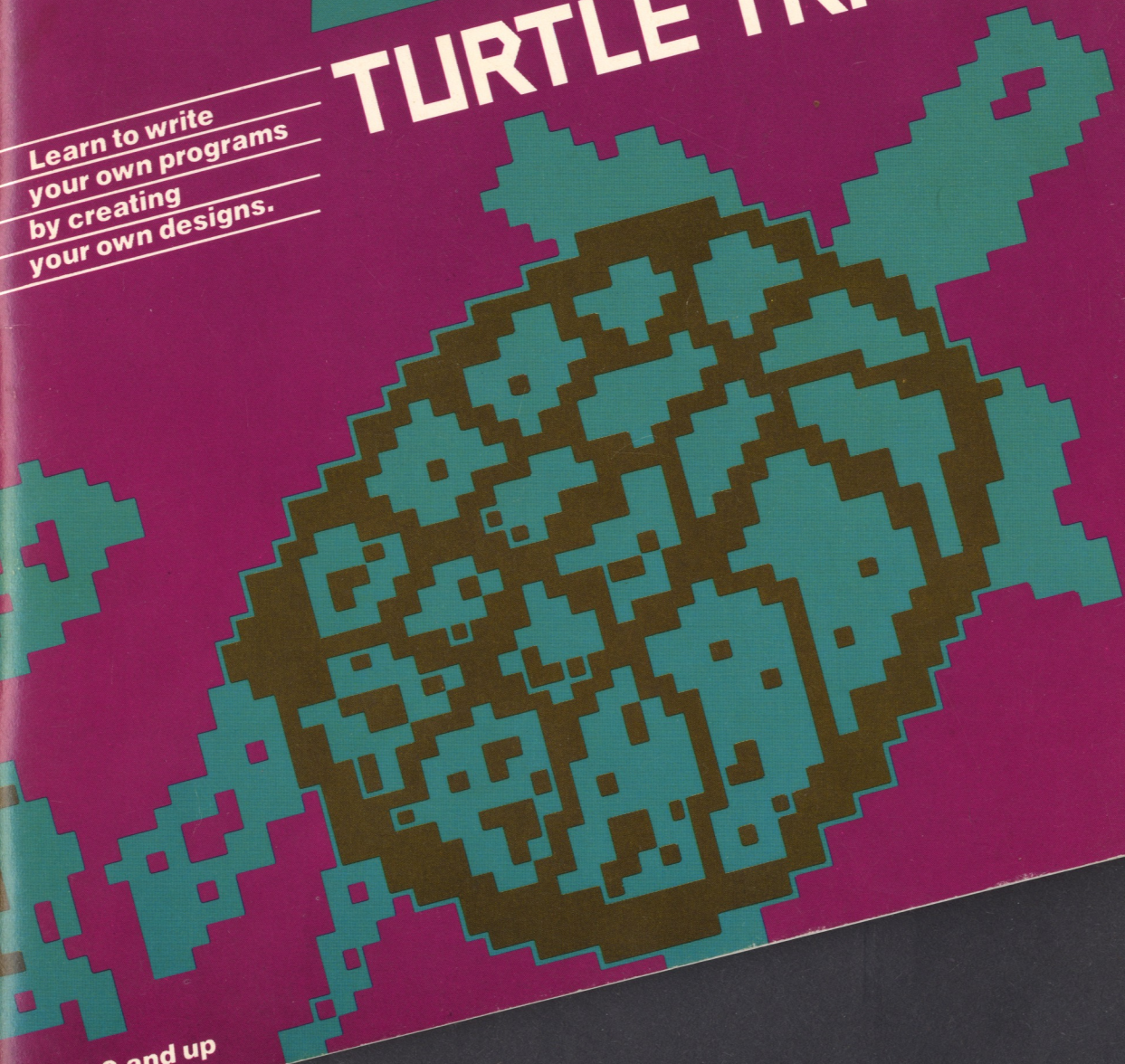


Scholastic **Wizware**™

TURTLE TRACKS™

Learn to write
your own programs
by creating
your own designs.

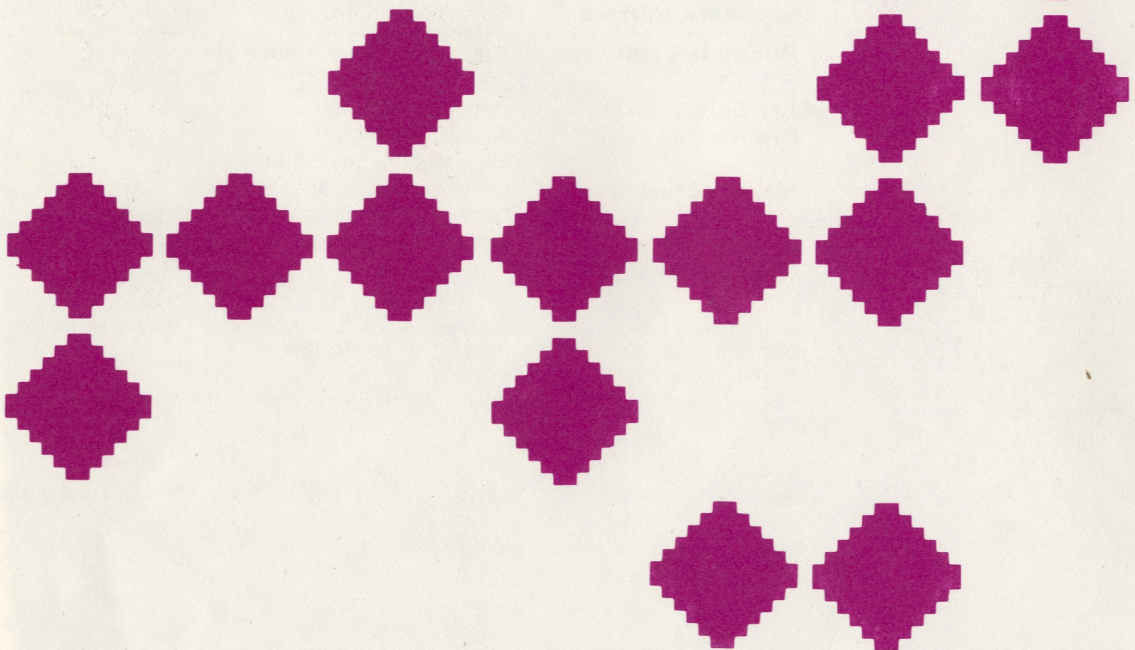
Ages 9 and up



Scholastic **Wizware**™

Turtle Tracks™

Designed and Developed by Thomas R. Smith with
James F. Wieder



Scholastic

New York Toronto London Auckland Sydney Tokyo

Designer, Developer Programmer	Thomas R. Smith with James F. Wieder
Software Research	This program was field-tested at Park School in Brookline, MA.
Creative Director	Deborah Kovacs
Project Manager	Stephen Gass
Software Editors	Robert Neumann/Jeffrey Siegel
Handbook Authors	Thomas R. Smith/Robert Neumann
Art Editor Production	Dennis Niswander Barbara Kellogg, Director Maryellen Kohn, Editor
Manufacturing	Louis deRienzo, Manager Deborah Honig, Assistant Manager
Package and Graphic Design	Robert P. Gersin Associates, Inc.

Notice: This work is fully covered by the Copyright Law of the U.S. (Title 17 of the U.S. Code) and the Universal Copyright Convention. Unauthorized copying is strictly prohibited.

Software: Copyright © 1982 by Thomas R. Smith.

Handbook: Copyright © 1983 by Scholastic Inc. All rights reserved. Printed in U.S.A. Published by Scholastic, Inc.

ISBN: 0-590-95602-7

12 11 10 9 8 7 6 5 4 3 2 1 4 3 4 5 6 7 8/8

Introduction

Welcome to *Turtle Tracks*

You are about to team up with a talented and unusual creature—the turtle. This turtle lives inside your computer, and paints, draws, and sings with great ability.

The trouble is, the turtle doesn't know what to do with its talents—and that's where you come in. Your job: to provide the brains and the imagination that the turtle so sorely needs to put its talents to good use. You become the artist, the musician, and the architect. You use the turtle as your paintbrush, musical instrument, and set of building blocks.

Turn your imagination loose. Do you want to paint a tropical island sunset—mango trees under a red sun? Design it and use the turtle as your paintbrush. Do you want to compose a hard-driving rock song? Write it and let the turtle play it.

This handbook will teach you to talk to the turtle. And as you learn to do that, you'll also be picking up another valuable skill—computer programming. *Turtle Tracks* is actually a miniprogramming language. As such, it shares lots of features with LOGO—another language that uses turtle graphics—and BASIC, the world's most popular language for microcomputers.

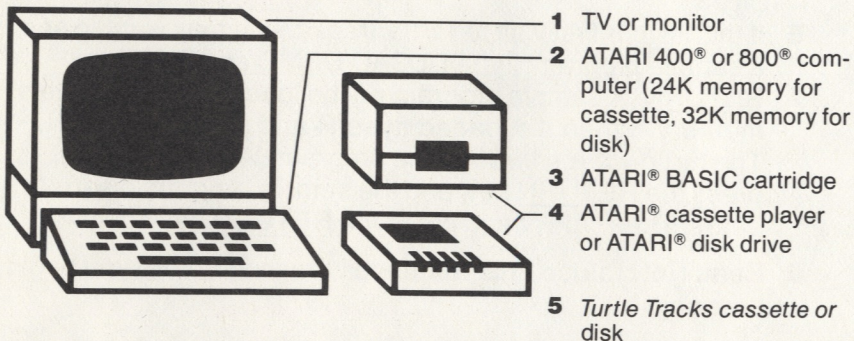
Have fun!

Contents

Introduction	3
How to Load <i>Turtle Tracks</i> Into Your Computer	5
How to Use This Handbook	8
Section I: Getting on Track	
Lesson 1: Talking to the Turtle	9
Lesson 2: More Turtle Talk	13
Lesson 3: A Second <i>Turtle Tracks</i> World	17
Section II: Picking Up Steam	
Lesson 4: Round and Round in Loops	21
Lesson 5: The Cast of Characters	26
Lesson 6: Sound and Color!	33
Lesson 7: Variables	40
Section III: Full Speed Ahead	
Lesson 8: Teaching the Turtle New Words	44
Lesson 9: Looping the Loops	50
Lesson 10: More About Miniprograms	55
Appendix A: Saving and Loading Your Work	59
Appendix B: A Glossary of <i>Turtle Tracks</i> Instructions	66
Appendix C: Some Common Mistakes	68
Appendix D: A Short Collection of Interesting Programs	69
Challenge Solutions	71

How to Load *Turtle Tracks* Into Your Computer

Equipment You Need



ATARI is a registered trademark of Atari, Inc., A Warner Communications Co.

Disk Loading Instructions

Follow these directions if your *Turtle Tracks* is on a disk. If it's on a cassette, skip to *Cassette Loading Instructions*.

- 1 Turn on your disk drive and TV or monitor.
- 2 Make sure your Atari BASIC cartridge is in the computer (left cartridge slot if you're using the Atari 800).
- 3 When the busy light on your disk drive goes off, insert your *Turtle Tracks* disk with the label facing up. Close the disk drive door.

NOTE: Never turn the disk drive on or off while there is a disk inside. This might damage the disk.

- 4 Turn on your computer. The disk drive will make a few clicking and whirring sounds. A Scholastic *Wizware* screen will appear, followed by a *Turtle Tracks* title screen. Then, a white rectangle will appear in the upper-left corner of the screen. Welcome to *Turtle Tracks*!
- 5 If this is your first visit to *Turtle Tracks*, you may want to see a sample program. Type DLOAD DEMO and press RETURN. A sample program will be loaded into the turtle's memory. When it's loaded, type R and press RETURN. The turtle will run through the program. When it finishes, press the RETURN key, and the sample program disappears. Then type N and press the RETURN key again.
- 6 Read How to Use This Handbook (page 8) before you begin Lesson 1.

Cassette Loading Instructions

- 1 Turn on your computer and your TV or monitor.
- 2 Type:
RUN "C:"
Press RETURN. The computer buzzes once. Press the PLAY button on the cassette player. Then press RETURN on the computer keyboard.
- 3 You'll soon see a Scholastic *Wizware* screen. When you do, press RETURN on the computer keyboard.

- 4** *Turtle Tracks* will take about four minutes to load in. When a READY message appears, type RUN and press RETURN. A white rectangle will appear in the upper-left corner of your screen. Welcome to *Turtle Tracks*!
- 5** If this is your first visit to *Turtle Tracks*, you may want to see the turtle run through a sample program. Type LOAD DEMO into the computer and press RETURN. The computer beeps once.
- 6** Press the RETURN key on the computer keyboard again. The screen will soon show:
LOADING . . .
And then
FILE LOADED or VERIFIED
- 7** Press STOP on the cassette player.
- 8** Type R, press RETURN, and watch the turtle run through the sample program.
- 9** When the turtle finishes, press the RETURN key and the sample program disappears. Then type N and press the RETURN key again.
- 10** Read How to Use This Handbook (page 8) before you start Lesson 1.

How to Use This Handbook

Work through the 10 easy-to-follow lessons in this handbook in order. After each lesson, play with the challenges we've suggested. They'll help you practice and apply what you have learned. Then, take as much time as you want to experiment with the turtle. That's really the only way to get comfortable with it. You'll learn as much from your mistakes as you will from your successes!

More information about *Turtle Tracks* is in the Appendices at the back of the handbook. In Appendix A (page 59), you'll learn how to save your hard work in *Turtle Tracks* on disks, cassettes, or paper. Read Appendix A when you're ready to save some of your work.

Appendix B (page 66) contains a list of all the instructions the turtle understands, and the lessons where you can find more information about the instructions. Consult Appendix B if you've forgotten how to use a certain instruction.

If the computer doesn't understand something you type, it will tell you so in a screen message. Correct your mistake and continue working. Appendix C (page 68) contains more information about common mistakes.

If you find yourself hopelessly lost in the middle of a lesson, type N, press the RETURN key, and start over at the beginning of the lesson.

And now—let's begin!

Section I: Getting on Track

Lesson 1: Talking to the Turtle

If you've loaded *Turtle Tracks* into your computer, you should see a small square at the top left of your screen. That's the *cursor*. The cursor shows where your typing will appear. Type your name. For example:

THOMAS SMITH

As you can see, the cursor moves along as you type.

Delete

On the top right of your keyboard you'll find an important key. It has the letters BACK S (back space) and DELETE on it. DELETE means to erase. Press this key. When you do, the cursor moves back one space and gobbles up the last letter of your name. You can now type a new letter in that space. Try it. Then keep pressing the DELETE key until the cursor erases all the letters in your name.

Numbers and Letters

Typists sometimes use lower-case L (l) to represent the number one (1), and upper-case O to represent the number zero (0). Computers don't let you do that. You must use the number keys only for numbers and the letter keys only for letters.



Draw Forward

Let's start talking to the turtle. Type in:

```
1 DF3
```

You have just typed an instruction that tells the turtle to Draw Forward three steps. Tell the turtle that you have finished giving it an instruction by pressing the RETURN key. The cursor will move to the next line.

Run

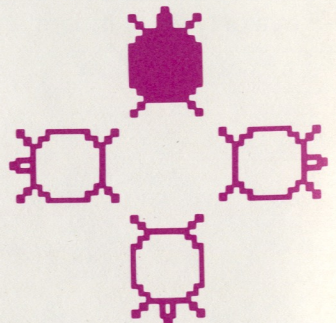
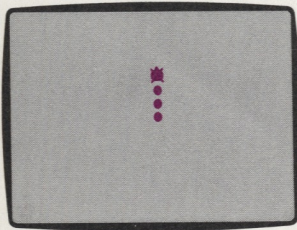
Now, type R, for Run, and press RETURN. This tells the turtle to carry out any instructions that you've given it so far. The turtle should Draw Forward three steps. It leaves behind a trail of three balls.

Line Numbers

The number one (1) that begins instruction 1 DF3 is the instruction's line number. Line numbers determine the order in which instructions will be carried out.

Two Screens

The screen you are looking at now is called the *turtle screen*. Press RETURN. The turtle disappears, and the cursor shows up again. This screen is called the *blackboard*. You'll be using the blackboard to give the turtle instructions.



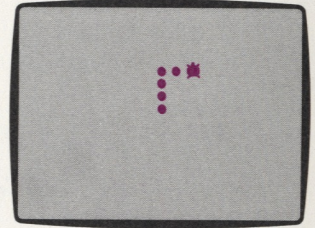
Turns

Type the following three instructions carefully. If you make a mistake, correct it with the DELETE key. Press RETURN after each instruction. (If the turtle finds a mistake, it will send you a screen message. Compare what's on the screen with what's here in the handbook, and then type the line over correctly.)

Type:

```
2 TR          [Press RETURN.]
3 DF2        [Press RETURN.]
4 TL          [Press RETURN.]
```

Type R and press RETURN. Here's what should happen. The turtle remembers line 1, DF3, and Draws Forward three steps. The instruction in line 2, TR, tells the turtle to Turn Right, and the turtle does so. The turtle then performs line 3, DF2, and Draws Forward two steps. Finally, the turtle obeys line 4, TL, and Turns Left.



List

Return to the blackboard. (Press RETURN.) Now type L, for List, and press RETURN. The instructions you've given the turtle so far will appear.



New

Now type N, for New, and press RETURN. N tells the turtle to erase old instructions, and prepare for new ones. If you type L now and press RETURN, you'll see that the turtle doesn't have any instructions to List. If you type R and press RETURN, you'll see that the turtle doesn't have any instructions to Run.

Challenges

Now is the time to start experimenting with the turtle. Return to the blackboard. (Press RETURN.) Give the turtle some instructions to carry out. Remember to begin instructions with line numbers, and to finish them by pressing the RETURN key. When you want to Run your instructions, type R and press RETURN. (The command R does not use a line number).

If you get a screen message and you don't know why, turn to Appendix C (page 68).

1. Draw a square. (In *Turtle Tracks*, a square means a figure with the same number of balls—or other characters—on each side. If you used a ruler, you'd find that the lengths aren't exactly the same).
2. Draw a square with exactly five balls on each side. (You may be a bit surprised by the answer!)
3. Draw a rectangle whose length is seven and width is three balls. (Another surprise!)
4. Draw a capital letter of your choice.
(Suggested Solutions on page 71.)

Lesson 2: More Turtle Talk

Jump Forward

Return to the blackboard (press RETURN). Type N, press RETURN, then type:

- 1 DF4
- 2 TR
- 3 JF5

Don't forget that you must press RETURN after each instruction, including the last one. Now type R and press RETURN. The JF instruction in line 3 tells the turtle to Jump Forward five steps. (Believe it or not, DF, JF, TR, and TL are the only instructions you'll need to move the turtle about the screen.)

Changing Instructions

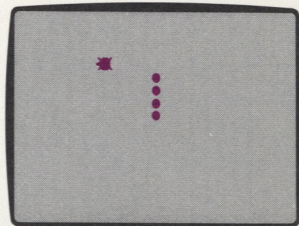
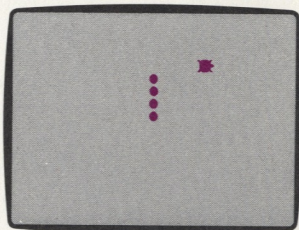
Return to the blackboard. Type L, press RETURN, and the following instructions should appear:

- L
- 1 DF4
 - 2 TR
 - 3 JF5

Here's how you can change an instruction. Change line 2 so that the turtle Turns Left by typing:

- 2 TL

Press RETURN after you type in the instruction, of course. Then type R and press RETURN. The final screen should look like the one at right:



What happened to the old instruction in line 2 (TR)? The turtle replaced it. The turtle automatically throws out old instructions when it sees new ones with the same line number. Return to the blackboard, type L, and press RETURN. As you can see, line 2 now contains the new instruction.

Erasing Instructions

To erase an instruction, simply type the line number and press RETURN. For example, type:

2

Press RETURN. Now, type L and press RETURN. Line 2 has vanished! Run the program (type R and press RETURN) and you'll see that the turtle doesn't care that line 2 no longer exists. The turtle goes straight from line 1 to line 3.

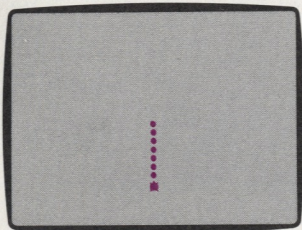
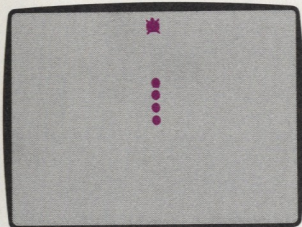
Backward Motion

Use numbers with minus signs (-) in front of them to make the turtle jump or draw backward. Return to the blackboard and type:

4 JF-10

5 DF-3

Run the program. (Type R and press RETURN). The turtle first carries out line numbers 1 and 3. (You erased line 2, remember?) Then, it jumps backward 10 steps (line 4) and draws backward three steps (line 5).





Line Numbers

Suppose you want to place an instruction between lines 4 and 5 of the program you just ran. You may do so by using line numbers with decimals. Return to the blackboard and type:

```
4.5 TR
```

Run the program (type R and press RETURN), and then List it (return to the blackboard, type L, and press RETURN).

You'll see that line number 4.5 has been properly inserted between lines 4 and 5. As you can see, the turtle doesn't care *when* it receives instructions. It only cares about their line numbers.

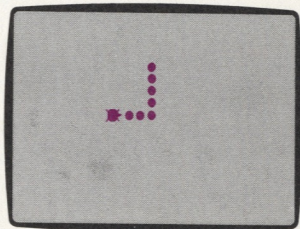
Most programming languages don't allow decimal line numbers. So, programmers often number their lines by tens (10, 20, 30, 40, etc.) to leave plenty of room to insert new lines. You may do that in *Turtle Tracks*, too.

More About Lists

You may list a part of a program instead of the entire program. Here are some ways to do this:

- | | |
|---------|---|
| L2 | [List line 2 only.] |
| L3- | [List line 3 and all following lines.] |
| L-20 | [List all lines up to and including line 20.] |
| L5.4-30 | [List lines 5.4 through 30.] |

Try different L commands with the program now in the turtle's memory. Use the numbers in the examples to see what happens, and then try your own line numbers.



Commands and Statements

You may be wondering why some instructions use line numbers and others don't. The answer is that there are two kinds of instructions:

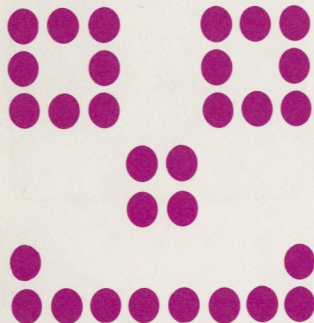
- *Statements* are parts of programs, and require line numbers. DF, JF, TR, and TL are statements.
- *Commands* tell the turtle what to do with programs. Since they are not part of programs, they do not use line numbers. L, N, and R are commands.

Challenges

Practice with the JF (Jump Forward) statement, and use minus numbers to jump backward and draw backward. Number your lines in different ways. For example, use decimals, or space line numbers 10 apart.

Try the following challenges:

1. Draw a square at the bottom of the TV screen.
2. Draw one square in the top-right corner of the screen, and another in the bottom-left corner.
3. Draw a slanted line.
4. Draw a capital N.
5. Draw a face. Make up your own, or try this one: (Suggested solutions on page 72).



Lesson 3: A Second Turtle Tracks World

Return to the blackboard, then type N and press RETURN.

Type:

```
5 TR
10 DF18
```

Run this program (type R and press RETURN) and watch the turtle scoot over to the right side of the screen. Now return to the blackboard. Type:

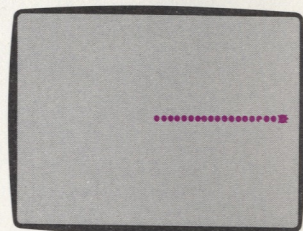
```
GR1
```

Press RETURN. The border color of your screen turns yellow. Now Run the program again (type R and press RETURN). This time, the turtle scoots off the right side of the screen, reappears on the left, and continues drawing!

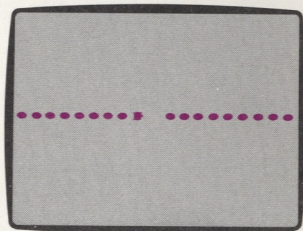
What happened? The command GR1 did more than change the border color. It told the turtle to enter Graphics One (GR1), a new *Turtle Tracks* world. The laws of GR1 are different than the laws of GR0 (Graphics Zero) the world the turtle has been in so far. You can see that the balls in GR1 are larger, look different, and that fewer of them fit on the screen.

Return to the blackboard and type:

```
GR0
```



GR0



GR1

Press RETURN. The screen border returns to red, and the turtle is back in Graphics Zero. When you're working on the blackboard you can tell whether the turtle is in GR1 or GR0 by the border color—red for GR0, yellow for GR1. The turtle moves from one graphics world to another only when you command it to switch with the proper GR command. Because GR0 and GR1 are commands, they do not use line numbers and cannot be used inside programs.

Each graphics world has advantages and disadvantages. You'll learn more about the differences between them throughout this handbook.

Screen Color

You may change the background color of the drawing screen by adding another number to the GR0 and GR1 commands. For example, type:

GR1;5 [You'll find the semicolon (;) next to the letter L on the computer keyboard.]

Press RETURN, and then go to the turtle screen by typing R and pressing RETURN again. Violet!

The command GR1;5 has two parts to it. The first part, GR1, is familiar. It tells the turtle to use Graphics One. The second part, the number 5, selects the background color. The chart on the next page shows the 16 background colors you may choose. The number 5, as you can see, selects violet.

These color commands only change the background color of the turtle screen, not the blackboard.

Color Chart

(For screen color and character color)

Color Number	Color
0	Gray
1	Gold
2	Orange
3	Red
4	Pink
5	Violet
6	Blue-purple
7	Blue
8	Light blue
9	Blue-green
10	Aqua
11	Green-blue
12	Green
13	Yellow-green
14	Orange-green
15	Orange



Reminders

You may leave yourself reminders inside programs. Use these reminders to help you remember what each part of a program does, or use them as titles. Return to the blackboard. Type N, press RETURN, then type:

```
20 'Draw 2 Lines [The apostrophe (') is on the 7 key. Hold
30 DF7           down the SHIFT key and press the 7 key.]
40 TR
50 DF6
```

Run this program, and then List it. Line 20 contains the *reminder*. The apostrophe (') tells the turtle to skip over the instruction, but it's there for you to see when you type L. Reminders may be as long as 20 characters, including the line number.

Clear

Hold down the SHIFT key and press the CLEAR key to erase the blackboard. You'll find the CLEAR key on the upper-right side of the keyboard. The CLEAR key also places the cursor at the top left of your screen. It does not affect the turtle's memory.

Challenges


1. Draw the largest rectangle possible in GR0.
2. Draw the largest square possible in GR1.
3. Draw the largest rectangle possible in GR1.
(Suggested Solutions on page 73.)

Section II: Picking Up Steam

Lesson 4: Round and Round in Loops

Return to the blackboard. Type N and press RETURN to clear the turtle's memory. Then type:

```
1 DF5  
2 TR  
3 GT1
```

A diagram showing a loop from line 3 back to line 1. A horizontal line connects the end of line 3 to the start of line 1, with a vertical line and an arrow pointing up from the horizontal line to line 1.

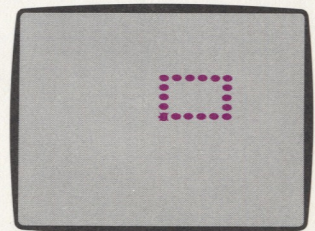
[This tells the turtle to Go To line 1.]

Run this program in Graphics One. (Type GR1, press RETURN, then type R and press RETURN.) You'll see that the turtle draws a square.

Here's how this program works. The turtle carries out the instruction in line 1 and Draws Forward five steps. Then it carries out the instruction in line 2 and Turns Right. In line 3, the turtle is told to Go To line 1. The turtle does so. It carries out the instructions in lines 1 and 2 again, finds itself back at line 3, and is sent to line 1 again.

Look at the arrow that connects lines 1 and 3. Do you see why we call this a loop? The turtle loops, or cycles, through these instructions over and over again.

If you look at the screen now, you'll see that the turtle is still tracing the square. The turtle can't escape from this loop. Every time it reaches line 3, it must go right back to line 1 again. Because this type of loop has no end, we call it an *infinite loop*.



GR1

Stopping and Starting

Stop the turtle by pressing the RETURN key (almost any other key will do, too). If you now press the yellow START key on the right side of the keyboard, the turtle will continue tracing the square.

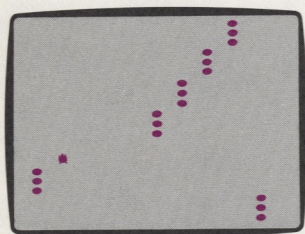
You can start and stop the turtle as many times as you want. To stop the turtle for good, and to return to the blackboard, simply press the RETURN key twice. Pressing START now won't start the turtle again.

Are you back to the blackboard? Go there now (press RETURN once or twice, whichever is necessary). Now type N, press RETURN, and type in another program with an infinite loop:

```
1 DF3 ←
2 TR
3 JF2
4 TL
5 GT1 —
```

Run the program, and watch the turtle for a while. The turtle draws three balls, hops over to the right two steps, turns left, draws three balls, hops over again, and keeps repeating these steps.

You'll notice that the turtle sometimes wanders off one side of the screen. Watch carefully, and you'll see it reappear on the opposite side. We saw the turtle do this before. The turtle



GR1

treats the screen as though opposite sides curled around and actually touched each other.

When you're tired of watching this program, press RETURN twice. Then type N and press RETURN.

A New Loop

Let's look at a different type of loop. First switch to GR0 (type GR0 and press RETURN), then type:

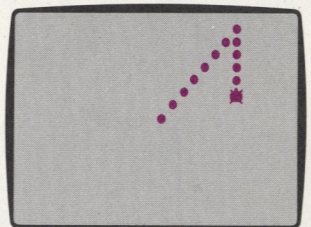
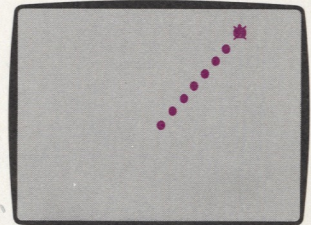
```
1 BL ← [This tells the turtle to Begin Loop.]
2 TR
3 DF1
4 TL
5 JF1
6 RL7 — [This tells the turtle to Repeat Loop seven times.]
```

Run this program and you'll see the turtle draw a slanted line. This type of loop begins with a BL statement (line 1) and ends with an RL statement (line 6).

Unlike loops that use GT statements, this type of loop has an end. Here, the end comes after the turtle repeats the loop seven times. The turtle can then go on to new instructions. For example, return to the blackboard and type:

```
7 DF-5
```

Run the program, and you'll see that the turtle draws the slanted line, and then draws backward five steps, as it's instructed to do in line 7.





GRO

Walk

You can examine loops more closely with a new command. This command is **W**, for Walk. **W** tells the turtle to carry out its instructions one at a time. You control the turtle's pace with the SPACE BAR. Try it. Return to the blackboard, type **W**, and press RETURN. The computer screen should then show:

PRESS SPACE BAR.

Walk the turtle through the program. Each time you press the SPACE BAR, a new instruction appears on the screen, and the turtle carries it out. Notice that when the turtle reaches line 6, it returns to line 2, the first real instruction in the loop. The turtle will perform the instructions in the loop seven times. Then it leaves the loop and goes on to the next instruction. That instruction, **7 DF-5**, tells the turtle to draw backward five steps, and the turtle does so. You may leave a Walk by pressing any key except the SPACE BAR.

Debugging

The **W** command is an excellent debugging (correcting) tool. Even expert programmers find that their programs don't generally work correctly the first time around. They then try to debug the program, as though nasty, unwanted bugs were to blame for the problems. Of course, they don't find bugs—but they do eventually find their mistakes. Use the **W** command and you will, too.

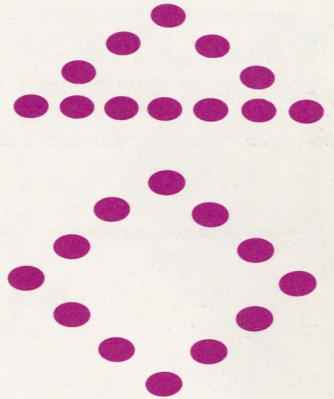
Challenges

You may find it helpful to use graph paper to design your programs. First, figure out the size of the screen. Do this using DF or JF statements to determine when the turtle leaves each edge of the screen. Then, outline the screen on graph paper. Trace the patterns you want the turtle to draw, then give the turtle the proper instructions.

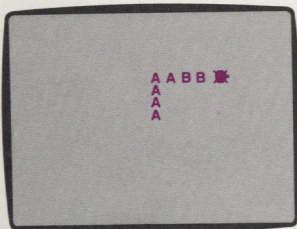
Try some of these challenges using loops:

1. Draw a square within a square.
2. Use loops to draw a triangle.
3. Use loops to draw a diamond.

(Suggested Solutions on page 74.)



Lesson 5: The Cast of Characters



Are you tired of watching the turtle draw balls? Tell it to switch. Type N, press RETURN, and type:

```
5 PA
10 DF3
20 TR
30 DF2
40 PB
50 DF2
```

Run the program. The turtle draws with A's, and then B's.

Statements 5 and 40 tell the turtle what type of character to print. The turtle prints the character following the P. In statement 5, that's an A, so the turtle will print A's when you give it Draw Forward (DF) statements. In line 10, the turtle draws three A's.

In line 30, the turtle draws two more A's. The turtle continues to print A's until you tell it to switch.

Line 40 tells the turtle to switch to printing B's. In line 50, the turtle does so.

The P# Statements

Here's another way to tell the turtle what to print. Change lines 5 and 40 by typing:

```
5 P#65
40 P#66
```


Run this program. It looks familiar, doesn't it? It's the same as the last program, even though the P statements look different. The key here is the # symbol after the P. This symbol means *number*. Now look at the chart on the next page. As you can see, different P# codes stand for different characters. Character #65, for example, is an A—and character #66 is a B. That explains why our new lines 5 and 40 have the same effect as the old ones.

GR0 and GR1

Run the program you've been working on in both GR0 and GR1. You'll see that characters in GR1 are wider than they are in GR0, and that fewer characters can fit across the screen.

A chart on page 32 details the differences between GR0 and GR1.

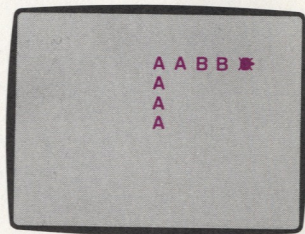
GR0 Characters

In GR0, you may use P# statements to tell the turtle to print any of 239 characters. The chart tells you which P# to use for each character.

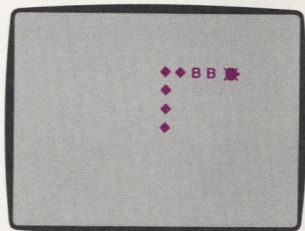
About half of these characters are also available through P statements. The chart tells you how to print those characters. For example, look at the \diamond , P#96. To print a \diamond , hold down the CTRL key and press the period (.) key. Change line 5 so that it tells the turtle to print \diamond 's. Type:

5 P \diamond

Make sure the turtle is in Graphics Zero (type GR0, if necessary). Then Run this program. You'll see the turtle draw \diamond 's instead of A's.



GR1



































































GR0

P # Codes

0	1	2	3	4	5	6	7
CTRL-,	CTRL-A	CTRL-B	CTRL-C	CTRL-D	CTRL-E	CTRL-F	CTRL-G
8	9	10	11	12	13	14	15
CTRL-H	CTRL-I	CTRL-J	CTRL-K	CTRL-L	CTRL-M	CTRL-N	CTRL-O
16	17	18	19	20	21	22	23
CTRL-P	CTRL-Q	CTRL-R	CTRL-S	CTRL-T	CTRL-U	CTRL-V	CTRL-W
24	25	26		32	33	34	35
CTRL-X	CTRL-Y	CTRL-Z		SPACE BAR	SHIFT-1	SHIFT-2	SHIFT-3
36	37	38	39	40	41	42	43
SHIFT-4	SHIFT-5	SHIFT-6	SHIFT-7	SHIFT-9	SHIFT-0	*	+
44	45	46	47	48	49	50	51
,	-	.	/	0	1	2	3
52	53	54	55	56	57	58	59
4	5	6	7	8	9	:	;

Purple characters available in GR1 and GR0.
 Black characters available in GR0 only.

60  <	61  =	62  >	63  SHIFT-/	64  SHIFT-8	65  A	66  B	67  C
68  D	69  E	70  F	71  G	72  H	73  I	74  J	75  K
76  L	77  M	78  N	79  O	80  P	81  Q	82  R	83  S
84  T	85  U	86  V	87  W	88  X	89  Y	90  Z	91  SHIFT-,
92  SHIFT+	93  SHIFT-	94  SHIFT>*	95  SHIFT-	96  CTRL-	97  a	98  b	99  c
100  d	101  e	102  f	103  g	104  h	105  i	106  j	107  k
108  l	109  m	110  n	111  o	112  p	113  q	114  r	115  s
116  t	117  u	118  v	119  w	120  x	121  y	122  z	123  ♦

124



128



129



130



131



132



133



134



135



136



137



138



139



140



141



142



143



144



145



146



147



148



149



150



151



152



153



154



160



161



162



163



164



165



166



167



168



169



170



171



172



173



174



175



176



177



178



179



180



181



182



183



184



185



186



187



188



189



190



191



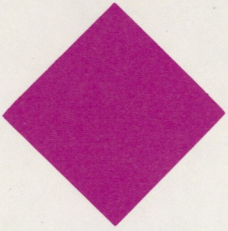
192



193



194	195	196	197	198	199	200	201
E	C	D	E	F	G	H	I
202	203	204	205	206	207	208	209
J	K	L	M	N	O	F	Q
210	211	212	213	214	215	216	217
R	S	T	U	V	W	X	Y
218	219	220	221	222	223	224	225
Z	E	\	J	^	...	◆	a
226	227	228	229	230	231	232	233
b	c	d	e	f	g	h	i
234	235	236	237	238	239	240	241
j	k	l	m	n	o	p	q
242	243	244	245	246	247	248	249
r	s	t	u	v	w	x	y
250	251	252					
z	♣	l					



Graphics Worlds Chart

	Graphic Zero	Graphics One
Screen Width	39 characters	20 characters
Screen Height	24 characters	24 characters
Character Capability	All chart characters	P#1-26; P#64-95
Color Capability	None	16 colors (Only four may be on the screen at one time.)

GR1 Characters

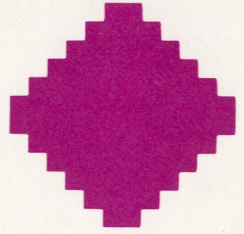
In GR1, the turtle can only print 58 different characters. These are characters #1-26, and characters #64-95. You may use either P or P# statements to tell the turtle to print any of these characters.

Type GR1 and press RETURN to enter Graphics One. Then run the program you've been working on (type R and press RETURN). You should get a screen message that tells you there's a P error in line 10. That's because the diamond, character #96, is not available in GR1. The turtle spotted this error in line 10, where it was told to Draw Forward.

Challenges

1. Use P statements to tell the turtle to print your name.
2. Use P# statements to tell the turtle to print your name.
3. Draw a rectangle made from lines and corners that connect. (This is a tough one!)
4. Use graphics character #160 to turn the entire screen white.
5. Use graphics characters 3, 5, 17, and 26 to draw a small square.

(Suggested Solutions on page 75.)



Lesson 6: Sound and Color!

The same P and P# statements you've been using can also tell the turtle to print characters in color. However, the turtle only prints in color when it is in Graphics One. Type GR1, and select a background color. For example:

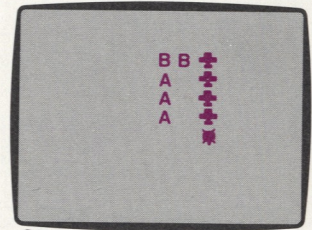
```
GR1;0
```

Here's a program that uses color. Type:

```
5 PA;3  
10 DF3  
20 TR  
25 PB  
30 DF2  
35 TR  
40 P#16;7  
50 DF4
```

Run this program. The turtle draws three red A's, turns right, draws two red B's, turns right, and then draws four blue crosses.

The color is introduced by the P and P# statements. Look at line 5. The first part of the statement looks familiar. It tells the turtle to print A's. The second part, the number 3, tells the turtle to print in color red. The same numbers you use in GR commands for background color serve to choose character color. The Color Chart is on page 19. As you can see, number 3 chooses color red.



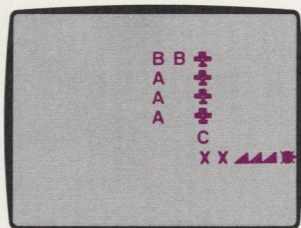
GR1

Once the turtle is given a color, it continues to use it until you tell it to change, or until you type N. That's why the turtle continues to use red when it is told to print B's in line 25. In line 40, the turtle is told to switch to color 7—blue—and to print P#16. So, in line 50, the turtle draws four blue crosses.

Four Colors

Only four different colors can be used on the screen at any one time. If you direct the turtle to use a fifth color, the first color you chose will switch to the new fifth color. To see this, add the following lines to the program:

```
60 PC;1
70 DF1
80 TL
90 PX;11
100 DF2
120 P#8;0
130 DF3
```



GR1

Run the program. Everything should run smoothly until the turtle draws character #8, a triangle. The turtle draws the triangle in color 0, gray. However, something strange happens to the three A's the turtle drew at the start of the program. They also turn to color gray!

Sometimes, characters will change color *before* the turtle uses up its limit of four colors. For example, type N, press RETURN, then type:


```
10 BL
20 PB;3
30 DF2
40 TR
50 RL4
60 JF5
70 PC;0
80 DF3
```

Run this program. Notice that two B's change color from red to gray, a color change that should happen only when the turtle uses more than four colors.

You might say the turtle is tricked into thinking that gray is the fifth color. How is it tricked? By the color number in line 20. The turtle thinks it is changing color each time it sees a color number—even if it is already using that color. The turtle sees line 20 four times, once each time it passes through the loop. In line 70, the turtle thinks it is switching to its fifth color instead of its second—so it changes the color of the first two B's it drew.

You can solve this problem by taking line 20 outside the loop. Type:

```
5 PB;3
20
```

The first color number is now outside the loop, so the turtle only comes across it once. When the turtle comes to line 70, it knows it is only using its second color. Run the program and see.

10

20

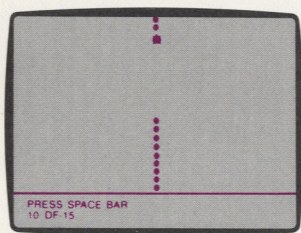
30

40

50

60

70



GR0

Walking in Color

When you Walk through a program, the bottom part of the screen is reserved for instructions. When the turtle crosses over this space, it won't draw. For example, type N, press RETURN, then type:

```
10 DF-15
```

Press RETURN, type W and press RETURN. Press the SPACE BAR. See? The turtle won't draw on the bottom part of the screen.

One more point about Walking: If you use a lot of colors in a program, the Walk screen sometimes flashes different colors. This is normal and will not affect the turtle.

Beeps

B statements bring sound to *Turtle Tracks*. Type N, press RETURN, and then type:

```
10 B
```

Run this instruction, and you'll hear the turtle beep. (Adjust the volume on your TV or monitor.)

Now give the turtle more exact instructions. Type:

```
10 B#121
```

Run this instruction, and the turtle should beep a C note. Why a C? Look at the B# chart on the next page. You'll see that different B# values correspond to different notes. B#121 is a C note in Octave 1.

Turtle Tracks

Musical Note Values

Octave 0

Note	B# Value
C	243
C#	230
D	217
D#	204
E	193
F	182
F#	173
G	162
G#	153
A	144
A#	136
B	128

Octave 1

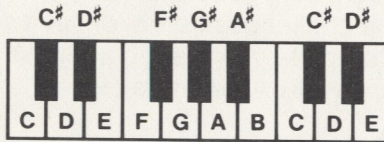
Note	B# Value
C	121
C#	114
D	108
D#	102
E	96
F	91
F#	85
G	81
G#	76
A	72
A#	68
B	64

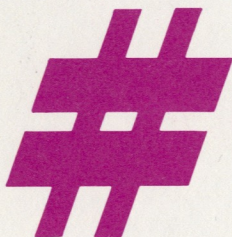
Octave 2

Note	B# Value
C	60
C#	57
D	53
D#	50
E	47
F	45
F#	42
G	40
G#	37
A	35
A#	33
B	31

Octave 3

C	29
---	----





The turtle can play notes in a range of more than three octaves—greater than most human voices. It will understand B# values between 0 and 255. As you can see on the chart, some of these B# values represent notes, and some fall between notes. The turtle plays them either way.

Duration

You can tell the turtle how long to hold notes. Type:

```
20 B#193;200
```

Run this instruction. You'll find that the turtle holds this note for a longer period of time than it held the last note. The number following the semicolon (;) controls the length of time a note is held. You may choose any number between 1 (shortest) and 999 (longest). If you don't give the turtle a number, it automatically chooses 25. In other words, the following two instructions mean exactly the same thing:

```
10 B#243;25    10 B#243
```

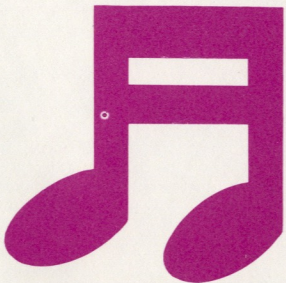
Music

Combine B# statements into melodies. Make up your own, or teach your favorite songs to the turtle.

If you haven't saved any of your programs on cassette, this might be a good time to start. Try saving your solution to one of the challenges below, or save any program of your choice. Complete information about saving and loading programs is in Appendix A, on page 59.

Challenges

1. Play the notes in a C scale in Octave 2: C D E F G A B C.
2. Play the notes in a G scale starting in Octave 1:
G A B C D E F# G.
3. Draw a ladder using four different colors. (Use P#4 and P#1.)
4. Draw a blue square. Use a different character for each side.
(Suggested Solutions on page 76.)



Lesson 7: Variables

Type N, press RETURN, and type the following short program:

```
5 X=3
10 DFX
```

Run the program. The turtle should Draw Forward three steps. Why would it do that? After all, there aren't any DF3 statements in this program.

Here's what happened. Line 5 told the turtle that X is 3. So, the turtle replaced the X in line 10 with a 3. Now type:

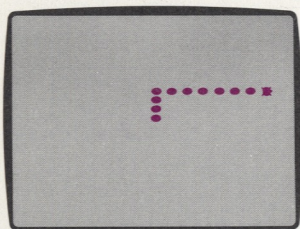
```
15 TR
20 X=7
30 DFX
```

Run the instructions. As before, the turtle Draws Forward three steps. Then it Turns Right and Draws Forward seven steps. Why? Because line 20 told the turtle that X is now 7. So the turtle translated line 30 to read DF7.

This program used DFX twice, and each time it meant something different. That's because the value of X was changed each time. Because you can vary—or change—the value of X throughout a program, X is called a *variable*.

You can use variables to create interesting shapes. Type N, press RETURN, and then type:

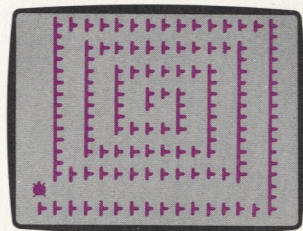
```
GR1;13
```



GR1

Press RETURN. You've just directed the turtle to use Graphics One with a yellow-green background. Now type this program:

```
5 X=1
10 BL
20 P#1;5
30 DFX
40 TR
50 X=X+1    [Tells the turtle to increase X by 1.]
60 RL16
```



GR1

The turtle should create a square spiral shape. Walk through the program. Notice that in line 5, before the loop begins, X is *initialized*. That means it is given an initial, or starting, value. In this case, line 5 said that X is 1.

Now look at line 50. It tells the turtle to increase X by 1 each time the turtle passes through the loop.

Try writing this program without variables, and you'll see just how helpful they are.

You may raise the value of variables by numbers other than 1. Here is a program that treats you to a sampler of the sounds the turtle can make. Type N, press RETURN, then type:

```
5 X=0
7 BL
10 B#X;100
20 X=X+10    [Tells the turtle to increase X by 10.]
30 RL25
```




A Second Variable

Turtle Tracks allows you to use two different variables in a program. Variables must be named X or Y. For example, type:

```
5 JF-9
7 TR
8 JF-9
9 TL
10 X=18
20 Y=0
30 BL
40 P#8;Y
50 DFX
60 TR
70 X=X-1
80 Y=Y+1
90 RL17
```

[The Y variable controls color.]

[The X variable controls the number of steps the turtle Draws Forward.]

Select a background color for GR1 and tell it to the turtle. Then run this program. Notice that you can decrease the value of a variable as well as increase it. The variable X begins at 18 and is lowered by 1 each time the turtle travels through the loop. Notice also how the colors change. You can never have more than four colors on the screen at the same time.

After a while, you'll get a screen message. The message will tell you that there's a color error in line 40. Why? Because Y has grown too large. When Y is 16, line 40 reads: P#8;16. The problem is that there is no color 16.

You may use variables instead of numbers wherever you'd like to in programs. You may use all of the following statements: DFX, DFY, JFX, JFY, RLX, RLY, and even GTX and GTY. But

line numbers themselves cannot be variables. The program at left, below, is okay, but the one at right is not:

Okay	Not Okay
5 X=30	5 GTX
10 GTX	10 JF5
20 JF5	X DF3
30 DF3	

You may change the value of a variable in a variety of ways. The following are some examples:

Y=5 X=-4 X=Y X=X+5

Y=Y+4 X=Y-9 Y=X+3

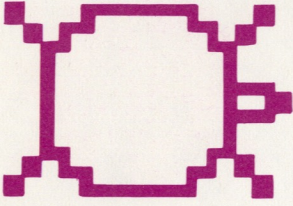
There are two things *you may not do* with variables:

- 1 You may not multiply or divide variables. (In fact, you can't use multiplication or division anywhere in *Turtle Tracks*.)
- 2 You can't use a variable after a plus or a minus sign. You can't say $X=5+X$. (The turtle ignores the "+X" part.) You can say, however, $X=X+5$.

Challenges

1. Use a variable to print the alphabet across the top row of your screen.
2. Use a variable to print TURTLE four times, each time in a different color.
3. Print 16 A's, each one in a different color. (Of course you'll never have more than four colors on the screen at one time.)

(Suggested Solutions on page 77.)



Section III: Full Speed Ahead

Lesson 8: Teaching the Turtle New Words

Miniprograms

You can tell the turtle to remember a group of instructions under a certain name. These groups of instructions are called *miniprograms*. Anytime you use the miniprogram name, the turtle carries out the entire group of instructions. For example, create a group of instructions to be remembered under the name SQUARE. Type:

TO SQUARE

Press RETURN, and the turtle responds:

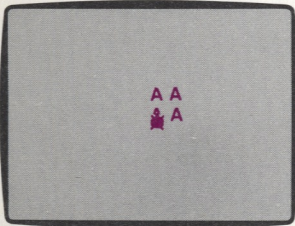
INVENTING SQUARE

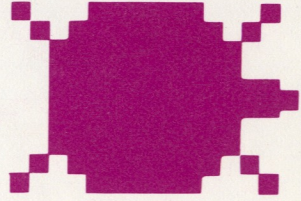
Type these instructions. Remember to use line numbers, as always:

```
5 PA;3
10 BL
20 DF1
30 TR
40 RL4
```

When you finish typing instructions for a miniprogram, see that the turtle does what you intended. Type R SQUARE and press RETURN.

GR1





Miniprogram Commands

You use the same types of commands with miniprograms that you use with main programs. You can Walk, List, or Run SQUARE by typing W SQUARE, L SQUARE, or R SQUARE. After W SQUARE or R SQUARE, the turtle thinks you are finished with SQUARE. If you then want to continue working on it, you must first type L SQUARE.

Erasing Miniprograms

You erase a miniprogram with the same command that creates it—TO followed by the miniprogram name. The TO command, like the N command, first erases instructions, and then prepares the turtle for new ones. For example, type:

```
TO DRAW
```

The computer says:

```
INVENTING DRAW
```

Type one line:

```
5 DF5
```

Now, type TO DRAW again and press RETURN. The computer says: DRAW ERASED. Actually, only the instructions for DRAW have been erased. The turtle expects new ones. To erase DRAW completely, you must now type any other command. For example, type D and press RETURN.

Directory

D stands for Directory, a list of the miniprograms you've taught the turtle. Only SQUARE is listed—you just erased DRAW.

Using Miniprograms as Instructions

The following short program uses SQUARE as an instruction. Type:

```
10 SQUARE
20 JF8
30 SQUARE
```

Run this program. Line 10 directs the turtle to carry out the instructions it has remembered under the name SQUARE. The turtle then goes on to line 20, where it Jumps Forward eight steps. In line 30, the turtle once again carries out instructions for SQUARE.

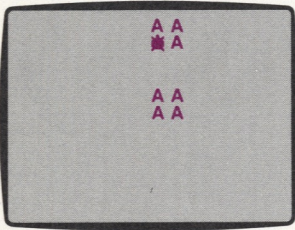
Variables in Miniprograms

SQUARE only draws squares made from red A's. If you use variables, SQUARE can make more *varied* squares. Type L SQUARE and press RETURN. Change line 10 of SQUARE by typing:

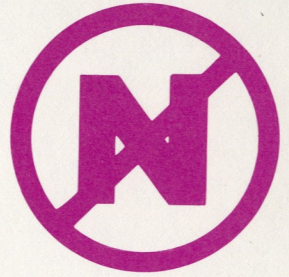
```
10P#X;Y
```

You just told the turtle that variable X determines the character used in SQUARE, and variable Y determines the color.

Now, you must tell the turtle what numbers to use for X and Y. You'll do so from the main program. Type D to go there.



GR1

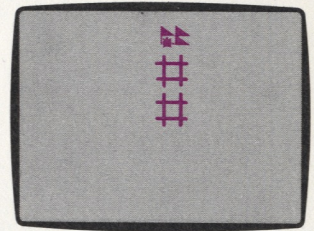


Changing the Main Program

You are now going to change the main program. But don't type N! That would erase miniprograms along with your instructions for the main program. Instead, we'll use the same line numbers over again. Type:

```
10 X=19
20 Y=6
30 SQUARE
40 JF3
50 SQUARE
60 X=10
70 Y=10
80 JF3
90 SQUARE
```

Run this program. It tells the turtle to carry out SQUARE three times. In lines 10 and 20, the main program gives the turtle values for X (the character) and Y (the color). The turtle uses those values when it carries out SQUARE the first time (line 30.) In line 50, the turtle is told to carry out SQUARE again. The main program has not changed X or Y, so the turtle draws the same square. In lines 60 and 70, the turtle receives new values for X and Y. So, in line 90, the turtle draws a different square.



GR1

Miniprograms Within Miniprograms

You may use a miniprogram to create new miniprograms. For example, type TO TRIPLE, and press RETURN. The computer screen shows: INVENTING TRIPLE

Now type the instructions of the program you just gave the turtle (10 X=19, 20 Y=6, etc.)

When you finish, check TRIPLE by typing R TRIPLE and pressing RETURN. If you want to make any changes, type L TRIPLE and then make them.

Now type D—and you'll see two words in the Directory, SQUARE and TRIPLE.

Rules for Using Miniprograms

The following four rules will help you understand the use of commands with miniprograms:

1. To change an existing miniprogram, first type L and the miniprogram's name. (If you're already working on the miniprogram, however, you can make the changes right away.)
2. To erase all instructions in a miniprogram, type TO and the miniprogram's name.

3. To completely erase a miniprogram, type TO and the miniprogram's name, and then type another command—D is a good choice.

4. To stop working on a miniprogram and return to the main program, type a command—L is a good choice.

If you forget to use a line number in a miniprogram, the turtle will surprise you and return to the main program. Remember to use line numbers!

Challenges

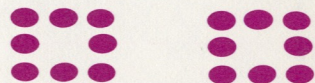
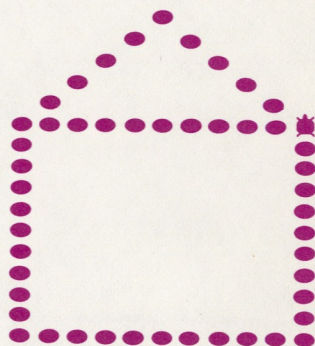
Use miniprograms to solve these challenges.

1. Draw a house:

2. Draw a face:

These examples use balls to keep things simple. Feel free, of course, to draw with any characters you choose.

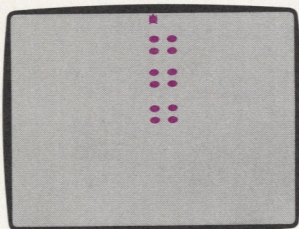
(Suggested Solutions on page 78.)



Lesson 9: Looping the Loops

Type N, press RETURN, and then type the following instructions:

```
1 P#20;3
2 BL ← [Tells the turtle to Begin Loop.]
5 BLA ← [Tells the turtle to Begin Loop A.]
10 DF1
20 TR
30 RLA4 [Tells the turtle to Repeat Loop A 4 times.]
40 JF3
50 RL3 ← [Tells the turtle to Repeat Loop 3 times.]
```



GR1

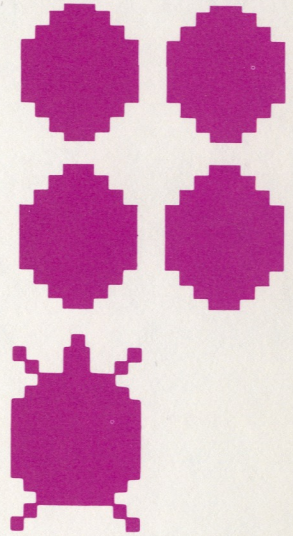
Run this program. The turtle should draw three squares made of red balls. Notice how similar this program is to the miniprogram TRIPLE in the last lesson. Instead of using miniprograms, however, it uses *nested loops*—loops that are arranged one inside another.

Look more closely at the program by walking through it. Type W, press RETURN, and control the turtle with the SPACE BAR. Notice that the turtle performs the inner loop, lines 5 through 30, four times before continuing to line 40. This inner loop draws a square. The instructions RLA and BLA are used for this loop.

The outer loop controls the distance between the squares (line 40) and the number of squares the turtle draws (line 50). Instructions RL and BL are used for this loop.

You may find it easier to understand this program if you see that the inner loop can be replaced with a miniprogram SQUARE. The program would then look like this:

```
TO SQUARE
10 BL      1 P#20;3
20 DF1     2 BL
30 TR      3 SQUARE
40 RL4     4 JF3
           5 RL3
```



Variables and Nested Loops

Here's a program that uses variables and nested loops together. Type N, press RETURN, and type:

```
5 BL ←
10 DF6
20 X=50
30 BLA ←
40 B#X;100
50 X=X+8
60 RLA5 —
70 TR
80 RL4 —
```


The turtle draws a square, and pauses to play five notes after drawing each side. You could also have typed:

```
TO TUNE
  5 BL          5 BL
 10 B#X;100    10 DF6
 20 X=X+8      20 X=50
 30 RL5        30 TUNE
                40 TR
                50 RL4
```

Here's another example of nested loops. This time, the outer loop uses the GT statement. That means this program will keep on running until you stop it. Type N, press RETURN, and type:

```
 2 Y=0
 5 X=1
10 BL ←
20 P#22;Y
30 DFX
40 TR
50 X=X+1
60 Y=Y+1
70 RL16 ←
80 Y=0
90 GT10 ←
```

Run the program. The inner loop, lines 10 to 70, directs the turtle to draw a spiral, changing color as it changes direction. In

line 80, Y changes back to 0, so that the turtle doesn't run out of colors. Line 90 sends the turtle back to line 10, setting up an infinite loop.

A Third Loop

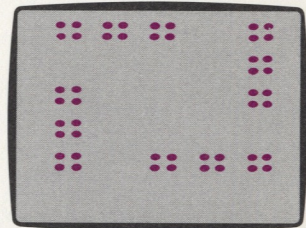
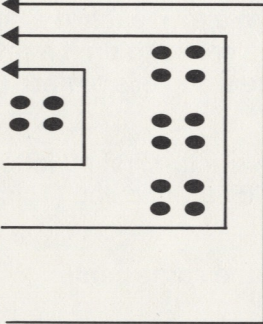
In *Turtle Tracks*, you may nest loops three deep. That means you may have a loop within a loop that's within another loop. Use instructions RLB and BLB for the third loop.

Here's an example:

```

10 JF-4
20 TR
30 JF-5
40 TL
50 P#20;3
60 BLB ←
70 BLA ←
80 BL ←
90 DF1
100 TR
110 RL4
120 JF3
130 RLA3
140 JF5
150 TR
160 RLB4

```



GR1

Run this program, and then walk through it. You may want to think about miniprograms as you do so. On page 48, we wrote a program that drew three squares. You could use that program here, in place of steps 70 through 130.

In *Turtle Tracks*, any loop may use instructions RLB and BLB, RL and BL, or RLA and BLA. The loop name ("A" or "B," for example) does not determine whether a loop is an inner loop or an outer loop. That's determined strictly by where you place your loops.

More than any other part of *Turtle Tracks*, you'll need lots of practice to get used to nesting loops. Expert programmers say it is one of the toughest skills to develop.

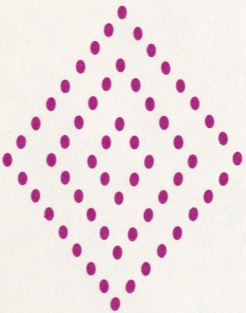
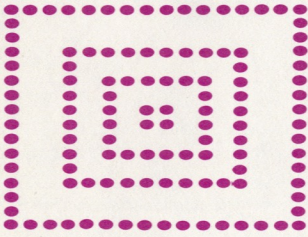
Make sure you understand the examples given in this chapter. Then, when you're ready, tackle some of these challenges.

Challenges

Use nested loops in the following challenges.

1. Have the turtle print four A's in a row, each in a different color. Then have it use the same colors to print a row of four B's underneath the A's. Continue using this pattern through the letter G.
2. Have the turtle draw four rows of the first 10 letters of the alphabet. Each row should be a different color. (This is very tricky.)
3. Draw four squares that have the same center.
4. Draw a diamond.
5. Draw two diamonds next to each other. (A tough challenge.)
6. Draw three diamonds, one inside the other. (A super challenger.)

(Suggested Solutions on page 79.)



Lesson 10: More About Miniprograms

If you haven't done so already, practice saving and loading some of your programs using your disk drive or cassette player. You need to know how to do this before you tackle this lesson. Instructions for loading and saving programs are in Appendix A, page 59.

Combining Programs

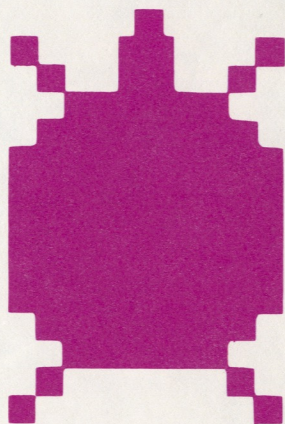
You may find it helpful to put together a library of miniprograms on cassettes or disks. You can then use these miniprograms to help you work on other programs.

Suppose, for example, that you are creating a program named FOREST. You remember a miniprogram OAK you once wrote, and you decide that OAK would be useful in FOREST. If you have instructions for OAK written down, you can add miniprogram OAK to FOREST by typing in OAK's instructions.

If you don't have the instructions written down, but you do have OAK saved on disk or cassette, you're still in luck. However, you shouldn't just load OAK directly into the computer. If you do, you'll erase FOREST. (The LOAD and DLOAD commands erase the turtle's memory.)

Here's what you can do instead:

1. Save FOREST on cassette or disk.
2. Load OAK (or the program OAK belongs to). FOREST is erased.
3. List OAK and copy down its instructions.



- 4 Load FOREST. OAK is erased.
- 5 Type instructions for OAK into the computer (first type TO OAK, and then type the instructions).

You may now continue working on FOREST. When you save it, OAK will be included.

Working with Miniprograms

Miniprograms are excellent building blocks for longer programs. They are easy to work with, easy to debug, and can often be used several times in the same program.

Let's take a close look at one program which involves several miniprograms — the sample program on your *Turtle Tracks* disk or cassette. Instructions for loading it from disk or cassette are in the following two sections.

Loading from a Disk

Place your *Turtle Tracks* disk in the disk drive if it's not already there. Type:

DLOAD DEMO

Press RETURN. The sample program loads in quickly. (If you type LOAD instead of DLOAD, the computer buzzes once, and expects a program to be loaded in by cassette. Press RETURN. You'll get a screen message in about a half minute. Then type DLOAD DEMO.

Loading from a Cassette

The sample program immediately follows *Turtle Tracks* itself on the cassette tape. If you didn't rewind the tape after you

loaded *Turtle Tracks*, the tape should be in the right position.

Type:

LOAD DEMO

Then follow normal cassette loading instructions.

If your *Turtle Tracks* cassette is rewound, you must advance the tape to the sample program's position. Your best strategy is to press the yellow SYSTEM RESET key, and then reload *Turtle Tracks*. The cassette should then be positioned for you to load in the sample program. Type:

LOAD DEMO

Then follow normal cassette loading instructions.

Running the Sample Program

Type GR1 and press RETURN, so that you may see the sample program in color. Then type R and press RETURN.

After the turtle runs through the sample program, type D and press RETURN. You'll see the names of the six miniprograms used in this program.

Pointing the Turtle

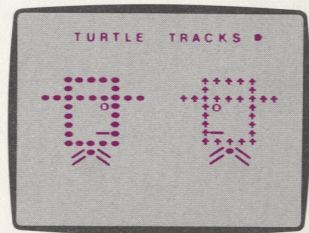
The turtle must be pointed in the right direction when it carries out instructions for a miniprogram. For example, type:

R BODY

Press RETURN. The turtle draws the same body it draws in the sample program. But now type:

R HAT

Press RETURN. HAT stands on end! The turtle wasn't pointed in the right direction.



The Main Program

The main program links together the six miniprograms. Take a quick look at the main program by typing:

L

Press RETURN. You'll notice that only three miniprograms are used as instructions — PERSON, NAME, and EYE. The other miniprograms — BODY, HAT, and FACE — are part of PERSON.

Erasing the Main Program

We want you to write your own programs using miniprograms from the sample program's directory. To do that, you first have to erase the sample's main program. Don't type N! That would erase the miniprograms as well as the main program. You must erase the main program line by line. (Or use the line numbers from the sample program in your own program. Of course, if your program is shorter, you still have to erase the extra lines.)

Challenges

You're now ready for the following *Turtle Tracks* toughies. A full listing of the sample program is on page 80.

1. Use miniprograms PERSON and EYE to construct one of the two figures from the sample program. Feel free to select your own characters and colors.
2. Erase miniprogram PERSON by typing TO PERSON and then typing D. Now, use miniprograms BODY, FACE, and HAT to try and reconstruct PERSON.
3. Use the miniprograms in different ways to create new figures, or entirely different shapes. Use different characters and colors, and feel free to change the miniprograms to suit your needs. Experiment!

Appendix A

Saving and Loading Your Work

When you type N and press RETURN, or when you turn off the computer, the program in the turtle's memory is erased. That means you must retype all the instructions if you later want to continue working on that program. That's no problem if the program is short. If it's long, however, you won't relish the thought of typing in all those instructions again.

Cheer up—there's a better way to do it. You can use the Atari cassette player or the disk drive to load your instructions into the computer in a fraction of the time it would take you to type them. To do that, of course, you must first save—tape—your programs before you erase them from the turtle's memory. In this Appendix, you'll find instructions for using disk drives, cassette players, and even printers.

If you make a typing error when saving or loading programs, you may lose control of your computer for a while. Be patient. Eventually, a screen message will tell you there's an error, and then you'll regain control of your computer. (The error number that appears on the screen may be helpful. Consult the Atari manuals for disk drives or cassette players for further explanation.)

Cassette Instructions

SAVING A PROGRAM

- 1** Place a cassette tape into the cassette player. Don't use the *Turtle Tracks* tape, or you might erase it! If you are saving your first program, wind the tape to the beginning. If you've already saved other programs on the same tape, make sure you know where each starts and ends. A simple way to keep track is to use the counter on the cassette player.
- 2** Type SAVE followed by the name of your program. For example,
SAVE BOX
Then press RETURN.
- 3** The computer will buzz twice. Press down the PLAY and RECORD buttons on your tape machine at the same time. Do this smoothly and carefully, using one finger from each hand. Then, press the RETURN key on the computer keyboard.
- 4** The computer makes a high-pitched sound while your program is being saved. Soon, you'll see on the screen that your program is saved. For example, BOX SAVED.
- 5** Press STOP on the cassette player.
- 6** You should now check to see that your program was properly saved. Rewind the cassette to the start of the program. Type VERIFY followed by the program name. For example, VERIFY BOX. Then press RETURN.

- 7** The computer buzzes once. Press the PLAY button only on the cassette player. *Don't press RECORD!* Then press RETURN on the computer keyboard. Wait a while. The screen will tell you whether the program is okay (FILE LOADED or VERIFIED) or not okay (VERIFY ERROR). If it's not okay, you'll have to save your program all over again. Sorry about that.
- 8** Press STOP on the cassette player.

LOADING A PROGRAM

Once you have saved a program, you may load it back into *Turtle Tracks* anytime *after* you have loaded *Turtle Tracks* itself.

- 1** Type LOAD followed by the name of your program. For example, LOAD BOX
- 2** The computer buzzes once. Press PLAY on the cassette player, and press RETURN on the computer keyboard.
- 3** Wait until you get a screen message that says FILE LOADED or VERIFIED.
- 4** Press STOP on the cassette player.

You're now ready to work with your program.

Disk Instructions

FORMATTING A STORAGE DISK

Before you can save a program on a disk, you must first prepare the disk. This is known as *formatting*. You must format disks *before* you enter *Turtle Tracks*. Here's how:

- 1** Turn on your disk drive and TV or monitor. (Don't turn the computer on yet. If it's already on, turn it off.)
- 2** Place your ATARI system disk into the disk drive. (Any other disk equipped with the DOS files will also work.)
- 3** Turn on your computer.
- 4** Type:
DOS
When the DOS menu appears, type the letter I , then press RETURN.
- 5** The computer will ask you which drive to format. Type 1 and press RETURN. (If you are using more than one disk drive, you will have to type the proper number.) Type Y and press RETURN when the computer tells you to do so.
- 6** The disk is now being formatted. It will click and whir for about half a minute.

- 7** When the disk is formatted, the drive stops whirring and you'll see **SELECT ITEM** or **RETURN FOR MENU**. Remove the disk and label it with a felt-tipped pen. Pencil or ballpoint pen may ruin the disk.
- 8** To enter *Turtle Tracks*, turn the computer off, place the *Turtle Tracks* disk in the drive, and then turn the computer on.

SAVING A PROGRAM

- 1** Take the *Turtle Tracks* disk out of the disk drive and place it in its protective envelope.
- 2** Place your formatted program storage disk into the disk drive. Close the disk drive door.
- 3** Type **DSAVE** followed by the program title. For example, **DSAVE SQUARE**. Press **RETURN**.
- 4** If you give your program a name that you've already used, the computer will say **FILE EXISTS**. You then have two choices. Change the name of the program you're now saving, or erase the old program that has the same name. You may erase any program by typing **ERASE**, followed by the program name.
- 5** The screen will soon show that your program was saved. For example, **SQUARE SAVED**.

- 6 Check to make sure your program was properly saved. Type DVERIFY followed by the program name. For example, DVERIFY SQUARE. Press RETURN. The computer will soon say if the program was properly recorded (FILE LOADED or VERIFIED) or if there was an error (VERIFY ERROR). If you used a name that doesn't exist, the screen will show FILE NAME ERROR or FILE NOT FOUND. If your program was not saved properly, save it again.

LOADING A PROGRAM

Load a program from a disk only *after* you are in *Turtle Tracks*.

- 1 Take out the *Turtle Tracks* disk and place it in its protective envelope.
- 2 Insert your program storage disk into the disk drive and close the door.
- 3 Type DLOAD followed by the program title. For example, DLOAD BOX. Press RETURN.
- 4 The program will quickly be loaded into the turtle's memory. You may then work on it.

Catalog of Programs

If you type C and press RETURN, you'll get a catalog of all the programs you've saved on your program storage disk. The number after the program name tells you how many *sectors* the program takes up on on the disk. The number on the bottom tells you how many more sectors are *free*, or available. As you can see, you could easily store 50 programs or more on your disk.

Printer

If you are using an ATARI printer, you may print out the screen display. To do so, press the yellow OPTION key after you Run or Walk a program. If you press any other key, during or following the printing process, you will be returned to the blackboard.

L PAPER

You may print a Listing of a program if you type L PAPER and press RETURN. You may not be able to print screen displays or lists with printers that are not made by Atari.

Appendix B

A Glossary of Turtle Tracks Instructions

Statements

(Statements require line numbers, though none are shown here.)

DF5	Draw Forward 5 steps. (Lesson 1)
DF-4	Draw backward 4 steps (Lesson 2)
JF3	Jump Forward 3 steps. (Lesson 2)
JF-2	Jump backward 2 steps. (Lesson 2)
TR	Turn Right. (Lesson 1)
TL	Turn Left. (Lesson 1)
GT50	Go To line 50. (Lesson 4)
X=7	X is 7. (Lesson 7)
Y=Y+1	Raise the value of Y by 1. (Lesson 7)
PA	Draw using A's. (Lesson 5)
P#16	Draw using the character #16 (+). (Lesson 5)
PB;3	Draw using B's in red. (Lesson 6)
B	Beep. (Lesson 6)
B#130;200	Beep at pitch #130. Hold for duration 200. (Lesson 6)
'Reminder	Message that the turtle skips. (Lesson 3)
BL, BLA,	Begin Loop and Repeat Loop instructions.
BLB, RL3,	(Lessons 4 and 9)
RLA4, RLB8	

Commands

(Commands don't use line numbers.)

L List entire program, or series of instructions.
(Lesson 1)

R Run the main program. (Lesson 1)

W Walk through the main program.
(Lesson 4)

L, W, and R May also be used with miniprograms.
(Lesson 8)

D Directory of miniprograms. (Lesson 8)

N New program. Erases old program and all
miniprograms. (Lesson 1)

TO WORD Readies computer to receive instructions for
a miniprogram (here called WORD as an
example). Erases any old miniprogram by
that name. (Lesson 8)

GR0 and GR1 Selects graphics world. (Lesson 3)

GR0;5 Selects graphics world zero, background
color violet. (Lesson 3)

Special Keys

CLEAR Clears the screen (SHIFT must be held
down). (Lesson 4)

START Starts program that's been stopped in prog-
ress. (Lesson 4)

CTRL Makes certain characters available from key-
board. (Lesson 5)

SHIFT Makes certain characters available from key-
board. (Lesson 5)

Appendix C

Some Common Mistakes

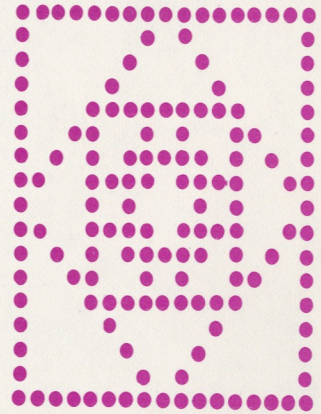
- 5 TR3** You may not tell the turtle to turn right three times. Do not follow TR or TL with a number.
- DF3** You forgot to use a line number with statement DF3. All statements require line numbers.
- 5 R** Do not use line numbers with commands R, N, W, L, D, SAVE, LOAD, or TO (as in TO SQUARE).
- 10 BL4** Do not follow BL (Begin Loop) with a number. The number should be used with the RL (Repeat Loop) statements.
- 5 DF2.5** You may only use decimals as line numbers—not in JF or DF statements.
- 10 P90** If you want the turtle to draw character number 90, type: 10 P#90.
- 5 P ♥ ◇** The turtle will only print one character at a time. When you want it to switch, use a new P statement.

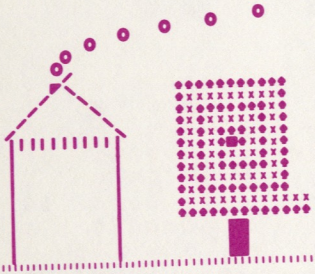
Remember: There are different laws for GR1 and GR0! Confusing the two is a common cause of errors.

Appendix D

A Short Collection of Interesting Programs

1		2
TO DRAW	1 JF1	1 JF9
10 BLA	2 TL	2 TL
20 BL	3 JF2	3 JF18
30 DF1	4 TR	4 TR
40 TR	5 X=2	5 Y=1
50 JF1	6 DRAW	6 X=65
60 TL	7 X=X+2	7 BLA
70 RLX	8 DRAW	8 P#X
80 TR	9 X=X+4	9 BL
90 RLA4	10 DRAW	10 DF1
100 BL		11 TR
110 DFX		12 JFY
120 TR		13 TL
130 DFX		14 RL2
140 RL4		15 TR
150 TL		16 RLA4
160 JFX		17 X=X+1
170 TR		18 Y=Y+1
		19 GT7





3

TO SQUARE	10 'House	210 TR	400 DFX
1 BL	20 JF-11	220 X=10	410 TL
2 DFX	30 X=10	230 P#160	420 RL10
3 TR	40 P#124	240 Y=1	430 TR
4 RL4	50 SQUARE	250 DFX	440 JF-10
	60 JF11	260 TR	450 TR
TO SLANT	70 X=5	270 DF1	460 P#34
1 BL	80 P#6	280 TR	470 DF58
2 DF1	90 SLANT	290 DFX	480 TL
3 TR	100 TR	300 P#123	490 JF17
4 JF1	110 P#7	310 X=12	500 Po
5 TL	120 X=1	320 SPIRAL	510 X=1
6 RLX	130 SLANT	330 DF1	520 BL
	140 P#138	340 TR	530 DF1
TO SPIRAL	150 X=1	350 PX	540 TL
10 BL	160 SLANT	360 DF1	550 JFx
20 DFY	170 X=4	370 X=X-1	560 TR
30 TR	180 P#7	380 BL	570 X=X+1
40 Y=Y+1	190 SLANT	390 X=X-1	580 RL6
50 RLX	200 JF-23		

Challenge Solutions

Lesson 1: Suggested Solutions

1	2		3	4
1 DF5	1 DF4	(Notice that this program uses DF4 statements to build a square of five balls to a side. The fifth ball is drawn when the turtle begins drawing the next side.)	1 DF2	Capital L
2 TR	2 TR		2 TR	1 TL
3 DF5	3 DF4		3 DF6	2 DF4
4 TR	4 TR		4 TR	3 TR
5 DF5	5 DF4		5 DF2	4 DF7
6 TR	6 TR		6 TR	Capital T
7 DF5	7 DF4		7 DF6	1 DF6
			2 TR	
			3 DF3	
			4 TL	
			5 TL	
			6 DF7	

Lesson 2: Suggested Solutions

1	2	3	4	5
1 JF-12	1 JF11	1 TR	1 DF4	1 DF1
2 DF3	2 TL	2 DF1	2 TR	2 TR
3 TR	3 JF-18	3 TL	3 DF1	3 DF1
4 DF3	4 DF2	4 JF1	4 TR	4 TR
5 TR	5 TL	5 TR	5 JF1	5 DF1
6 DF3	6 DF2	6 DF1	6 TL	6 TR
7 TR	7 TL	7 TL	7 DF1	7 DF2
8 DF3	8 DF2	8 JF1	8 TR	8 JF2
	9 TL	9 TR	9 JF1	9 TL
	10 DF2	10 DF1	10 TL	10 JF1
	11 JF-23	11 TL	11 DF1	11 DF1
	12 TR	12 JF1	12 TR	12 TL
	13 JF-36	13 TR	13 JF1	13 DF7
	14 DF2	14 DF1	14 TL	14 TL
	15 TL	15 TL	15 DF1	15 DF2
	16 DF2	16 JF1	16 TL	16 JF3
	17 TL		17 JF-1	17 DF2
	18 DF2		18 DF5	18 TL
	19 TL		19 JF4	19 DF2
	20 DF2			20 TL
				21 DF2
				22 TL
				23 DF2
				24 JF-7
				25 DF2
				26 TL
				27 DF2
				28 TL
				29 DF2
				30 TL
				31 DF2
				32 JF-8

Lesson 3: Suggested Solutions

1	2	3
1 JF-12	1 JF-12	1 JF-12
2 TR	2 TR	2 TR
3 JF-20	3 JF-10	3 JF-10
4 DF38	4 DF19	4 DF19
5 TL	5 TL	5 TL
6 DF23	6 DF19	6 DF23
7 TL	7 TL	7 TL
8 DF38	8 DF19	8 DF19
9 TL	9 TL	9 TL
10 DF23	10 DF19	10 DF23

Lesson 4: Suggested Solutions

1	2	3	
1 BL	1 BL	1 BL	15 BL
2 DF3	2 DF1	2 DF1	16 DF1
3 TR	3 TR	3 TR	17 TR
4 RL4	4 JF1	4 JF1	18 JF1
5 JF6	5 TL	5 TL	19 TL
6 TR	6 RL3	6 RL3	20 RL3
7 JF-3	7 TR	7 TR	21 TR
8 BL	8 BL	8 BL	22 BL
9 DF9	9 DF1	9 DF1	23 DF1
10 TR	10 TR	10 TR	24 TR
11 RL4	11 JF1	11 JF1	25 JF1
	12 TL	12 TL	26 TL
	13 RL3	13 RL3	27 RL3
	14 TR	14 TR	
	15 TR		
	16 DF7		

Lesson 5: Suggested Solutions

1	2	3	4	5
Here's how you	1 TR	GR0 only	GR0 only	1 P#17
would tell the turtle	2 P#84	5 P#26	1 JF11	2 TR
to write	3 DF1	10 DF1	2 TR	3 DF1
TOM SMITH:	4 P#79	15 P#124	3 P#160	4 TR
1 TR	5 DF1	20 DF5	4 BL	5 P#5
2 PT	6 P#77	25 P#17	5 DF39	6 DF1
3 DF1	7 DF1	30 TR	6 TR	7 TR
4 PO	8 JF1	35 DF1	7 JF1	8 P#3
5 DF1	9 P#83	40 P#18	8 TL	9 DF1
6 PM	10 DF1	45 DF5	9 RL23	10 P#26
7 DF1	11 P#77	50 P#5		11 DF1
8 JF1	12 DF1	55 TR		
9 PS	13 P#73	60 DF1		
10 DF1	14 DF1	65 P#124		
11 PM	15 P#84	70 DF5		
12 DF1	16 DF1	75 P#3		
13 PI	17 P#72	80 TR		
14 DF1	18 DF1	85 DF1		
15 PT		90 P#18		
16 DF1		95 DF5		
17 PH		100 JF10		
18 DF1				

Lesson 6: Suggested Solutions

1	2	3	4
1 B#60	1 B#81	GR1 only	GR1 only
2 B#53	2 B#72	10 P#4;5	10 P#8;7
3 B#47	3 B#64	20 DF5	20 DF5
4 B#45	4 B#60	30 P#4;7	30 TR
5 B#40	5 B#53	40 DF5	40 P#10
6 B#35	6 B#47	50 TL	50 DF5
7 B#31	7 B#42	60 JF1	60 TR
8 B#29	8 B#40	70 TL	70 P#16
		80 JF1	80 DF5
		90 P#1;10	90 TR
		100 DF5	100 P#19
		110 P#1;1	110 DF5
		120 DF5	

Lesson 7: Suggested Solutions

1	2	3
GR0	GR1 only	GR1 only
1 JF11	5 JF4	5 TR
2 TR	10 TR	10 JF-8
3 JF-20	20 JF-3	20 $X=0$
4 $X=65$	30 $X=0$	30 BL
5 BL	40 BL	40 PA;X
6 P#X	50 PT;X	50 DF1
7 DF1	60 DF1	60 $X=X+1$
8 $X=X+1$	70 PU	70 RL16
9 RL26	80 DF1	
	90 PR	
	100 DF1	
	110 PT	
	120 DF1	
	130 PL	
	140 DF1	
	150 PE	
	160 DF1	
	170 JF-6	
	180 TR	
	190 JF3	
	200 TL	
	210 $X=X+1$	
	220 RL4	

Lesson 8: Suggested Solutions

1

TO SLANT 1 TR
5 BL 2 JF-5
10 DF1 3 TL
20 TR 5 SLANT
30 JF1 10 TR
40 TL 20 SLANT
50 RL5 30 TR
 40 SQUARE
TO SQUARE
5 BL
10 DF10
20 TR
30 RL4

2

TO SMILE 4 X=2
5 DF1 5 SQUARE
15 TL 10 TL
20 DF7 20 JF3
25 TL 30 SQUARE
30 DF2 40 JF-1
 50 TR
TO SQUARE 60 JF-3
5 BL 70 X=1
10 DFX 80 SQUARE
20 TR 90 TL
30 RL4 100 JF3
 110 TL
 120 JF1
 125 SMILE
 140 JF-4

Lesson 9: Suggested Solutions

1	2	3	4	5	6
GR1 only	GR1 only	10 X=1	1 BL	10 BLB	5 X=2
1 TR	5 Y=3	20 BLB	2 BLA	20 BLA	10 TL
2 X=65	10 TR	30 BL	3 DF1	30 BL	20 JF2
4 BL	15 JF-5	40 DFX	4 TR	40 DF1	30 TR
10 Y=0	20 BLA	50 TR	5 JF1	50 TR	40 BLB
20 BLA	25 X=65	60 RL4	6 TL	60 JF1	50 BLA
30 P#X;Y	30 P#65;Y	70 JF-2	7 RLA3	70 TL	60 BL
40 DF1	35 BL	80 TL	8 TR	80 RL3	70 DF1
50 Y=Y+1	40 P#X	90 JF2	9 RL4	90 TR	80 TR
60 RLA4	45 DF1	100 TR		100 RLA4	90 JF1
70 JF-4	50 X=X+1	110 X=X+4		110 TL	100 TL
80 TR	55 RL10	120 RLB4		120 JF8	110 RLX
90 JF1	60 JF-10			130 TR	120 TR
100 TL	65 TR			140 RLB2	130 RLA4
110 X=X+1	70 JF2				140 TL
120 RL7	75 TL				150 JF3
	80 Y=Y+2				160 TR
	85 RLA4				170 X=X+3
					180 RLB3

Lesson 10: Sample Program Listing

TO EYE	TO HAT	TO FACE	TO BODY
10 PO	10 DF9	10 BL	10 P#6;7
20 DF1	20 JF-3	20 TL	20 DF1
30 JF1	30 BL	30 DF4	30 TR
40 P#18	40 TR	40 RL4	40 DF1
50 DF1	50 DF2		50 P#20
	60 TR		60 DF1
	70 DF4		70 TR
	80 RL2		80 P#7
			90 DF2
TO PERSON	TO NAME		
10 HAT	10 PT	10 JF2	
20 FACE	20 DF1	20 TL	
30 JF-1	30 PU	30 JF2	
40 TR	40 DF1	40 PERSON	
50 JF-6	50 PR	50 EYE	
60 BODY	60 DF1	60 JF-4	
70 JF-6	70 PT	70 TR	
	80 DF1	80 JF-14	
	90 PL	90 P#16;3	
	100 DF1	100 PERSON	
	110 PE	110 TR	
	120 DF1	120 JF2	
	130 JF1	130 TL	
	140 PT	140 EYE	
	150 DF1	150 JF-10	
	160 PR	160 TL	
	170 DF1	170 JF-11	
	180 PA	180 PT;5	
	190 DF1	190 NAME	
	200 PC		
	210 DF1		
	220 PK		
	230 DF1		
	240 PS		
	250 DF1		

Have a question?

Call the WizlineSM number and ask a Wizkid what to do.

(201) 567-8512

ISBN 0-590-95602-7



Scholastic

Wizline is a service mark of Scholastic Inc.