

Currently, users must export from the Flash Card Maker, import that file into the TI Link Application, and then transmit the converted file to the calculator. By directly linking the Flash Card Maker to the TI-83+, we would eliminate many potential stumbling points along this path.

Internet Parser

While editing the Flash Card Maker's tab delimitated file format makes the generation of large stacks easier, editing tab delimitated files is still too difficult for many users (see TI Red's report for more detail). Providing an easier means of converting existing data on the WWW into flashcard format would open up a world of content to users of the application.

Memory Warnings

On several occasions during development, we overloaded the storage space of the calculator. We concluded that the Flash Card Maker is able to calculate the size of the data file and provide a warning if it approaches a dangerous size.

Centering Text

This feature was asked for specifically by several users, and would help make the Flash Card Maker's flashcards more like traditional flashcards. A note of caution, several of the teachers we interviewed believed that given too many editing features to students might distract from the generation of new cards.

Listview & Treeview

Through brainstorming and observations, we generated a collection of features dealing with stack manipulation. Providing more holistic views of the flashcards may allow for such features as – making new flashcard sets from old ones, reordering cards, and viewing multiple stacks at once.

Sorting Options

Several users ask for the ability to sort their flashcards in alphabetical order once they were finished generating them. As this would make searching and editing later easier, it seems to be a usefull feature to add.

Spell Checking

Through teacher interviews and observation, we learned that teachers spend a good amount of time preparing materials for their class and in proofreading this material as best they can.

Upper & Lowercase Letters

Readability on the calculator would be greatly improved if the Flash Card Viewer could display both upper and lower case. If this change is made, the Flash Card Maker will need to reflect this new feature.

Appendix A: References

Carroll, M. Nelson, T.O. & Kirwan, A. (1997). Tradeoff of semantic relatedness and degree of overlearning: Differential effects on metamemory and on long-term retention. Acta Psychologica, 95(3), 239-253.

Dunlosky, J. & Nelson, T.O. (1997). Similarity between the cue for judgments of learning (JOL) and the cue for test is not the primary determinant of JOL accuracy. Journal of Memory & Language, 36(1), 34-49.

Keleman, W.L. & Weaver, C.A., III. (1997). Enhanced memory at delays: Why do judgments of learning improve over time? Journal of Experimental Psychology: Learning, Memory, & Cognition, 23(6), 1394-1409.

Lindsay, D. Stephen, K., & Colleen M. (1996). Creating illusions of familiarity in a cued recall remember/know paradigm. Journal of Memory & Language, 35(2), 197-211.

Miner, A. & Reder, L.M. (1994) A new look at feeling of knowing: Its metacognitive role in regulating question answering. In: Metcalfe, J. and Shimamura, A. (Eds). Metacognition: Knowing about knowing. Cambridge, Mass: MIT Press.

Nielsen, Jakob (1999). *Heuristic Evaluation*. Retrieved August 1, 1999 from the World Wide Web: http://www.useit.com/papers/heuristic/

Reder, L.M. & Anderson, J.R. (1982). Effects of spacing and embellishments on memory for the main points of a text. Memory and Cognition, 10, 97-102.

Reder, L.M. & Ritter, F. (1992) What determines initial feeling of knowing? Familiarity with question terms, not with the answer. Journal of Experimental Psychology: Learning, Memory, and Cognition, 18, 435-451.

Russo, R., Parkin, A.J., Taylor, S.R.; & Wilks, J. (1998). Revising current two-process accounts of spacing effects in memory. Journal of Experimental Psychology: Learning, Memory, & Cognition, 24(1), 161-172.

Widner, R.L., Jr., Smith, S.M., & Graziano, W.G. (1996). The effects of demand characteristics on the reporting of tip-of-the-tongue and feeling-of-knowing states. American Journal of Psychology, 109(4), 525-538.

Appendix B: Usability Heuristics

Visibility of system status
The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Match between system and the real world
The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

User control and freedom
Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Consistency and standards
Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention
Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

Recognition rather than recall
Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use
Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Aesthetic and minimalist design
Dialogues should not contain information that is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors
Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation
Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Appendix C: Example User Test Protocol

Instructions

(Collect consent forms. If they don't have their consent form, we will have to reschedule.)

<u>Introduce Study</u>

"The point of today's study is to discover how high school students may interact with an electronic flash card application running on a TI-83 Plus graphing calculator. The application works like normal flash cards in that you have the ability to flip between front- and back- sides of the card, and to move from one card to the next. I am interested in finding out how easy the application is for people to use, without reading a user's manual or doing a tutorial.

<Show the calculator and ask "Have you used one of these before?" If no, say "let me tell you a little bit about the equipment we will be using" and talk a little about the calculator: what it is capable of doing, maybe the On, Prgm, 2nd, and Alpha keys. If yes, ask them how often they used it and what they did the last time they used one. Also briefly explain the recording setup.>

I'm testing the software, I'm not testing you. I'm looking for places where the flash card application might be difficult to use, so if you can't do some things please don't feel bad. That is exactly what we are looking for.

Your participation in this study is completely voluntary. Although I don't know why this would happen, if you become uncomfortable in any way please feel free to stop at any time, and you will still receive your $25 for participating.

We are going to have you do three things during this user testing session:

1. Think aloud while you use the calculator application for the first time so we can discover where the program is confusing. I'll tell you more about thinking aloud in a minute.
2. Study some flash cards and take a short quiz -- you'll get $5 additional dollars if you score above 75% on the quiz.
3. Think aloud while you use an application for creating flash cards on the PC.

<u>Explain Thinking Aloud</u>

In this observation, I am interested in what you think about as you perform the tasks I will be asking you to do. I'm going to ask that you 'think aloud' while you are using the software. What I mean by 'think aloud' is that I want you to tell me EVERYTHING that you are thinking from the first time that you see the statement of the task until you finish the task. I would like you to talk aloud CONSTANTLY from the time I give you the task until you have completed it. I don't want you to try to plan out what to say or try to explain to me what you are thinking. Just act as if you are alone, speaking to yourself -- just a little louder. You'll probably do things like skip words, and reread things that wouldn't make sense to someone listening. This is OK -- it means you are doing it right!

When you are working on the computer, you'll be looking for things, and seeing things that catch your attention. These things you are searching for and things that you see are as important for our observation as thoughts you are thinking from memory, so please verbalize these too.

(Optional: Give them the guidelines for thinking aloud sheet.)

I'm going to do an example for you, so you get an idea of what thinking-aloud sounds like. Listen to the types of things I say as I think-aloud while I play solitaire for a minute or two.

Now it's your turn to think aloud as you are using the computer. Please think aloud as you set the background color to blue.

I'm going to give you a few final instructions:

As you're doing the tasks, I won't be able to answer any questions. But if you do have questions, go ahead and ask them anyway so that I can learn more about what kinds of questions the flash card application brings up. I'll answer your questions after the session. Also, if you forget to think aloud, I'll say, "Please keep talking."

Do you have any questions about thinking aloud?

(Turn on the calculator and start the flash card application.)

Guidelines for Thinking Aloud

- Say whatever's on your mind. Don't hold back hunches, guesses, wild ideas, images or intentions
- Speak as telegraphically as you please. Don't worry about complete sentences and eloquence.
- Don't overexplain or justify what you are doing. Analyze no more than you would normally.
- Don't talk about the past. Try to get into the pattern of saying what you're thinking at the same time you're thinking it, not of thinking for a while and then describing what your thoughts were.
- Speak as continuously as possible. Try to say something at least once every five seconds, even if it's only "I'm drawing a blank."
- Speak audibly. Watch for your voice dropping, as you become involved with what you are doing.

Viewer Task: Economics

Task Description:

You are taking an Economics class. Your teacher has told you that at the beginning of the next class period, she will give you a pop quiz on the vocabulary for chapter. She has provided some electronic Flash Cards for you to study from, and you have already downloaded them into your calculator.

Please study the Economics terms, for several minutes, beginning now.

Questions to ask participants:

- How did you know how to flip cards and jump between cards? ?How could you tell the front from the back?
- What do you think happens when you press Yes? When you press No? How should those buttons be used? Tell me what you think about the Yes/No buttons…
- Did you notice that some of the words you pressed Yes for were reappearing? What is your opinion about this?
- Did you have problems with any part of the application?
- What did you think the statistics screen was telling you? What might make it more useful?

Creator Task:

Task Description:

You have been transported back in time about two months. It is now May, you're back in school, and you are preparing to study for a test that you will have in about a week.

You have a great new program that lets you create flash cards that can be viewed on your TI-83 calculator. You decide to give this program a try, and make some flash cards to use when studying for your test.

Your teacher has given you a file (called WorldCapitals.fcd) that contains flash cards for 232 world capitals. However, she told you that there are a few mistakes that you need to correct before you begin studying the flash cards:

1. Your teacher included the capital of the United States as part of the flash card set. She later decided that it was too easy, and wants this card removed from the set.
2. The capital of Australia was entered incorrectly in this flash card set. Make sure that the capital of Australia reads Canberra, not Sydney.
3. The flash card for Yaounde, the capital of Cameroon, was left out of the set by mistake. Please create a new flash card for Yaounde, Cameroon at position 15 in the flash card set.
4. The city was left off the back of the very last flash card in the set. It should be Harare, the capital of Zimbabwe. Please correct this mistake.
5. Finally, the card for the Philippines is misspelled in two places. There is only one 'L' in Philippines and in Manila.

Questions to ask participants:

- Was it easy to understand how to jump between cards and make new ones?
- Did you have problems with any part of this program?
- What do you think could be improved about this program?

Experimenter: Instructions for the Quiz Task

What I want you to do next is to study a subset of the flash cards for World Capitals, which is already on the calculator. I am going to give you a short quiz when you have finished studying the flash cards. You will receive $5 more if you score a 75% or better on the quiz.

I don't want you to think aloud any more -- in fact, I am going to leave the room while you study. Please take your time studying the flash cards, and when you are ready for the quiz call me back into the room. I can hear what you are saying through the microphone you are wearing, and I can see what you are doing through the camera. I won't be paying attention to anything you are doing, unless you begin speaking to me directly to ask a question or call me back into the room.

(Start the application)

Are you ready to begin?

_____ INDIA	A. ATHENS
_____ GREECE	B. CARACAS
_____ PORTUGAL	C. HAVANA
_____ JAPAN	D. LISBON
_____ VENEZUELA	E. NEW DELHI
_____ NORTH KOREA	F. OTTAWA
_____ CUBA	G. PYONGYANG
_____ CANADA	H. SEOUL
_____ NEW ZEALAND	I. TOKYO
_____ SOUTH KOREA	J. WELLINGTON

_____ AFGHANISTAN	K. ABUJA
_____ ANDORRA	L. ANDORRA LA VELLA
_____ AUSTRALIA	M. CAIRO
_____ CAMEROON	N. CANBERRA
_____ EGYPT	O. JOHANNESBURGH
_____ EQUADOR	P. KABUL
_____ NIGERIA	Q. MADRID
_____ PHILIPINES	R. MANILA
_____ SOUTH AFRICA	S. QUITO
_____ SPAIN	T. YAOUNDE

__E___ INDIA	A. ATHENS
__A___ GREECE	B. CARACAS
__D___ PORTUGAL	C. HAVANA
__I___ JAPAN	D. LISBON
__B___ VENEZUELA	E. NEW DELHI
__G___ NORTH KOREA	F. OTTAWA
__C___ CUBA	G. PYONGYANG
__F___ CANADA	H. SEOUL
__J___ NEW ZEALAND	I. TOKYO
__H___ SOUTH KOREA	J. WELLINGTON

__P___ AFGHANISTAN	K. ABUJA
__L___ ANDORRA	L. ANDORRA LA VELLA
__N___ AUSTRALIA	M. CAIRO
__T___ CAMEROON	N. CANBERRA
__M___ EGYPT	O. JOHANNESBURG
__S___ EQUADOR	P. KABUL
__K___ NIGERIA	Q. MADRID
__R___ PHILIPINES	R. MANILA
__O___ SOUTH AFRICA	S. QUITO
__Q___ SPAIN	T. YAOUNDE

Appendix D: Prototyping

Flash Card Viewer

VERSION	**FUNCTIONALITY**
Version 1	Next card, flip, quit (low fidelity -- internal only, no user testing)
Version 2	Compare different screen clearing mechanisms Flip on up & down arrows, Next and Previous on right & left arrows Button labels are | Help | Yes | Shaky | No | <blank> | Compare borders to differentiate front from back and different word spacing Two screens of help, 2nd-Quit to quit the app 'End of stack' message
Version 3	'Shaky' button eliminated, clear by screen retained Border on front, no border on back Button labels are (help) (menu) (yes) (no) ( ) Pressing Yes for a card hides that card in the next shuffle; Next and No don't Instructions and help button are identical Start menu at beginning of app – Use last stack – Pick new stack – Instructions – Quit Middle menu reached from menu button: – Continue – Pick new stack – Statistics (blank) – Instructions – Quit End menu – Tells how many cards are left – Shuffle and repeat – Pick new stack – Instructions – Quit
Version 4	Shuffle reintroduces cards with a 20% chance, Help / Instructions has 4 screens Button labels are (help) (menu) (yes) (no) (flip)
Version 5	More info in help screens Button labels are (help) (menu) (yes) (no) (flip) Statistics screen lists all cards – Sorted by ratio of yes to no answers – Displayed with number of yes and no answers Reintroduction of a card – Chance depends on ratio of yes and no answers – Cards never receiving no have 20% chance of reintroduction Left button cannot back out of Help screens, CAN back out of Statistics screens "Please wait…" changed to "Shuffling Stack…"
Version 6	Button labels are (help) (yes) (no) (menu) (flip) Faster card sort, User can dismiss message screens by pressing any key Users can back out of pick stack menu Statistics and Help use 'MENU' on first and last page instead of prev/next menu Compare 2 versions: – Between-card message: 'MOVING TO CARD #' – Label on each card: 'FRONT OF CARD #'

Flash Card Maker

VERSION	FUNCTIONALITY
Version 1	Next, Previous, New Card, and Delete Buttons ‘Front of Card’ and ‘Back of Card’ text fields Simple Save and Open Display active card # and total card # Simple system message box
Version 2	Export to calculator data format added Calculator Look and Feel added
Version 3	Tab switching between ‘Front of Card’, ‘Back of Card’, and ‘New’ button Position Cursor at end of text field on navigate and tab switch Highlighting of active text field or button Added keyboard shortcuts for common operations Language now matches calculator’s
Version 4	Jump to first/last card buttons and menu items added Searching by string ‘Go To Card #…’ form added Cut/Copy/Paste works reliably
Version 5	Made export to calculator file format more efficient Added confirm to delete cards Removed buttons for jump to first and last card Fixed silent save error that saved over old files

Appendix E: User Test Data Sources

#	M/F	YEAR	SCHOOL	PUBLIC / PRIVATE
1.	M	Soph	Allderdice	Public
2.	M	Fr	Mestivta	Private
3.	F	Grad	Shadyside Academy	Private
4.	F	Soph	Allderdice	Public
5.	F	Sr	Trez' Academy	Private
6.	M	Soph	Allderdice	Public
7.	M	Soph	Schenley	Public
8.	M	Fr	Allderdice	Public
9.	M	Soph	Allderdice	Public
10.	F	Soph	Peabody	Public
11.	M	Soph	Allderdice	Public
12.	M	Sr	Oakland	Private
13.	F	Jr	Allderdice	Public
14.	M	Grad	Andrews (TX)	Private
15.	F	Grad	Faith Christian	Private
16.	M	Fr	Central Catholic	Private
17.	M	Sr	Central Catholic	Private
18.	M	Soph	Steel Valley	Private
19.	M	Sr	Allderdice	Public
20.	M	Soph	Mestivta	Private
21.	F	Soph	Allderdice	Public
22.	M	Sr	Allderdice	Public
23.	M	Jr	Schenley	Public
24.	F	Jr	Ellis	Private
25.	M	Fr	Central Catholic	Private
26.	M	Sr	Allderdice	Public

Appendix F: TI-Basic code documentation for Flash Card Viewer application

Purpose of this document

This document replaces the traditional comments in program code. There are 2 reasons for not commenting the program code directly: 1) Comments in an interpreted TI-Basic program makes the program run slower and 2) the code viewing space in TI-Graph Link is so limited that it is difficult to communicate effectively.

Data Structure

The contents of each flash card stack are captured in the variables Str1, Str2, and lists POS1, POS2, LIN1, LIN2, SPC1, and SPC2.

Str1 and lists POS1, SPC1, and LIN1 are for the front sides of the cards.
Str2 and lists POS2, SPC2, and LIN2 are for the back sides of the cards.

Str1 contains the text for the front sides of the cards in the stack.
Str2 contains the text for the back sides of the cards in the stack.
list POS1 contains the starting positions of the sub-strings in Str1.
list POS2 contains the starting positions of the sub-strings in Str2.
list LIN1 relates the lines on the cards to the contents of list POS1.
list LIN2 relates the lines on the cards to the contents of list POS2.
list SPC1 contains info on leading spaces for the strings in Str1.
list SPC2 contains info on leading spaces for the strings in Str2.

The data structure for storing the contents of the backs of cards is the same as storing the contents of the fronts of cards, just different variables (for example, Str2 versus Str1). We will explain the data structure using the back sides of the following 2 card stack, since the back sides have more complexity to them:

	LINE #	Content of front of card	Content of back of card
Card 1	line 1		WELLINGTON
	line 2		
	line 3	NEW ZEALAND	ENGLISH, MAORI
	line 4		
	line 5		
	line 6		
	line 7		
Card 2	line 1		BUCHAREST
	line 2		
	line 3	ROMANIA	ROMANIAN, HUNGARIAN,
	line 4		GERMAN
	line 5		
	line 6		

	line 7		

Contents of string Str2 (The numbers are to help the reader see the positions of the characters):

```
“WELLINGTONENGLISH,   MAORIBUCHARESTROMANIAN,   HUNGARIAN,GERMANZ”
          1         2         3         4         5         6
 1234567890123456789012345678901234567890123456789012345678901234567890
```

1. The final Z in Str2 is a “place keeper” that makes finding the contents of the last sub-string in Str2 no different than finding the contents of any other sub-string in Str2.
2. There are 3 spaces in Str2 for each word space. On the calculator, that will translate to 3 pixels of word spacing.

POS2 contains the starting position of each sub-string. Each sub-string is the contents of a single line of text on a card. Blank lines on a card are **not** represented in either Str2 or POS2. Blank lines are represented in LIN2 (see description of LIN2 on this page).

Contents of POS2: {1,11,27,36,58,64}

- WELLINGTON starts at 1
- ENGLISH, MAORI starts at 11
- BUCHAREST starts at 27
- ROMANIAN, HUGARIAN starts at 36
- GERMAN starts at 58
- the final Z is at 64

SPC2 contains the number of spaces that precede the text on the line. Since all lines of text are left justified in this example, SPC2 is all zeros.

Contents of SPC2: {0,0,0,0,0}

- WELLINGTON has 0 leading spaces
- ENGLISH, MAORI has 0 leading spaces
- BUCHAREST has 0 leading spaces
- ROMANIAN, HUGARIAN has 0 leading spaces
- GERMAN has 0 leading spaces

LIN2 relates the sub-strings in Str2 and POS2 to specific lines on the card. Each side of a card holds 7 lines of text, some of which are blank lines. The numbers in LIN2 are in groups of 7. The first 7 numbers relate to the contents of card 1, the second 7 numbers relate to the contents of card 2, etc.

LIN2: {1,0,2,0,0,0,0,3,0,4,5,0,0,0}

The first set of 7 numbers in LIN2 is 1,0,2,0,0,0,0. These numbers correspond directly to lines 1 to 7 of the back side of card 1.

To find the text for line 1 of the (back side) of card 1.

- Take the 1 from element 1 of LIN2.
- Go to element 1 in POS2.
- That number is 1.

- The text for line 1 begins at position 1 in Str2 and ends at position 11. (11 is the list element after 1 in POS2).

To find the text for line 2 of the (back side) of card 1.

- Take the 0 from element 2 of LIN2.
- The 0 means line 2 is a blank line.

To find the text for line 3 of the (back side) of card 1.

- Take the 2 from element 3 of LIN2.
- Go to element 2 in POS2.
- That number is 11.
- The text for line 3 begins at position 11 in Str2 and ends at position 27. (27 is the list element after 11 in POS2).

To find the text for line 4 of the (back side) of card 1.

- Take the 0 from element 4 of LIN2.
- The 0 means line 4 is a blank line.

To find the text for line 5 of the (back side) of card 1.

- Take the 0 from element 5 of LIN2.
- The 0 means line 5 is a blank line.

To find the text for line 6 of the (back side) of card 1.

- Take the 0 from element 6 of LIN2.
- The 0 means line 6 is a blank line.

To find the text for line 7 of the (back side) of card 1.

- Take the 0 from element 7 of LIN2.
- The 0 means line 7 is a blank line.

The second set of 7 numbers in LIN2 is 3,0,4,5,0,0. The numbers correspond directly to lines 1 to 7 of the backside of card 2.

To find the text for line 1 of the (back side) of card 2.

- Take the 3 from element 8 of LIN2.
- Go to element 3 in POS2.
- That number is 27.
- The text for line 1 begins at position 27 in Str2 and ends at position 36. (36 is the list element after 27 in POS2).

To find the text for line 2 of the (back side) of card 2.

- Take the 0 from element 9 of LIN2.
- The 0 means line 2 is a blank line

To find the text for line 3 of the (back side) of card 2.

- Take the 4 from element 10 of LIN2.
- Go to element 4 in POS2.
- That number is 36.

- The text for line 3 begins at position 36 in Str2 and ends at position 58. (58 is the list element after 36 in POS2).

To find the text for line 4 of the (back side) of card 2.
- Take the 5 from element 11 of LIN2.
- Go to element 5 in POS2.
- That number is 58.
- The text for line 4 begins at position 58 in Str2 and ends at position 64. (64 is the list element after 58 in POS2).

To find the text for line 5 of the (back side) of card 2.
- Take the 0 from element 12 of LIN2.
- The 0 means line 5 is a blank line.

To find the text for line 6 of the (back side) of card 2.
- Take the 0 from element 13 of LIN2.
- The 0 means line 6 is a blank line.

To find the text for line 7 of the (back side) of card 2.
- Take the 0 from element 14 of LIN2.
- The 0 means line 7 is a blank line.

This data structure was not designed with space efficiency as the primary concern. The decision was to trade more space for faster code and development flexibility. For instance word spacing is 3 pixels (which reduce eyestrain greatly). The data structure has three spaces for every word space (which will render on the screen as 3 pixels). This data structure also allowed us to explore text-formatting issues in our program development.

Data variables and lists shared between Code modules

All variables in TI-Basic are global. We identified the following variables for communicating values within a Code Module and between different Code modules.

Variable R indicates the number of lines of text on a card. R is use by Code modules DISPFRNT and DISPBACK to read from LIN1 and LIN2 correctly.

Variable S indicates which stack is currently in use or was most recently used.
- When the Viewer application starts, it checks S to determine if there was a previously used stack that can be presented in the Menu as USE LAST STACK. Currently, expected values are 1 to 3 and they map to the code modules INITDAT1, INITDAT2 and INITDAT3.

Variable Z serves as “return value” parameter for Code modules HELP and PICKSTAC

- In Code module HELP, variable Z indicates if help was requested from the Card View or from a menu. This determines the appropriate label to use for the first help screens.
- In Code module PICKSTAC, variable Z indicates if the user did picked a new stack to study or aborted the action and wanted to return to the previous menu.

Variable N indicates the number of cards in the stack currently in use. Code module INITMISC uses N to setup Lists SORT, YES, NO, SHOW, SCOR

Variable T indicates which side of the card is showing.

- T = 1 means to show front of the card.
- T = 0 means to show back of the card.

Variable C indicates the element in list ORDR whose value is the card index of the card being shown.

Variable K stores the key code of the last button pressed.

List SHOW indicates which cards are hidden.

- SHOW(c) = 1 means to show card c in the next pass.
- SHOW(c) = 0 means to hide card c in the next pass.

List ORDR contains the card indexes to show in this pass in the order in which they will be shown.

List YES and List NO keep track of the number of times Yes and No are pressed for each card in the stack.

List SCOR is used in Code module STATS to contain the ratio of corresponding elements' values in List No to List Yes. The values in List SCOR are for sorting the statistics table.

List SORT is used in Code module STATS to contain the sorted card indexes in descending order of the ratio of number of No(s) to Yes(es) for that card.

Program structure and Code modules

Module Name	**Brief Description**
prgmVIEWER	Main program loop. Button presses direct program flow within prgmVIEWER or call Code modules listed below. prgmVIEWER keeps track of the Yes and No counts; clears and displays the appropriate card view (PIC 0 or PIC 1); and displays the message screen that tells the user the number of cards left to study when the user walks off the last card in this pass of the stack.

Whenever the user presses Yes for a card, the corresponding element in List YES is incremented by 1. The card is hidden in the next pass of the stack by setting the corresponding element in List SHOW to 0.

Whenever the user presses No for a card, the corresponding element in List NO is incremented by 1. The card is unhidden in the next pass of the stack by setting the corresponding element in List SHOW to 1.

Pressing the Left or Right arrow buttons does not change Lists YES, NO and SHOW and therefore have no effect on hiding/un-hiding and re-introduction of cards.

The initial menu has two versions, depending on whether or not the last-viewed (last -used) stack is available. Currently this determination is hard-coded (the value of variable S is checked). We expect the assembler version could track this information in a different way.

prgmPICKSTAC TI-Basic menu that lists the stacks available for studying on the calculator. Currently the menu is hard-coded with Economics, Latin and World Capitals stacks. prgmPICKSTAC calls prgmINITDATn where n is currently hard-coded as 1 for Economics; 2 for Latin and 3 for World Capitals. prgmPICKSTAC then display an information screen that says which stack is being loaded (which is also currently hard-coded), the number of cards in the stack and a “Shuffling…” message.

prgmPICKSTAC will next call prgmINITMISC to setup additional Lists.

In the actual TI-Assembly implementation making use of the Archive memory, we expect the program to check what stacks are in the Archive and present the user with a choice menu. Thus there shall be no hard-coding in the TI-Assembly implementation.

prgmINITDATn, where n = 1,2,3 Essentially a series of store statements. Stores the text appearing on the front of cards (concatenated) in Str1; Stores the text appearing on the back of cards (concatenated) in Str2; Setups Lists POS1, POS2, LIN1, LIN2, SPC1 and SPC2; And store the number of cards in this stack in N.

prgmINITMISC Always called after a call to prgmINITDATn. Initializes Lists SHOW, YES, NO, SCOR and SORT using the variable N.

prgmDISPFRNT Display the text for the front of the current card, indicated by the variable C. prgmVIEWER makes sure C is updated before calling

	prgmDISPFRNT. If you understand how prgmDISPFRNT works, you understand the data structure works.
prgmDISPBACK	Display the text for the back of the current card, indicated by the variable C. prgmVIEWER makes sure C is updated before calling prgmDISPBACK. If you understand how prgmDISPBACK works, you understand the data structure works.
prgmCARDNUM	Clears the screen, displays the inter-cards message. Uses a do-nothing loop to keep the message on the screen long enough to be read by the user before clearing the screen. After the screen is cleared, puts up a PIC (with the skeleton of the card view, i.e. the button labels only).
prgmPREVMSG	Called when user presses the Left Button (move to previous card) at the first card of the current pass of the stack and therefore there is no previous card to move to. prgmPREVMSG will display an information screen ("you are at the first card of the stack " and the number of cards in this pass of the stack). Uses a do-nothing loop to keep the message on the screen long enough to be read by the user before clearing the screen. The user can clear the screen before the do-nothing loop finishes by pressing any button.
prgmMIDMENU	TI-Basic menu that appears when user presses the Menu button from card view. Allows the user to • continue (return to card view) • pick a new stack to study • view Statistics • read the help screens • quit Viewer
prgmHELP	Displays a series of help screens navigated by the Left and Right Buttons. Current implementation describe what the 4 arrow buttons do, the mapping of the top row of calculator buttons and the replacement onscreen button labels, what the "HELP", "YES", "NO", "MENU", "FLIP" buttons do and what Yes and No mean.
prgmSTATS	Displays a series of screens navigated by the Left and Right Buttons. Each screen is part of a table that lists the first line of text on the front of cards in descending order of the ratio of the number of No(s) to the number of Yes(es). The first screen lists the first 6 entries and so on.
prgmENDMENU	TI-Basic menu that appears after the "cards left and cards re-introduced" message screen when the user walks off the last card of this pass of the stack. Allows the user to • shuffle cards (remaining "NO" cards & re-introduced cards) and repeat the stack (in order to keep studying) • pick a new stack to study

- view Statistics
- read the help screens
- quit Viewer.

prgmSHUFFLE Called when the user walks off the last card of this pass of the stack. prgmSHUFFLE first try to re-introduce hidden cards (cards with the corresponding element in List SHOW set to 0).

If the user had been pressing Yes to a card exclusively (No count is zero), the probability of re-introduction is 20%, or 1 in every 5 passes where the card hidden.

Otherwise, the probability of re-introduction of a hidden card is the ratio of Yes(es) to the total number of Yes(es) and No(s) presses. For example, if a card is currently hidden and the user pressed Yes once and No three times, the probability of the card being re-introduced is 75%. If the user pressed Yes three times and No once, the probability of the card being re-introduced is 25%.

Overall effect: Cards that the user responded with "No" several times before finally responding with "Yes" are more likely to be re-introduced than cards with less "No"s before a "Yes" response.

After the re-introduction of hidden cards and displaying the Re-introduction message. All the cards that are to be shown in the next pass of the stack have their order of presentation shuffled.

prgmQUIT Display a parting message and Stop program execution.

PIC 1 Picture of front card view (button labels and card border). Refer to prgmPICDRAW for drawing code.

PIC 0 Picture of back card view (button labels only). Refer to prgmPICDRAW for drawing code.