Leone Learning Systems, Inc.
Wonder. Create. Grow.

Leone Learning Systems, Inc.
237 Custer Ave
Evanston, IL 60202
Emal tj@leonelearningsystems.com

Phone 847 951 0127
Fax 847 733 8812

Berkeley Logo

An introduction and quick reference guide

TJ Leone

October 2004

Introduction

This guide was written to introduce you to features that are special to Berkeley Logo (also referred to as UCB Logo). It does not assume that you have used Logo before, but it doesn't try to teach you Logo, either. You'll need to look elsewhere to learn details of the language.

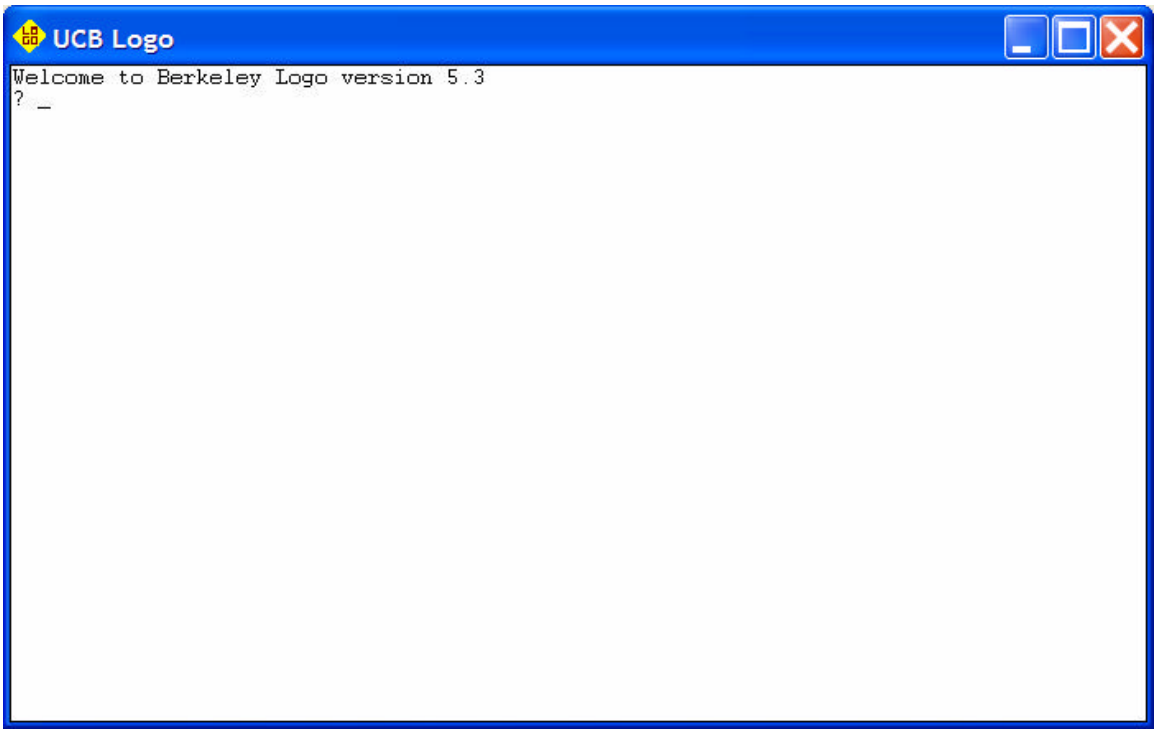
You should look at this document while you have Berkeley Logo running on your computer. If you don't understand all the examples on your first reading of this document, that's OK. The purpose of your first reading is to get comfortable enough with the Berkeley Logo environment so that you can write and run example procedures. You can start writing procedures of your own later on, when you understand the Logo language better.

As you learn more about Logo, you can use this document as a reference. There are some nice features of Berkeley Logo that you might not appreciate on a first reading, such as the Step and Trace functions.

Berkeley Logo was built by Brian Harvey at the University of California at Berkeley. Brian's web site (http://www.cs.berkeley.edu/~bh/) is a great source of information on Logo, especially the free downloadble PDFs for the three volumes of the second edition of *Computer Science Logo Style*.

The Berkeley Logo Screen

When you start up your Berkeley Logo application, you'll see a window that looks like the one in the picture below.



Entering Commands

Try typing

```
? cs
```

at the Logo prompt. Remember, the question mark is the prompt. You don't type a question mark. You only type `cs`. Then press the Enter key. You should see:

Whenever you send Logo a turtle command (e.g., cs, rt, fd, setpos, setheading), Logo brings up the graphic portion of the screen and executes the command. The triangle in the middle of the screen is called a *turtle*. You can switch between screen modes with the commands `fullscreen` (for graphics mode) and `textscreen` (for text mode).

Try some other commands, like

```
? rt 90
? fd 60
```

Then type

```
? cs
```

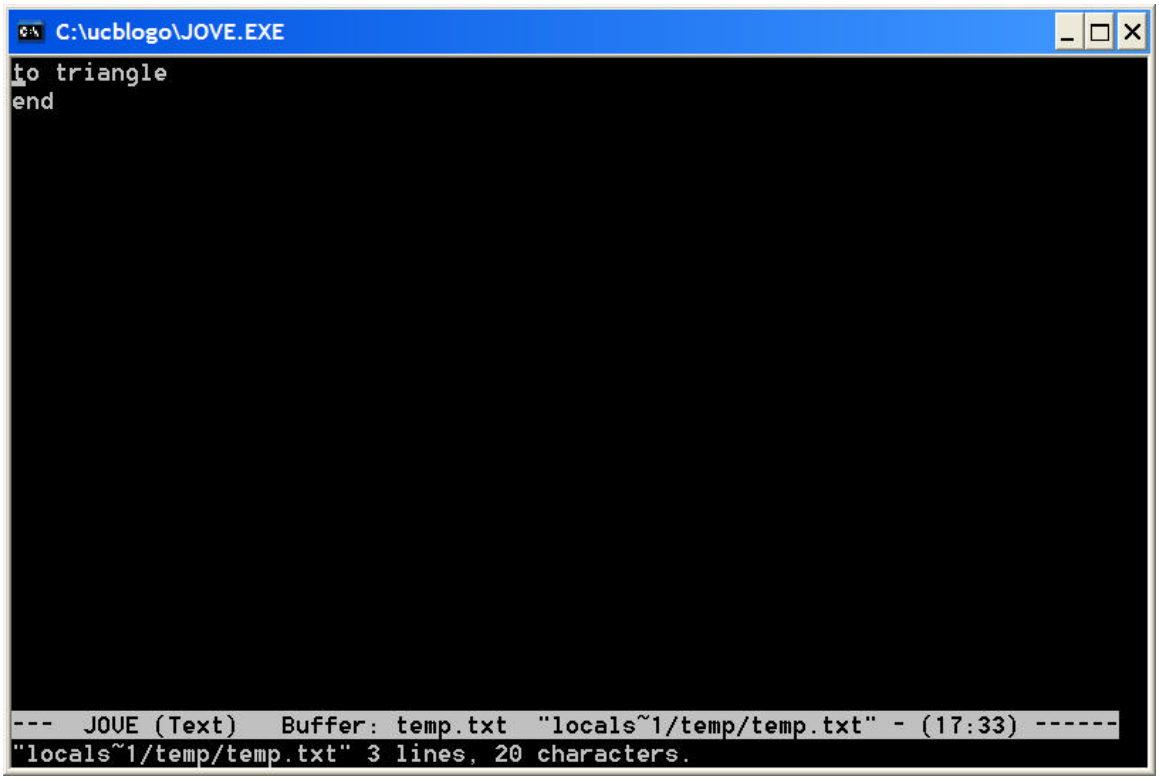
again. What happens?

Writing Procedures

Click in the Input Box and type

```
? edit “triangle
```

The edit window will appear:



Notice the highlighted line near the bottom of the screen. JOVE is the name of the application you brought up with the edit window. There are instructions available on Brian Harvey’s web site for replacing JOVE with your favorite editor, if you like.

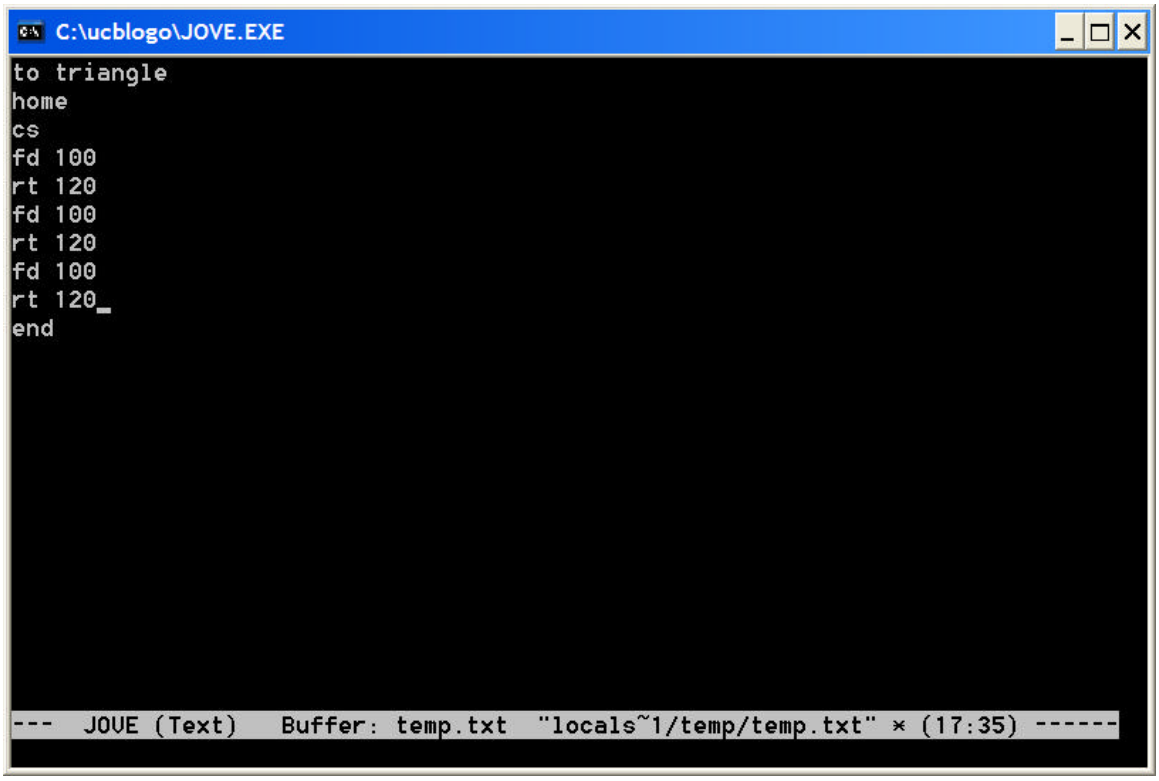
After JOVE (Text) is the word Buffer. A buffer is a temporary holding place for information. The word temp.txt that follows Buffer: is the default name of the file where the buffer will be written. For other text editing applications, like Microsoft Word, we usually use the expressing “saving a file” instead of “writing a buffer”.

The file name “temp.txt” can be of as the default names “Untitled”, or “Document1” that you see in other applications. It’s probably not the name you actually want to give to your file. We’ll see how to specify a file name shortly.

Writing Procedures (continued)

Add lines to your procedure so that the completed procedure looks like the below. Start by using the arrow keys on the keyboard to move the cursor past the "e" in `triangle` on the first line, then press the Enter key to start a new line.

In the picture below, the cursor is positioned after the last `rt 120`. There's no need to add any more lines at this point, but if you wanted to add a line after the last `rt 120`, you would do so by pressing the Enter key with the cursor in its current position.



Hold down the Control key. This key is usually labeled Ctrl. While you still have the Control key down, press X. Keep the Control key down and press W. In the rest of the document, abbreviations such as Ctrl-X and Ctrl-W will be used to mean "Hold down the Control key and press X" (or W or whatever).

Type the name of a file in which you'd like to save your procedure, and press Enter. I called my file "myfile".

Close the jove window with Ctrl-X Ctrl-C.

Writing Procedures (continued)

To load the procedure into Berkeley Logo, I used the command:

```
? load “myfile
```

Now run your procedure by typing “triangle” into the input box and pressing the Enter key.

```
? triangle
```

What happens?

Stepping through Procedures

Enter the following instruction:

```
? step “triangle
```

Try running the triangle procedure again. The following will appear

```
? triangle
[home] >>>
```

Press Enter again. Keep pressing Enter and see what happens. You should eventually see all of the instructions in the triangle procedure as Logo carries them out:

```
? triangle
[home] >>>
[cs] >>>
[fd 100] >>>
[rt 120] >>>
[fd 100] >>>
[rt 120] >>>
[fd 100] >>>
[rt 120] >>>
?
```

You can use the step command when you want to slow down the action in a procedure in order to see what’s happening, one step at a time. When you want to execute triangle again without stepping through it, use

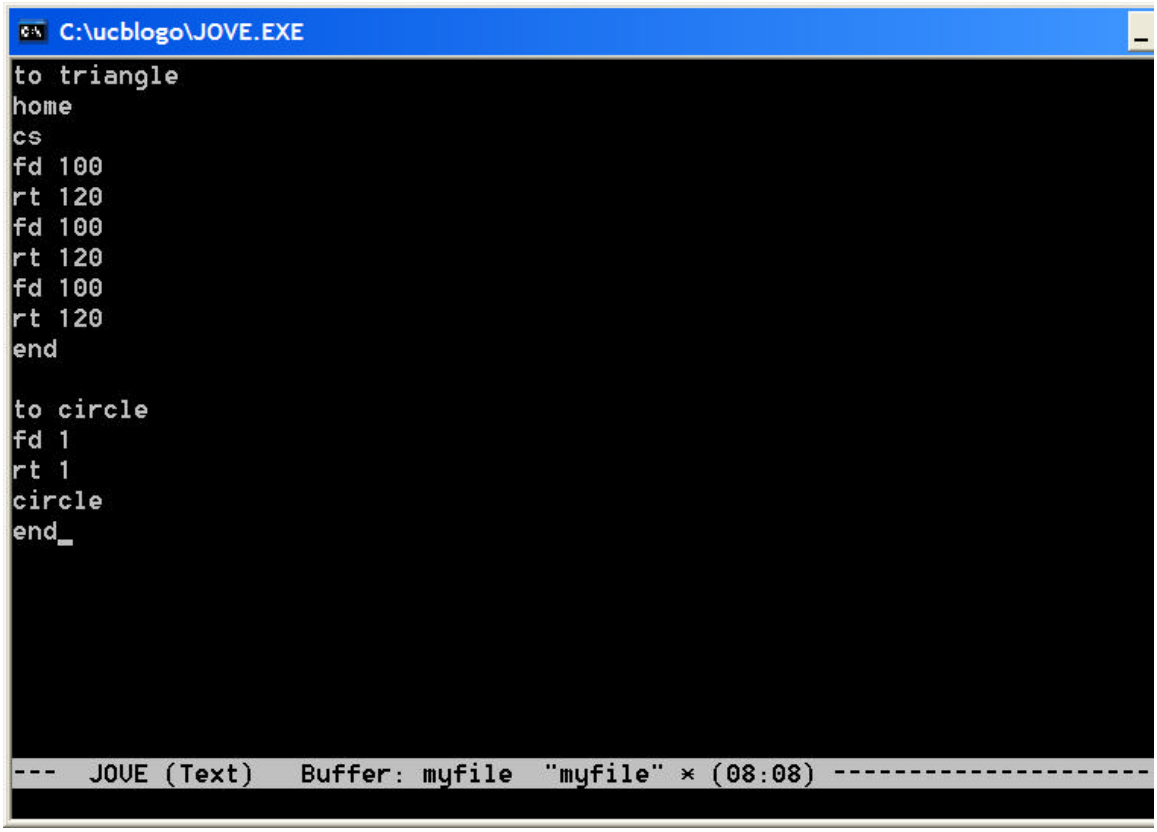
```
? unstep “triangle
```

Halting Procedures

Enter the following at the Logo prompt:

```
? editfile “myfile
```

Then press the Enter key or click on the Execute button. Add the circle procedure to your program buffer:



Press Ctrl-X Ctrl-W. At the bottom of the screen, you should see a message that says:

```
: write-file (default myfile)
```

Press the Enter key, then press Ctrl-X Ctrl-C to exit.

Halting Procedures

Type the following in your Input Box to move the turtle to the center of screen and clear the screen:

```
? home cs
```

Then press the Enter key or click on the Execute button. Now execute the circle procedure by typing

```
? circle
```

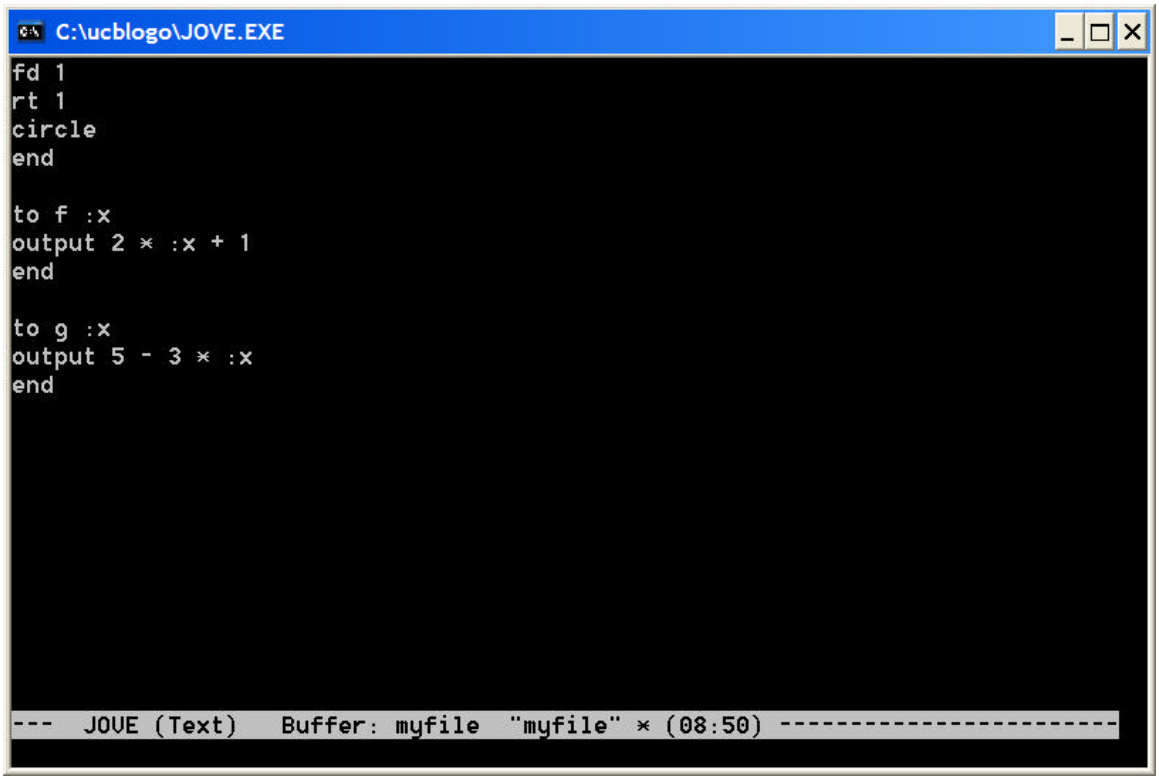
at the Logo prompt and pressing Enter.

To stop the turtle, press the interrupt keys for your computer:

DOS/Windows	Macintosh	Unix
Ctrl-Q	Command-.	Usually Ctrl-C

Tracing Procedure Calls

If you don't have your Editor window open, click on the Edall button to open it. Now add the procedures f and g as show below:



Be careful with spaces. There should be a space between the `f` and the `:x`, but no space between the `:` and the `x`. Also make sure you have spaces between words and numbers, and spaces on either side of the `*` and `+` operators. Write your buffer (save your file) with Ctrl-X Ctrl-W, and close the editor window with Ctrl-X Ctrl-C.

At the Logo prompt, type

```
? show f 1
```

and press the Enter key. You should see:

```
? show f 1
3
```

What's happening?

Tracing Procedure Calls (continued)

The `trace` command helps us get an inside view of how a procedure is evaluated.

First, let's make sure we have room to see what's being traced. If you're not in text mode (i.e., if you can see the turtle on your screen), enter text mode with the command

```
? textscreen
```

Now let's try to trace the `f` operation. Enter the following:

```
? trace "f
```

Execute `show f 1` again by typing it at the Logo prompt and pressing Enter. You should see this:

```
? show f 1
     ( f 1 )
     f outputs 3
3
```

Why do you suppose `f` outputs `3`? Remember, the output of the `f` procedure is

```
2 * :x + 1
```

Try using other inputs to the `f` function. For example, try:

```
? show f 10
? show f 2
? show f 5
```

Can you see what's happening? If not, don't worry about it at this point. We have plenty of time to get a good understanding of functions like this one.

If the outputs make sense to you, try to predict outputs for `g` and then execute

```
? show g 1
? show g 2
```

Tracing Procedure Calls (continued)

You can also trace the `g` procedure. If you feel comfortable with both the `f` and `g` functions, try combining them:

```
? show f g 3
? show f f 5
```

You can also trace Logo instructions like `show`. To trace a group of procedures, put them in a list, like this:

```
? trace [f g show]
```

The Author

TJ Leone owns and operates Leone Learning Systems, Inc., a private corporation that offers tutoring and educational software. He has a BA in Math and an MS in Computer Science, both from the City College of New York. He spent two years in graduate studies in education and computer science at Northwestern University, and six years developing educational software there. He is a former Montessori teacher and currently teaches gifted children on a part time basis at the Center for Talent Development at Northwestern University in addition to his tutoring and software development work. His web site is http://www.leonelearningsystems.com.