* Our goal is to help beginning blind students “see” simple pictures that we often easily show sighted students.
* Our method is to transform those simple pictures into figures the blind can feel.
* We use Braille printers to produce touchable figures.
* Beginners, sighted or not, are often computer savvy but not programming savvy.
* Examples may be useful but are often too abstract. Use pictures.

**show\_square\_loop\_colors.py** - An example for the sighted

A Program to display a square with colored sides

Execute the program before detailed explanation.

A screenshot of a computer

AI-generated content may be incorrect.

Comments

Just a description – drawing instructions – verbal “Etch-a-Sketch”

Language of our own – e.g., Best Friends for Ever - BFF, Lots of Luck – LOL

List of colors

For each color in list of colors

Line width(thickness, in pixels) – pixel, a tiny bit of screen or page

Line color

Move 200 pixels

Turn right 90 degrees

**show\_square\_loop\_colors\_w80.py** – same program with minor change

Just a minor change --> thicker lines

Execute program before detailed explanation.

A screenshot of a computer

AI-generated content may be incorrect.

**show\_square\_loop\_colors\_braille.py** - our first program, changed for the blind

A screenshot of a computer

AI-generated content may be incorrect.

* The colored square figure that the sighted student sees.
* Window simulation of Braille version of text map – more to come
* A “text map” in the Windows clipboard which can be Braille printed.
* When printed as Braille, this text creates a touchable version of the figure.
  + Note that our students can easily paste text into an application which prints the Braille.
  + For practice and demonstration, we use the common application notepad.exe. Notepad does no Braille printing, but it shows text and prove our pasting operation.
  + Start Notepad.exe to mimic application prints Braille.

Text map:

The graphics screen is viewed as a rectangular array of 40 columns wide by 25 rows .

One text map character stands for each one column by one row region. This character stands for this region’s color.

Each text map character is the first letter of the region’s color, “r” – red, “o”, orange,…. The character comma (,) is used for non-trailing blank/space areas.

Spaces as commas? Why? Hint: First time I used spaces!

Braille Window:

The Braille Window looks like the Braille printed from the text map.

The window presents each Braille character as a rectangle holding colored dots inside. These dots form the Braille six-dot code for the corresponding text map letter. For simplicity, we left the empty regions blank. To remind me, new to Braille, the dots’ color is the color of the picture’s region.

The user can navigate the Braille Window by using the keyboard. One method uses the keys positioned on the numeric keypad:

789

456

123

Move one square in one of the 8 directions: 4-left, 7-up to left, 8-up, 9- up to right,, 6-right, 3-down to right, 2-down, 1-down to left. Pressing “5” just announces the current color and location.

Navigation audio is available. Examples to follow

Creating text map from screen figure drawing:

Our software evaluates each one row by one column region:

If the region has color, the letter of this color is placed in the text map.

If the region is empty, the space character is used in the text map.

A grid with a colorful square

AI-generated content may be incorrect.

To aid Braille printing, our software performs the following actions:

* The colored figure is moved toward the upper left corner, reducing excess spaces.
* The Braille software normally compresses repeated spaces. This distorts our graphic figures. Non-trailing spaces are replaced by commas (a single dot Braille character), to avoid this distortion.

**show\_square\_loop\_colors\_w80\_braille.py**

The smallest change– thicker line**.**

**A screenshot of a computer

AI-generated content may be incorrect.**

Note the thicker lines – approximate because of the coarse rendition

Paste the new map into Notepad - can send multiple pictures at once – saves walking

Navigate on Braille window, using numeric pad

4(left), 8(up), 3(down to right), 3, 3

Note that the **Braille window gives audio/speech feedback**.

Spoken feedback:

1. The key pressed
2. The region’s color
3. The current location

Example:

“green at row 13 column 19” when Inside figure

or

“right 1 down 1” as instructions how to move to the figure when outside the figure.

Magnification

Select rectangle enclosing ALT+M, S

A screenshot of a game

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

The magnification operation can be repeated.

A screenshot of a computer

AI-generated content may be incorrect.

What can you do?

Our tool is a software program, written in the Python programming language. To use this tool, the user writes or is provided with a simple Turtle program to create a simple graphic such as a square. When run this program generates the standard Turtle graphics output, plus text output which when sent to a standard Braille embosser produces a “touchable” rendition of the square.

Our tool, while functional, is a prototype. We could use lots of help.

Our greatest need is a small user community which can use the tool and provide us with feedback as to its usefulness. We can use information as to the tool’s strengths and shortcomings. We would appreciate suggestions on how we can make our tool more useful. Where is it in need of improvement? What are the best points?

The program needs packaging so that we can better distribute it. The software needs testing.

Options, such as “Expand to Top” were stubbed out, and need work. Experimental features such as perimeter scanning (the audio presentation of the perimeter of the graphics figure), need improvement.

Research, or development, of available, low cost, tools such as a reusable high speed Braille tablet would be great. Integration with high performance touch screens might be helpful.