

CLAIRE BAILEY PASSANTINO

ITTY BITTY BYTES OF SPACE

FOR THE
TI-99/4A[®]
COMPUTER



A CREATIVE PASTIMES BOOK

Itty Bitty Bytes of Space for the TI-99/4A® Computer

Claire Bailey Passantino

Text Illustrations by Nancy Gurganus



**A Reston Computer Group Book
Reston Publishing Company, Inc.
A Prentice-Hall Company
Reston, Virginia**

A CREATIVE PASTIMES BOOK

Library of Congress Cataloging in Publication Data

Passantino, Claire Bailey.

Itty bitty bytes of space for the TI-99/4A computer.

(A Creative pastimes book)

"A Reston Computer Group book."

Summary: Twenty programs designed to reinforce beginning computer concepts and skills, including games, contests, races, songs, riddles, and other activities.

1. TI 99/4A (Computer)—Programming—Juvenile literature. 2. Basic (Computer program language)—Juvenile literature. 3. Computer programs—Juvenile literature. 4. Computer games—Juvenile literature. [1. TI 99/4A (Computer)—Programming. 2. Basic (Computer program language) 3. Computer programs. 4. Computer games. 5. Programming (Computers) I. Title. II. Series.

QA76.8.T133P37 1984

001.64'2

83-24579

ISBN 0-8359-3318-0

TI-99/4A® is a registered trademark of Texas Instruments.

Cover illustration by Bethann Thornburgh

Cover design by Nancy Sutherland

© 1984 by Claire Bailey Passantino

All rights reserved. No part of this book may be reproduced in any way, or by any means, without permission in writing from the author and the publisher.

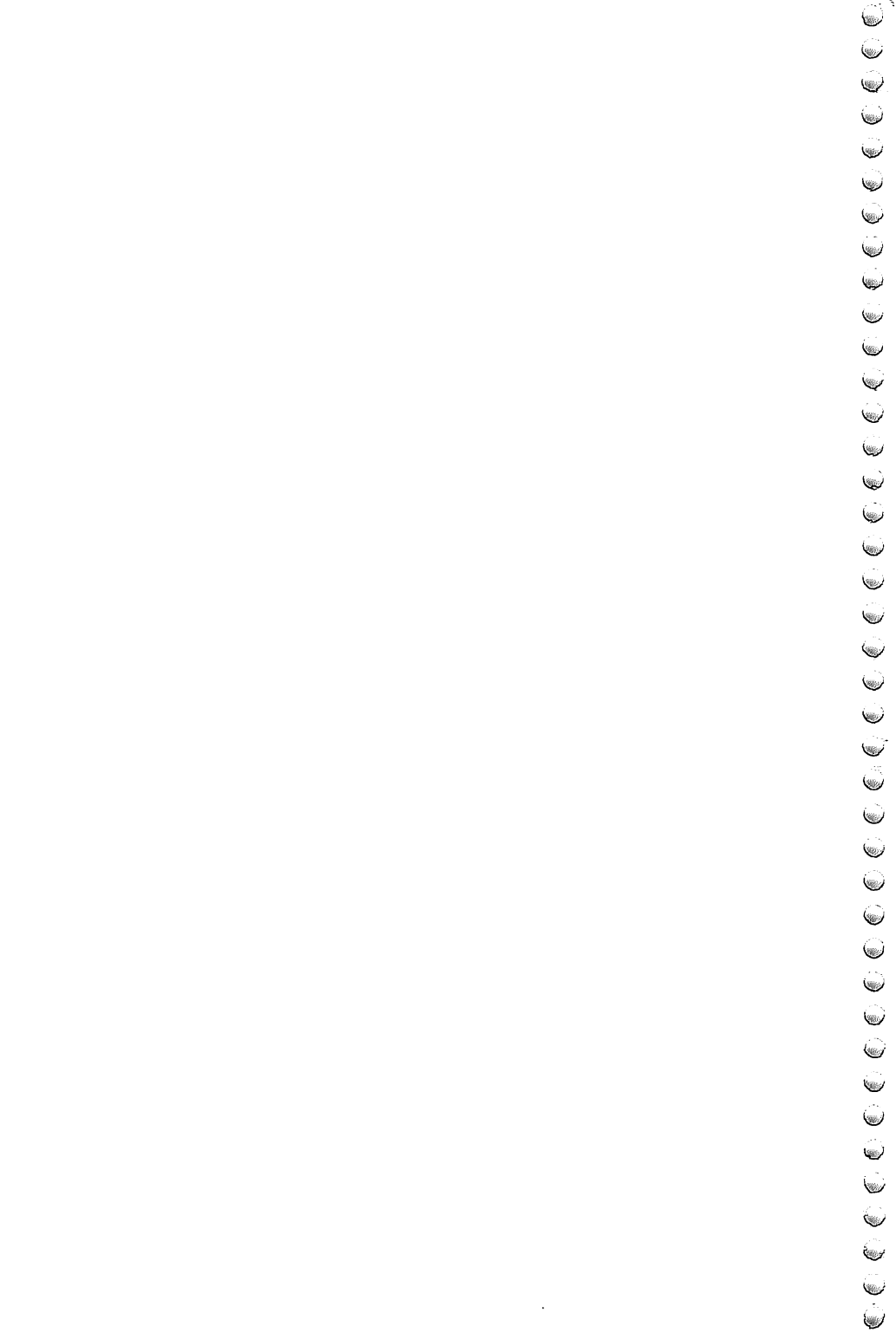
10 9 8 7 6 5 4 3 2 1

Printed in the United States of America

Other Itty Bitty Bytes Books:

**Matilda, the Computer Cat
School Days**

**These titles are also available for
the ATARI® Computer and
the Commodore 64® Computer.**



Contents

Foreword, vii

A Note to Parents and Teachers, ix

Passing the Test, 2–3

Gravity, 4–5

Test Your Spacecraft, 6–7

Take Off, 8–9

Count the Stars, 10–11

Music from Mother Earth, 12–13

Make Your Own Kind of Music, 14–15

What Is It?, 16–17

Who Is It?, 18–19

Hit or Miss?, 20–21

Nighty Night!, 22–23

Good Morning!, 24–25

Meteors, 26–27

How Much Fuel?, 28–29

Space Sketch, 30–31

Design Your Parachute, 32–33

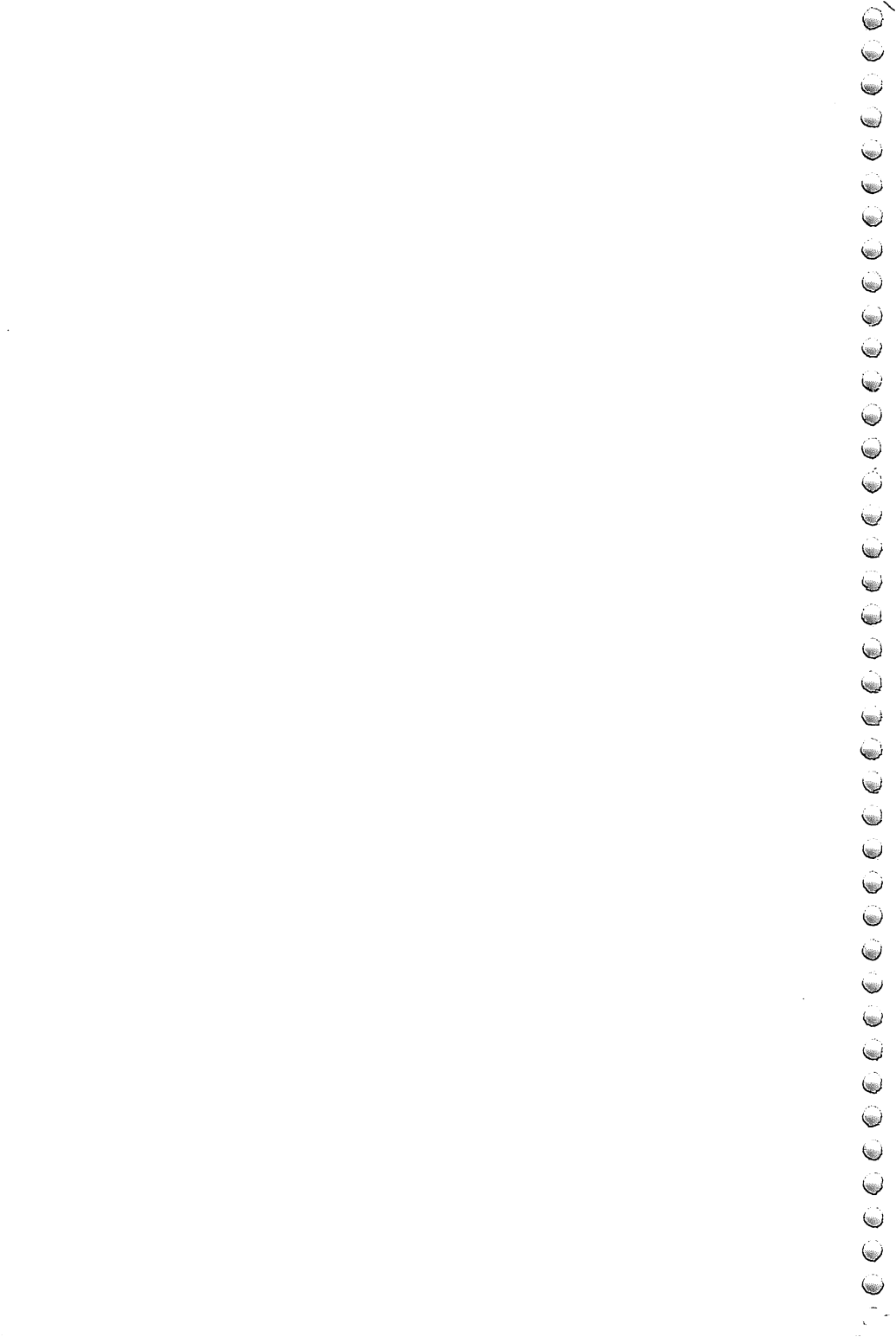
Happy Landing!, 34–35

Final Report, 36–37

The Race for Space, 38–39

Break the Code, 40–41

Appendix, 43



Foreword

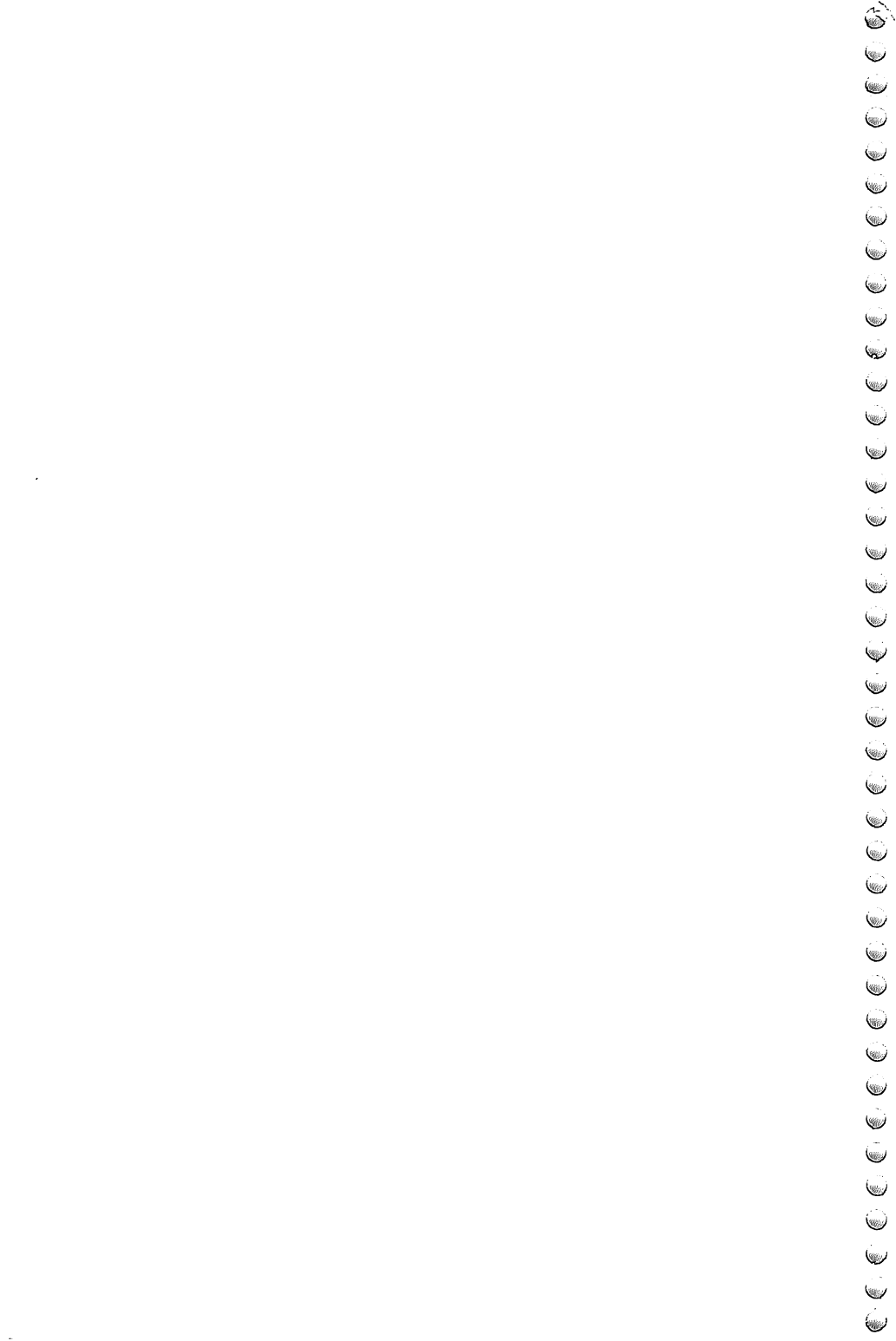
This is the first of the Itty Bitty Bytes books for the TI-99/4A®. Each book in the series is designed to bring you twenty fun programs and, at the same time, teach you something about BASIC. Typing in the programs is hard work. You must be careful to number the lines exactly, spell the words correctly, and put in all the right punctuation. But you will feel extra good when your program runs smoothly!

Always read the explanations next to each program. Sometimes there are additions or changes that you can make to improve the program. Once you see what the program is all about, feel free to experiment with it—and by all means combine programs that work well together. The more you can do to make a program reflect your own special personality, the more you will enjoy it. Let me know if you find some nifty changes. I love to get mail!

The Itty Bitty Bytes books have grown out of the teaching experiences I've had with my computer students. Special thanks go to all my "COMPU-KIDS"—and to my own kids, too—who keep presenting me with one good idea after another. They are my severest critics, but my most outstanding assets.

Happy computing!

Claire Bailey Passantino



A Note to Parents and Teachers

You bought the computer. You read the manuals. You did the demo programs. You and/or your children may even have taken some computer classes. But now the computer is just sitting there. Everyone was so enthusiastic. *What happened?*

The novelty of a new computer will take you just so far. Beyond that point, a continuing interest in this incredible tool is directly related to its usefulness. "Not useful" equates to "not used." So the problem becomes, what can computers do that children would find useful? What kinds of things would encourage children to expend the energy needed to create their own computer programs?

Each Itty Bitty Bytes book in the Creative Pastimes series is packed with computer activities appealing to young programmers. Simple games, contests, races, pictures, designs, songs, riddles, charts, tests, and more—all are designed to be fun while reinforcing beginning computer concepts and skills.

Besides providing fun, there are fringe benefits to having children write their own programs. In working with children, I have found that computer programming encourages them to:

- Think creatively.
- Use logical thinking skills.
- Attend to details.
- Take small steps to achieve a goal.
- Personalize programs.
- Develop pride and self-esteem.
- Appreciate packaged software.

To help children enjoy doing their own programs, here are some suggested DOs and DON'Ts:

- DO encourage children to type in their own programs. With younger children, bargain: "You type this line and I'll type the next one." (Save the long lines for yourself!)
- DON'T criticize typing expertise. Speed and correct fingering are typing skills that are minimally related to computer programming.
- DO allow children to make mistakes.
- DO help them find the errors they've made. (This is called "debugging" the program.)
- DON'T worry when you get an error message. This means you've made a mistake. Check your program for "bugs."
- DO encourage children to read and understand the program explanations.

-
- DON'T, however, force the issue. Some people learn by reading. Others learn by doing. As skills are repeated over and over in different contexts, children may just "catch on."
 - DO be aware of some common pitfalls. Remember to:
 1. Use line numbers.
 2. Press ENTER after you type in a line.
 3. Give great attention to spelling and punctuation. Quotation marks, commas, semicolons, colons, and even spaces are often important.
 4. Save your program before you turn off the computer.
 - DO help children save their programs on tape or disk so they can use them again and again. If you have a printer, use it to make "hard" copies of each program. People like to see themselves in print.
 - DON'T type the "FCTN" and the plus (+) keys simultaneously. This combination will erase the program from memory and you will have to start over.
 - DO remember that typing the "FCTN" and the "4" simultaneously will break the program and get you out of an infinite loop.
 - Do type in and run all programs with the ALPHA LOCK on (that is, in the *down* position). The only exceptions are joystick programs, which are run with the ALPHA LOCK off.
 - DO praise children for a job well done. And enjoy the programs that they've created.
 - DO modify and use programs that you yourself find useful.

It is my sincere hope that the Itty Bitty Bytes books will help you and your children establish a healthy working relationship with your computer. Take that computer off the shelf! And let me know how things are working out!



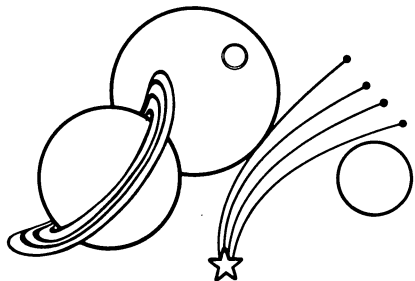
Itty Bitty Bytes of Space for the TI-99/4A® Computer

Main ideas: A test using simple data statements
Scoring the test

C = Number correct
T = Total number of questions
PL\$ = Name of planet
F\$ = Fact about planet
A\$ = Answer given
SC = Score

10–20 Sets these variables to 0.
30–40 Pauses.
50 Clears the screen.
60 Counts over 9 spaces and prints the title.
70–90 Prints 10 blank lines (positions the title).
100 Reads the name of the planet and the fact about it (see lines 300 to 390).
110 Checks for the end of the data (see line 400).
120 Prints the fact.
130–140 Your chance to guess the answer.
150 The total number of questions goes up one.
160 If the answer is right, the computer skips to line 190.
170–180 Prints the correct answer (if you were wrong) and returns to line 30 for the next question.
190–210 If you were right, it prints "RIGHT!", adds 1 to the number correct, and returns to line 30 for the next question.
220–280 Scoring the test:
220 This formula computes your score.
230 If your score is below 90, it skips to line 270.
240–260 If your score is 90 or above, it prints a message and skips to line 280.
270 Prints a message.
280 Prints your score.

If you know more facts about the planets, you may add them to your data list. Make sure the flag data comes last! (You may also use this program to make up a test about something else!)

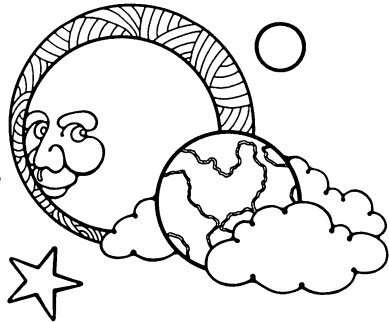


Passing the Test

How well do you know your planets?

```
10 C=0
20 T=0
30 FOR P=1 TO 500
40 NEXT P
50 CALL CLEAR
60 PRINT TAB(9); "PLANET TEST"
70 FOR P=1 TO 10
80 PRINT
90 NEXT P
100 READ PL$,F$
110 IF PL$="END" THEN 220
120 PRINT F$
130 PRINT "WHAT IS THE PLANET?"
140 INPUT A$
150 T=T+1
160 IF A$=PL$ THEN 190
170 PRINT "WRONG, THE ANSWER WAS "; PL$
180 GOTO 30
190 PRINT "RIGHT!"
200 C=C+1
210 GOTO 30
220 SC=INT(C/T*100+.5)
230 IF SC<90 THEN 270
240 IF SC>=90 THEN 250
250 PRINT "YOU PASSED THE TEST!"
260 GOTO 280
270 PRINT "SORRY. YOU DID NOT PASS!"
280 PRINT "YOUR SCORE WAS "; SC

300 DATA SATURN,THE PLANET WITH RINGS
310 DATA EARTH,OUR PLANET
320 DATA VENUS,THE MORNING & EVENING 'STAR'
330 DATA MARS,THE RED PLANET
340 DATA JUPITER,THE LARGEST PLANET
350 DATA PLUTO,THE FARTHEST PLANET
360 DATA MERCURY,NEAREST TO THE SUN
370 DATA NEPTUNE,THE GOD OF THE SEA
380 DATA URANUS,7TH FROM THE SUN
390 DATA PLUTO,A FAMOUS DOG
400 DATA END,END
```

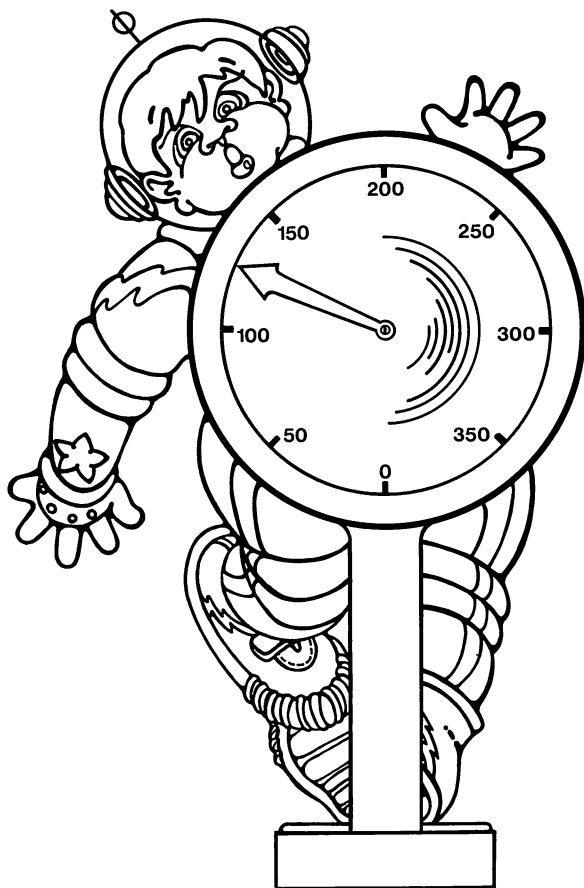


To be an astronaut,
you must pass this test.
A score of 90 is passing.
Good luck!

Main ideas: Working with a simple conversion formula
Setting up a chart

- 10 Clears the screen.
- 20–30 Your chance to input your weight on Earth.
- 40 Clears the screen again.
- 50–90 Prints and underlines titles for the chart.
- 100 Skips a line.
- 110–180 Prints the name of each planet; then computes and prints your weight on that planet.

Note the use of the comma in lines 80 to 180. This puts your data into two neat columns or “print zones.”



Gravity

Surface gravity varies on the planets. As part of your training, you must experience various gravitational pulls.

```
10 CALL CLEAR
20 PRINT "WHAT IS YOUR EARTH WEIGHT?"
30 INPUT W
40 CALL CLEAR
50 PRINT " WEIGHT ON EARTH:";W
60 PRINT " WEIGHT ON OTHER PLANETS"
70 PRINT " -----"
80 PRINT " PLANET", "WEIGHT"
90 PRINT " -----", "-----"
100 PRINT
110 PRINT " MERCURY", .37*W
120 PRINT " VENUS", .876*W
130 PRINT " MARS", .381*W
140 PRINT " JUPITER", 2.637*W
150 PRINT " SATURN", 1.151*W
160 PRINT " URANUS", .79*W
170 PRINT " NEPTUNE", 1.12*W
180 PRINT " PLUTO", .025*W:::
```

How much would you weigh on Pluto?
on Venus? on Jupiter? Find out!

Main ideas: Color “explosion”
Infinite loop

- 10 Clears the screen.
- 20 Allows you to input your name. (This is really a combination PRINT–INPUT statement!)
- 30 Clears the screen.
- 40 Inserts your name.
- 50 Allows you to answer the question.
- 60 Clears the screen again.
- 70 Counts over to 5 and prints a message.
- 80 Picks a number from 1 to 16 for the color.
- 90 Changes the color of the screen to the color picked in line 80.
- 100 Counts over to 12 and prints “WRONG.”
- 110 Picks a random sound from 110 to 1110.
- 120 Makes the sound picked in line 110.
- 130 Goes to line 80 to pick a new color, setting the program into a never-ending loop! Press FCTN–4 to break the loop.

In the program above, “WRONG” is always placed at the 12th box. This addition will change all that:

- 95 The computer picks a number from 1 to 24.
- 100 Prints “WRONG” at the tab picked.

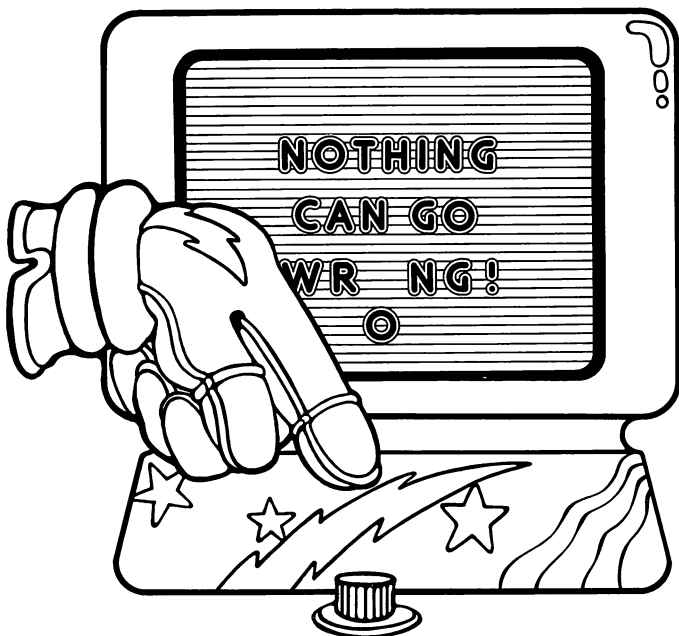
(Each line has 28 spaces. By picking any number from 1 to 24, the word “WRONG” will always fit on the line!)

Test Your Spacecraft

You are about to take off into outer space. Before embarking on your journey, run this program to test your spacecraft. Don't worry! Nothing can go wrong!

```
10 CALL CLEAR
20 INPUT "WHAT IS YOUR NAME? ":N$
30 CALL CLEAR
40 PRINT "HELLO, CAPTAIN ";N$
50 INPUT "ARE YOU READY TO GO? ":A$
60 CALL CLEAR
70 PRINT TAB(5);"NOTHING CAN GO WRONG!":
80 C=INT(16*RND)+1
90 CALL SCREEN(C)
100 PRINT TAB(12);"WRONG"
110 N=INT(1000*RND)+110
120 CALL SOUND(1,N,10)
130 GOTO 80
```

```
95 TB=INT(24*RND)+1
100 PRINT TAB(TB);"WRONG"
```



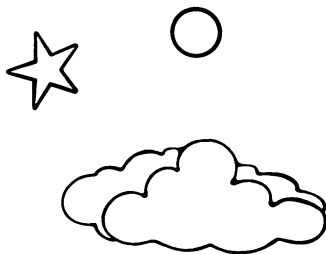
Main ideas: Color "explosion"
Fading sound

- 10 Clears the screen.
- 20 Prints "BLAST OFF!"
- 30-60 Exploding color loop:
 - 30 There will be 14 rounds of color change. (You can change the 14 to a different number!)
 - 40 Picks a number, 1 to 16.
 - 50 Clears the screen with the color picked.
 - 60 Goes back to the beginning of the loop.
- 70-100 Fades the loudness from 1 (the loudest) to 30 (the softest):
 - 80 Plays the sound. The -7 gives you a static kind of noise.
 - 90 The print makes the screen scroll up one line. This makes the "BLAST OFF!" message move up.

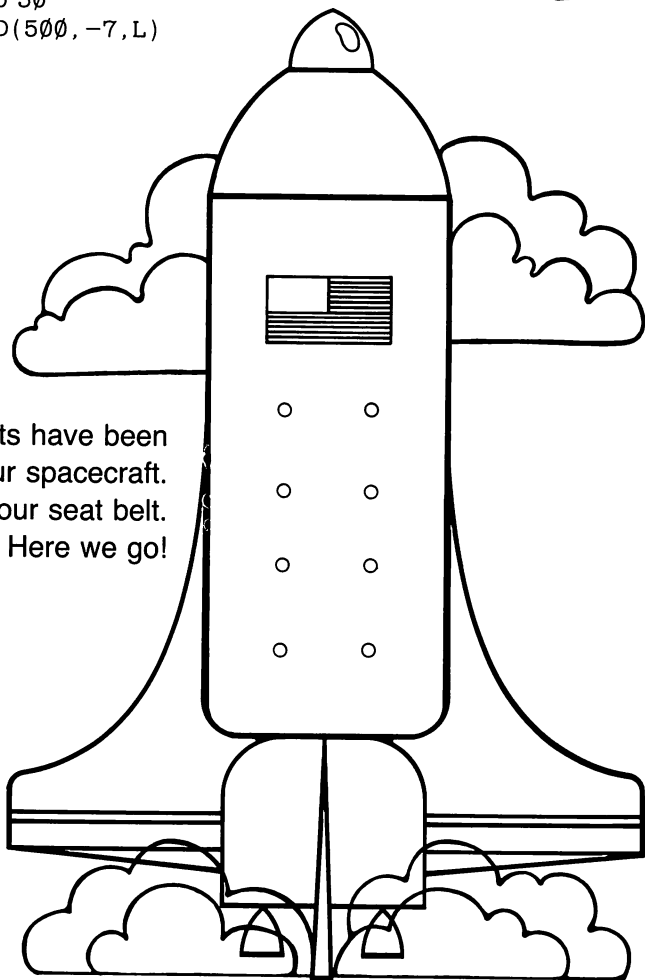


Take Off

```
10 CALL CLEAR
20 PRINT TAB(10); "BLAST OFF!"
30 FOR EX=1 TO 14
40 A=INT(16*RND)+1
50 CALL SCREEN(A)
60 NEXT EX
70 FOR L=1 TO 30
80 CALL SOUND(500, -7, L)
90 PRINT
100 NEXT L
```



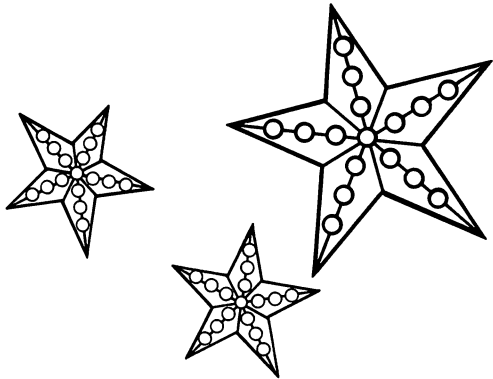
Adjustments have been
made to your spacecraft.
Fasten your seat belt.
Here we go!



Main ideas: Counting by incrementing

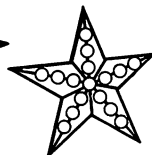
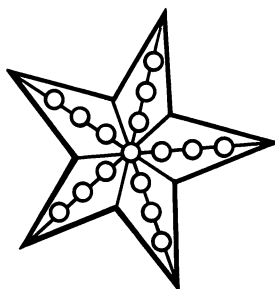
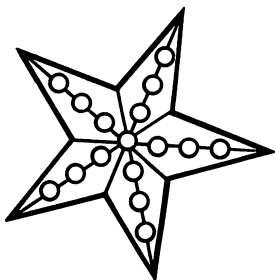
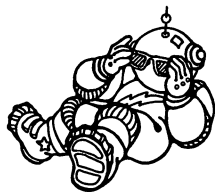
- 10 Sets the number of the star to 1.
- 20 Clears the screen.
- 30 Makes the color of the screen blue.
- 40–60 Prints the title. (The colons at the end of line 60 position the title up 9 rows.)
- 70–80 Pauses while the computer counts to itself to 500. (This gives you a chance to read the message.)
- 90 Clears the screen.
- 100 We are going to be printing white stars (color 16) on a blue background (color 6). Since the * is in character set 2, the result is CALL COLOR (2,16,2). Refer to the Character Sets and the Color Chart in the Appendix.
- 110 Picks a number for the tab from 1 to 23. This allows enough room to print the number *and* the star!
- 120 Prints the number and the star.
- 130 Increases the number by 1. The next star will be #2, the next star #3, etc.
- 140 Goes to line 110 to pick a spot to “count” the next star.

(To get yourself out of the infinite loop, press FCTN–4.)

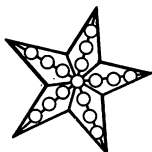


Count the Stars

The stars are just beautiful!
It's amazing how many of them there are.
Start to count them.



```
10 N=1
20 CALL CLEAR
30 CALL SCREEN(6)
40 PRINT " TWINKLE TWINKLE LITTLE STAR"
50 PRINT TAB(8); "HOW I WONDER"
60 PRINT TAB(5); "HOW MANY THERE ARE!": .....:
70 FOR P=1 TO 500
80 NEXT P
90 CALL CLEAR
100 CALL COLOR(2,16,6)
110 TB=INT(23*RND)+1
120 PRINT TAB(TB);N;"*"
130 N=N+1
140 GOTO 110
```



Press FCTN-4 to break the program.
Type CONTINUE to continue the counting.

Main idea: Using READ-DATA to write a song

- 10 Randomizes the selection of colors (line 40) so that the order is different each time the program is run.
- 20 Clears the screen.
- 30 Prints the message—the colons scroll it up the screen.
- 40 Picks a number from 1 to 16.
- 50 Changes the screen color to the number picked in line 40.
- 60 Reads the note and number of beats (data in lines 100 to 120).

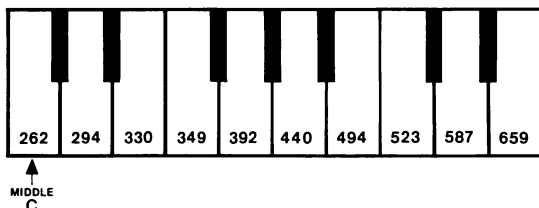
In this program, N = Note

B = Number of beats to hold the note

The numeric equivalents of some of the notes are shown in the piano diagram. To play this song on the piano, you would play:

Note N for B Beats

C(262) for 1 beat
C(262) for 1 beat
G(392) for 1 beat
G(392) for 1 beat
A(440) for 1 beat
etc.



This is how the data lines were put together. If you want a different song, simply change the data for the notes and beats.

- 70 Checks for the “flag” data—to signal the end of the song (see line 130).
- 80 Makes the sound—each beat gets 200 milliseconds. (If you want the song faster, make the number less than 200. If you want the song slower, make the number more than 200.)
- 90 Goes to line 40 to change the color of the screen and play the next note.
- 100–120 Data for the notes and beats.
- 130 Flag data.
- 140 Prints a message.
- 150 A dummy line freezes the image on the screen.

To make the music repeat, add these extra lines:

- 150–160 Pauses to let you read “My Favorite Song.” (Leave it out and see how fast it blinks!)
- 170 Restores the data in lines 100 to 130 so it can be read again.
- 180 Goes to line 20 to completely clear the screen and start over.

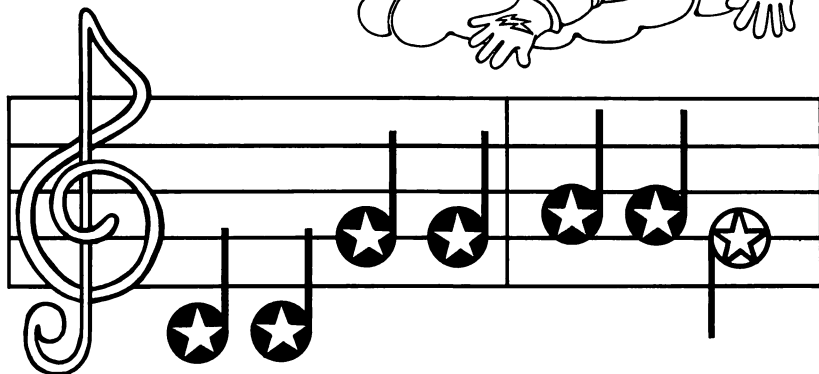
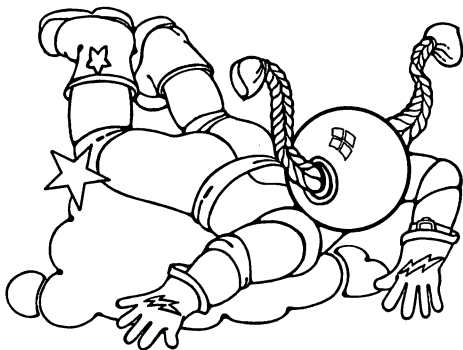
Remember: Press FCTN-4 to break out of an infinite loop! You can type CONTINUE and it will continue from where it stopped.

Music from Mother Earth

What's this? They're sending you music from Earth.
They must be testing the sound system!

```
10 RANDOMIZE
20 CALL CLEAR
30 PRINT " MUSIC FROM MOTHER EARTH" : .....
40 C=INT(16*RND)+1
50 CALL SCREEN(C)
60 READ N,B
70 IF N=-1 THEN 140
80 CALL SOUND(B*200,N,5)
90 GOTO 40
100 DATA 262,1,262,1,392,1,392,1,440,1,440,1,392,2
110 DATA 349,1,349,1,330,1,330,1,294,1,294,1,262,2
115 DATA 392,1,392,1,349,1,349,1,330,1,330,1,294,2
120 DATA 392,1,392,1,349,1,349,1,330,1,330,1,294,2
130 DATA -1,-1
140 PRINT TAB(7); "MY FAVORITE SONG!":...
150 GOTO 150

150 FOR P=1 TO 500
160 NEXT P
170 RESTORE
180 GOTO 20
```



Main ideas: Using the CALL KEY statement
ASCII codes
Converting key strokes to notes

- 10 Clears the screen.
- 20 Counts over to 7th position and prints.
- 30 Counts to 5th position and prints. The colons make the message scroll up. (Experiment with the number of colons you like!)
- 40 The CALL KEY statement allows you to press a key on the keyboard and turn it into an action. (In this case, the "action" is playing a note! Don't hold the key too long, or you'll get more notes than you want!)
- 60 You are awfully slow compared to a computer! This is to give you enough time to press a key. ST is the status indicator. If ST=0, it means "no key pressed yet."
- 70–150 Refers to the Character Sets in the Appendix. Each character has its own code, called the ASCII code. The code for 1 is 49, 2 is 50, etc. Lines 140 and 150 are checks to make sure that you pressed a number from 1 to 8. If not, the ASCII code number will be less than 49 or more than 56, and you'll go back to line 10!
- 155 Defines what note is to be played, as do lines 170, 180, 190, 200, 210, 220, and 230. Notice that after all these lines you go to line 160, which plays the note.
- 160 Plays the note (see the Musical Notes in your Appendix to see where the numbers for the notes come from!)
- 165 Sends you back to line 40 to get the next key pressed and start all over!

Make Your Own Kind of Music

Why not play your own song? Press keys 1 to 8!

```
10 CALL CLEAR
20 PRINT TAB(7); "TO PLAY MUSIC,"
30 PRINT TAB(7); "PRESS KEYS 1-8" : .....:
40 CALL KEY(0,N,ST)
60 IF ST=0 THEN 40
70 IF N=49 THEN 170
80 IF N=50 THEN 180
90 IF N=51 THEN 190
100 IF N=52 THEN 200
110 IF N=53 THEN 210
120 IF N=54 THEN 220
130 IF N=55 THEN 230
140 IF N<49 THEN 10
150 IF N>56 THEN 10
155 NOTE=523
160 CALL SOUND(500,NOTE,5)
165 GOTO 40
170 NOTE=262
175 GOTO 160
180 NOTE=294
185 GOTO 160
190 NOTE=330
195 GOTO 160
200 NOTE=349
205 GOTO 160
210 NOTE=392
215 GOTO 160
220 NOTE=440
225 GOTO 160
230 NOTE=494
235 GOTO 160
```



Main idea: How to make a drawing

To make this creature, I decided to use these characters:

Eyes	@	ASCII #64	Character set #5
Nose	<	ASCII #60	Character set #4
	>	ASCII #62	Character set #4
Bellybutton	.	ASCII #46	Character set #2

I could use any other character set number to make plain colored squares for drawing, but I decided on set #3 and picked a "2" as the character to be drawn, ASCII #50. (In general, avoid using set #1 when doing graphics because it contains the code for a space, ASCII #32. This makes things complicated!)

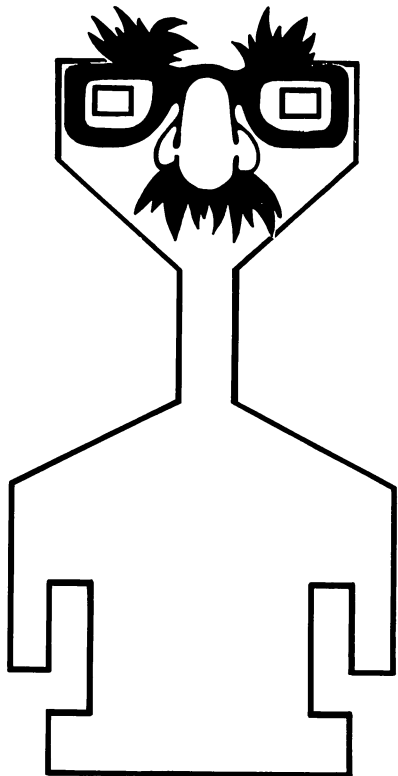
Now I can explain the program. Type it in section by section to help you understand how it works. (Better yet, see if you can draw this guysy on your Graphics Sheet in the Appendix.)

- 10 Clears the screen.
- 20 Makes the background light blue.
- 30 You decide what color creature you want (see chart of Colors).
- 40 Clears the screen again.
- 50 All characters in set #3 will be the color you chose on the same color background. Since the color of the character is the same color as the background, you get plain colored squares.
- 60–80 Head—3 rows, 5 squares in each row.
- 90 Neck—a vertical row of 4 squares.
- 100–110 Shoulders—2 horizontal rows of 9 squares each.
- 120–130 Arms—2 vertical rows of 4 squares each.
- 140–160 Body—5 horizontal rows of 5 squares each.
- 170–180 Feet—2 horizontal rows of 11 squares each.
- 190–210 Eyes:
 - 190 All characters in set #5 (including the "@") will be black on the color you chose.
- 200–210 Draws the two "@" signs (character #64).
- 220–240 Nose:
 - 220 All characters in set #4 (including the "<" and ">") will be black on the color you chose.
 - 230 Draws the "<".
 - 240 Draws the ">".
- 250–260 Bellybutton:
 - 250 All characters in set #2 (including the ".") will be black on the color you chose.
 - 260 Draws the ".".
 - 270 A dummy line keeps the picture on the screen and prevents a "DONE" message.

What Is It?

Something scary has appeared on your screen. What is it?

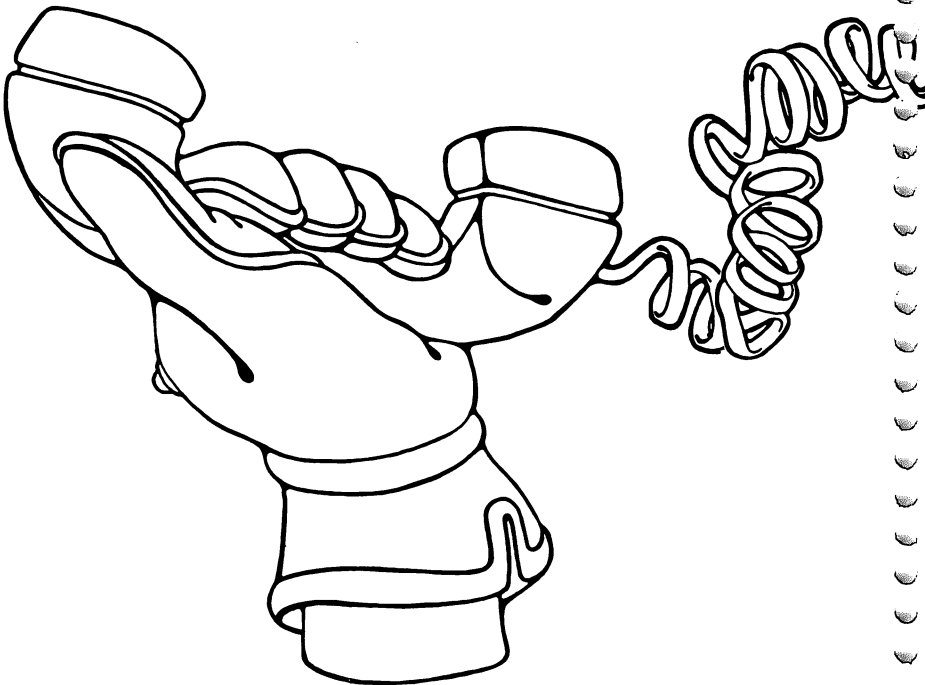
```
10 CALL CLEAR
20 CALL SCREEN(6)
30 INPUT "WHAT COLOR IS HE?(1-16)":C
40 CALL CLEAR
50 CALL COLOR(3,C,C)
60 FOR R=8 TO 10
70 CALL HCHAR(R,14,50,5)
80 NEXT R
90 CALL VCHAR(11,16,50,4)
100 CALL HCHAR(15,12,50,9)
110 CALL HCHAR(16,12,50,9)
120 CALL VCHAR(17,12,50,4)
130 CALL VCHAR(17,20,50,4)
140 FOR R=17 TO 21
150 CALL HCHAR(R,14,50,5)
160 NEXT R
170 CALL HCHAR(22,11,50,11)
180 CALL HCHAR(23,11,50,11)
190 CALL COLOR(5,2,C)
200 CALL HCHAR(7,14,64)
210 CALL HCHAR(7,18,64)
220 CALL COLOR(4,2,C)
230 CALL HCHAR(8,15,60)
240 CALL HCHAR(8,17,62)
250 CALL COLOR(2,2,C)
260 CALL HCHAR(18,16,46)
270 GOTO 270
```



Main idea: Adding music to graphics

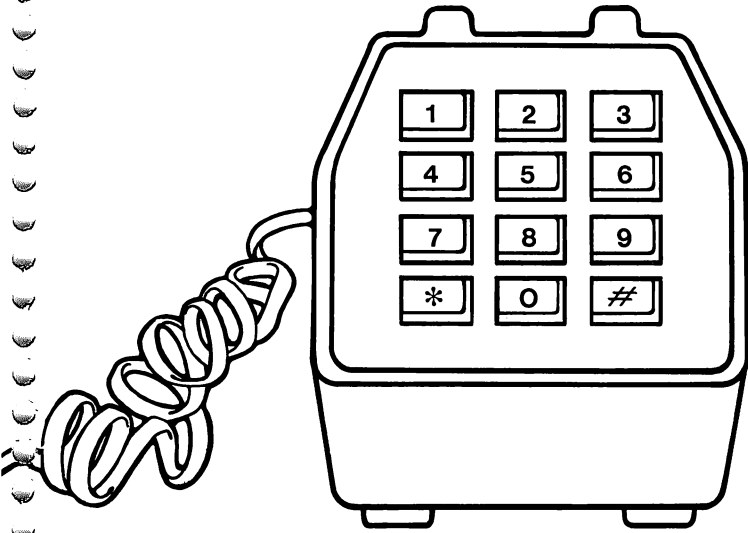
(Be sure to take out line 270 if you are adding this program to the last one!)

- 300 Reads N (for note) and B (for number of beats). The data is at lines 340 to 360.
- 310 Checks for the "flag" data to see if the song is finished. If the song is finished, the program skips to line 370.
- 320 Plays the note. (Notice that each beat receives 150 milliseconds. You can change the 150 to speed up or slow down the song!)
- 330 Goes back to line 300 to read the next 2 numbers for the note and beats.
- 340–360 Data for the notes and beats.
- 370 Some of the letters in my message (lines 380 to 400) were coming out on a colored background. I had to write this line to make the letters come out on a blue background. Notice what this does to the creature's eyes.
- 380–400 Prints the messages.
- 410 A dummy line freezes the image and prevents a "DONE" message.



Who Is It?

Aha! Do you know who it is?

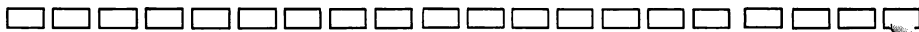
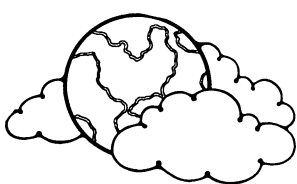


Add these lines to your last program to solve the mystery!

```
300 READ N,B
310 IF N=-1 THEN 370
320 CALL SOUND(B*150,N,5)
330 GOTO 300
340 DATA 262,3,392,6,349,1,311,1,294,1,262,1,247,3,262,6
350 DATA 220,3,440,6,392,1,349,1,330,1,294,1,277,3,294,6
360 DATA -1,-1
370 CALL COLOR(5,2,6)
380 PRINT "IT'S TEE HEE!!"
390 PRINT "UNFRIENDLY ALIEN"
400 PRINT "HE NEVER EVER PHONES HOME!"
410 GOTO 410
```

Main ideas: Random numbers
Initializing and incrementing

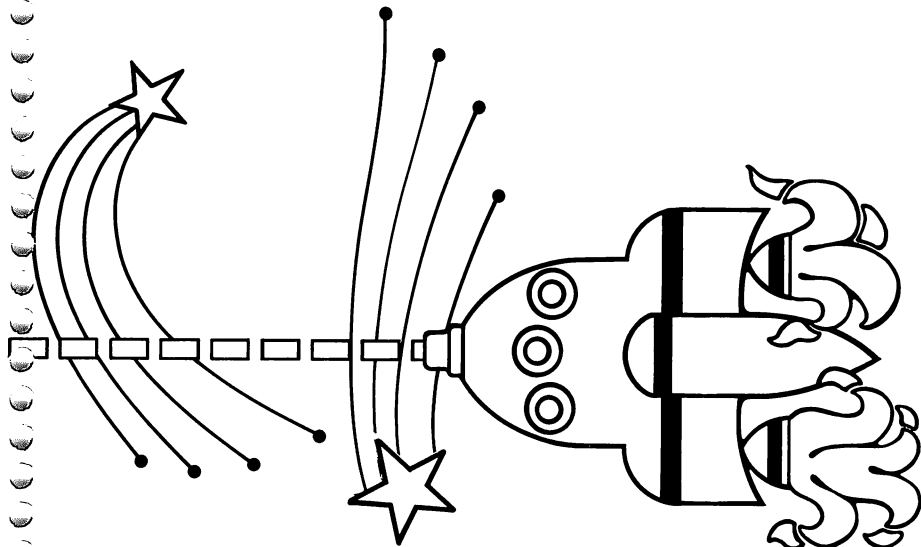
- 10–15 At the beginning of the game, there are no hits and no misses.
16 Randomizing is necessary if you want the score to come out differently each time. (Try the program without this line and see!)
- 20–75 Goes through 100 rounds:
30 Selects 2 numbers, a 0 or a 1.
40 If it selects a 0, it skips to line 60 (where it adds 1 to the hits).
50 Otherwise it adds 1 to the number of misses and goes to line 55.
55 Skips to line 70.
60 Adds 1 to the hits (only if it came from line 40).
70 Prints the score. (Note: The semicolon makes the computer print 2 pieces of information right next to each other. The comma sends the computer to the second print zone, halfway across the screen.)
75 Finishes the round and goes back to line 20.
77 Skips a line by printing a blank line.
- 80–120 Evaluates the results:
80 If the hits are more than or equal to 50, it skips to line 100 (and prints a congratulatory message).
90 Prints the message and goes to line 95.
95 Ends.
- 100–120 Prints only if the program is sent here from line 80.



Hit or Miss?

You have 100 rounds of ammunition. Fifty hits will knock him out. Can you do it? If not, he has promised to visit Earth and destroy all the telephones!

```
10 H=0
15 M=0
16 RANDOMIZE
20 FOR R=1 TO 100
30 X=INT(2*RND)
40 IF X=0 THEN 60
50 M=M+1
55 GOTO 70
60 H=H+1
70 PRINT "HITS "; H, "MISSES "; M
75 NEXT R
77 PRINT
80 IF H>=50 THEN 100
90 PRINT "NO MORE CALLS TO GRANDMA!"
95 END
100 PRINT "YOU DID IT!"
110 PRINT "PHONE HOME AND TELL THEM"
120 PRINT "ALL ABOUT IT!"
```



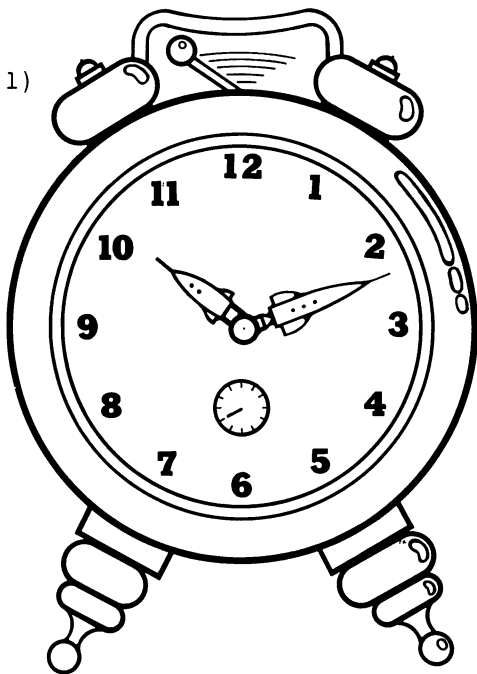
Main ideas: Using sound for something besides music
Scrolling, fast and slow
Setting all the characters in the same format

- 10 Clears the screen.
- 20 Makes the screen black (color 2).
- 30–50 Makes all characters yellow (color 12) or black (color 2).
- 60 Counts over 8 spaces, prints, and scrolls up. (Vary the TAB setting and the number of colons to position the message somewhere else!)
- 70–80 Pauses while the computer counts to 1000.
- 90 Clears the screen.
- 100–130 Makes 10 trills (for an alarm clock sound) by playing 2 close tones, 10 times each. (I used tones 300 and 301. You can chose others!)
- 140 Clears the screen in yellow (color 12).
- 150–170 Makes all characters red (color 9) on yellow (color 12).
- 180 Counts over 8 spaces and prints a message at the bottom of the screen.
- 190–220 Slowly moves the message up the screen as the sound gets higher:
 - 190 Since the sounds start at line 110, I started P at 2 so that the first note in line 210 would be note 200.
 - 200 Prints a blank line. (The message moves up.)
 - 210 Makes the sound 100*P, holding it for 50 milliseconds, loud.
 - 220 Completes the loop.
- 230 A dummy line freezes the image and prevents a “DONE” message. Type FCTN–4 to break the program.

Nighty Night!

That was hard work! It's time to get a good night's sleep.

```
10 CALL CLEAR
20 CALL SCREEN(2)
30 FOR S=1 TO 12
40 CALL COLOR(S,12,2)
50 NEXT S
60 PRINT TAB(8); "NIGHTY NIGHT!!":;
70 FOR P=1 TO 1000
80 NEXT P
90 CALL CLEAR
100 FOR A=1 TO 10
110 CALL SOUND(1,300,2)
120 CALL SOUND(1,301,2)
130 NEXT A
140 CALL SCREEN(12)
150 FOR S=1 TO 12
160 CALL COLOR(S,9,12)
170 NEXT S
180 PRINT TAB(8); "RISE AND SHINE!"
190 FOR P=2 TO 20
200 PRINT
210 CALL SOUND(50,1000*P,1)
220 NEXT P
230 GOTO 230
```



Main ideas: Using logic to draw a geometric design
Using CALL COLOR and CALL SCREEN for quick color changes

- 10 Clears the screen.
- 20 Makes the screen cyan (color 8).
- 30 I am going to “draw” with character #42, which is in set #2. For yellow boxes, I have made all characters in set #2 yellow (color 12) on yellow (color 12).
- 40–130 This loop will make 6 sets of dots. The first time through the loop, all dots will be at the same location (6,6). See if you can work your way through the formula and complete the chart below.
- 210 Counts over 3 and prints.
- 220 A dummy line freezes the image and prevents a “DONE” message. Type FCTN–4 to break the program.

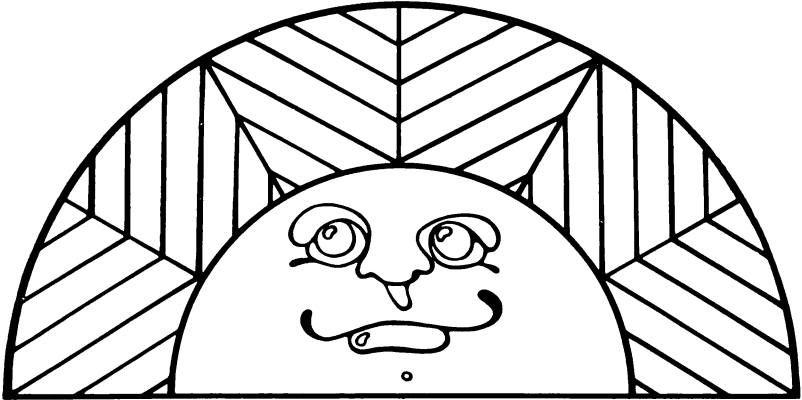
<i>Line numbers</i>	<i>Loop 1</i>	<i>Loop 2</i>	<i>Loop 3</i>	<i>Loop 4</i>	<i>Loop 5</i>	<i>Loop 6</i>
	<i>N = 6</i>	<i>N = 5</i>	<i>N = 4</i>	<i>N = 3</i>	<i>N = 2</i>	<i>N = 1</i>
50 CALL HCHAR (6,N)	(6,6)	(6,5)				(6,1)
60 CALL HCHAR (N,N)	(6,6)	(5,5)				(1,1)
70 CALL HCHAR (N,6)	(6,6)	(5,6)				(1,6)
80 CALL HCHAR (N, 12-N)	(6,6)	(5,7)				(1,11)
90 CALL HCHAR (6, 12-N)	(6,6)	(6,7)				(6,11)
100 CALL HCHAR (12-N,12-N)	(6,6)	(7,7)				(11,11)
110 CALL HCHAR (12-N,6)	(6,6)	(7,6)				(11,6)
120 CALL HCHAR (12-N,N)	(6,6)	(7,5)				(11,1)

Do you see any patterns? For a real challenge, try plotting the dots one at a time, beginning with loop 1 and continuing through loop 6!

For some splashy color, add the lines at the bottom of the page. They will take you through a loop of colors.

- 140 The loop will go through colors 8 to 16, calling each color C.
- 150 The screen will be color C–5.
- 160 The sun will be color C.
- 170–180 A pause (so you can see this color combination).
- 190 End of the loop.
- 200 Final color setting will be medium red (color 9) on a dark yellow (color 11) background. (This is because the last CALL SCREEN was 16–5, or 11!)

Good Morning!



Each time you circle Earth, there's a beautiful new sunrise!
The colors are outstanding!

```
10 CALL CLEAR
20 CALL SCREEN(8)
30 CALL COLOR(2,12,12)
40 FOR N=6 TO 1 STEP -1
50 CALL HCHAR(6,N,42)
60 CALL HCHAR(N,N,42)
70 CALL HCHAR(N,6,42)
80 CALL HCHAR(N,12-N,42)
90 CALL HCHAR(6,12-N,42)
100 CALL HCHAR(12-N,12-N,42)
110 CALL HCHAR(12-N,6,42)
120 CALL HCHAR(12-N,N,42)
130 NEXT N
210 PRINT TAB(3); "A SPECTACULAR SUNRISE!!";
220 GOTO 220

140 FOR C=8 TO 16
150 CALL SCREEN(C-5)
160 CALL COLOR(2,C,C)
170 FOR P=1 TO 500
180 NEXT P
190 NEXT C
200 CALL COLOR(2,9,9)
```

Main idea: Using the arrow keys to draw on the screen

- 10–20 Clears the screen and makes it black.
- 30 We are going to make an “@” for each meteor. @ is character #64 in set # 5 (see Appendix). This line makes set #5 yellow (color 12) on black (color 2).
- 40 Makes random numbers.
- 50–90 Makes 50 @s on the screen at random locations.
- 100–110 Sets the beginning row and column at the bottom right.
- 120 Makes the characters in set #6 light blue on light blue (color 6).
- 130 Character #72 is in set #6, so you get a light blue square at 1,1.
- 140 Makes the characters in set #2 red on red (color 9).
- 150 Makes a red square at R,C (see lines 100 to 110).
- 160 Checks to see if you're at (1,1). If R=1, you are sent to line 320 to see if C=1.
- 170 Checks to see if you've pressed any key.
- 180 If you haven't, you get another chance.

The next section is best understood if you follow the routes.

- 190→230→240→310→150: If you press “↑” (on the E key, code 69), then subtract 1 from R (the rows) and draw a square at the new location.
 - 200→250→260→310→150: If you press “↓” (on the X key, code 88), then add 1 to R (the rows) and draw a red square.
 - 210→270→280→310→150: If you press “→” (on the D key, code 68), then add 1 to C (the columns) and draw the red square.
 - 220→290→310→150: If you press “←” (on the S key, code 83), then subtract 1 from the columns and draw the red square at the new location.
- 320 If you're at the first column, then the program skips to line 340.
 - 330 Goes to line 170 to check for the next key pressed.
 - 340–350 Congratulates you.

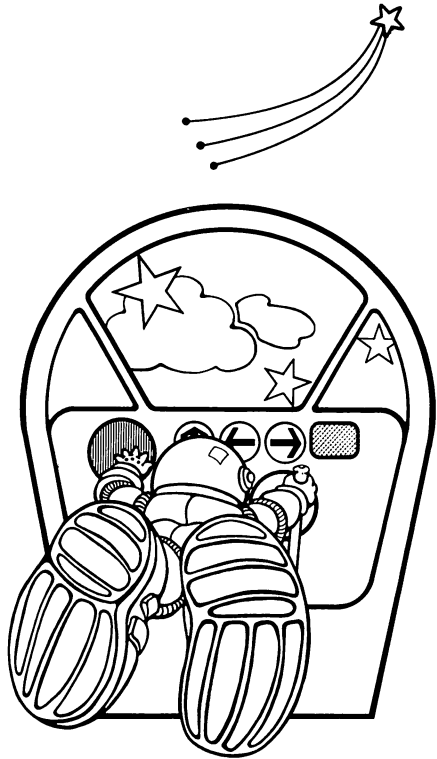
Be careful not to go too far left, right, up, or down. You will get an error because the computer cannot draw off the screen! The error is especially annoying as you near your destination at the upper left-hand corner. To avoid the error, add these lines to your program:

```
150 IF R<1 THEN 400
151 IF C<1 THEN 500
360 END
400 R=1
410 GOTO 155
500 C=1
510 GOTO 155
```

Meteors!

It's a meteor shower!
Use your arrow keys to
steer your way through it!
You're safe if you get to
the upper left corner!

```
10 CALL CLEAR
20 CALL SCREEN(2)
30 CALL COLOR(5,12,2)
40 RANDOMIZE
50 FOR M=1 TO 50
60 R=INT(24*RND)+1
70 C=INT(32*RND)+1
80 CALL HCHAR(R,C,64)
90 NEXT M
100 R=24
110 C=32
120 CALL COLOR(6,6,6)
130 CALL HCHAR(1,1,72)
140 CALL COLOR(2,9,9)
150 REM
155 CALL HCHAR(R,C,42)
160 IF R=1 THEN 320
170 CALL KEY(0,KEY,STATUS)
180 IF STATUS=0 THEN 170
190 IF KEY=69 THEN 230
200 IF KEY=88 THEN 250
210 IF KEY=68 THEN 270
220 IF KEY=83 THEN 290
230 R=R-1
240 GOTO 310
250 R=R+1
260 GOTO 310
270 C=C+1
280 GOTO 310
290 C=C-1
310 GOTO 150
320 IF C=1 THEN 340
330 GOTO 170
340 CALL CLEAR
350 PRINT "YOU MADE IT!"
```



Main ideas: Guessing game

Using TAB and colons to position a message

- 10 Clears the screen.
- 20 For random numbers.
- 30 Selects a number, 0 to 99.
- 40 Allows you to guess.
- 50 If you've guessed correctly, goes to line 130.
- 60 If you've guessed too low, skips to line 100.
- 70–90 If you've guessed too high, prints a message, skips a line, and goes back to line 40 for another guess.
- 100–120 Prints a message, skips a line, and goes back to line 40 so you can guess again.
- 130 Prints "RIGHT!"

Instead of the simple message "RIGHT!", you may make the suggested addition to the program.

- 130 Clears the screen.
- 140 If the fuel left is less than 30 gallons, skips to line 200. Line 200 prints the bad news and moves the message up the screen.
- 150 If the fuel left is higher than 30 but less than 60 gallons, skips to line 180. Line 180 prints the message and moves it up the screen.
- 160–170 Prints this if you have 60 or more gallons of fuel.

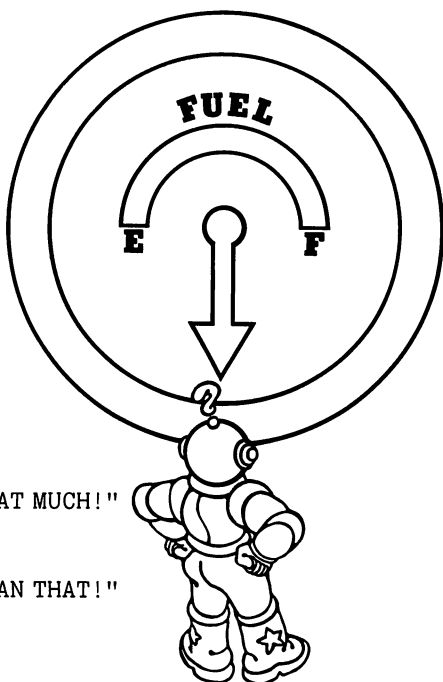
Note: Lines 170 and 210 are dummy lines, which freeze the message on the screen. To break the program, you must press FCTN–4.

How Much Fuel?

Oh no!

The fuel gauge is broken!
Guess how much fuel
you have left. (A full tank
holds 100 gallons.)

```
10 CALL CLEAR
20 RANDOMIZE
30 F=INT(100*RND)
40 INPUT "HOW MANY GALLONS
   ARE LEFT?":G
50 IF G=F THEN 130
60 IF G<F THEN 100
70 PRINT "YOU DON'T HAVE THAT MUCH!"
80 PRINT
90 GOTO 40
100 PRINT "YOU HAVE MORE THAN THAT!"
110 PRINT
120 GOTO 40
130 PRINT "RIGHT!"
```



By the way, you need at least 30 gallons to land safely!

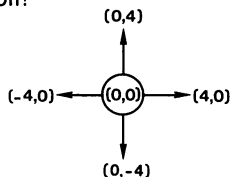
```
130 CALL CLEAR
140 IF F<30 THEN 200
150 IF F<60 THEN 180
160 PRINT TAB(10);"EXCELLENT!":.....
170 GOTO 170
180 PRINT TAB(8);"SHOULD BE OK!":.....
190 GOTO 190
200 PRINT TAB(10);"BAD NEWS!":.....
210 GOTO 210
```


Main idea: Using the joystick to draw
Using the fire button to change colors

Before you begin, you might like to type in this short program to see how the joystick works. See if you can find the numbers on the diagram below.

Note: To work properly, the ALPHA LOCK must be off!

```
1Ø CALL JOYST(1,X,Y)
2Ø PRINT X;Y
3Ø GOTO 1Ø
```



Now type "NEW" and start the program:

```
100 Clears the screen.
101-104 Allows you to pick a color to draw with (HUE) and a color for the
screen (SCR).
110-120 Sets the beginning row and column to 1. Point (1,1) is in the upper
left corner.
130 Clears the screen.
135-140 Your choice of colors for the drawing and screen.
200 Reads the coordinates of the joystick (X,Y).
205-500 See if you can trace the program. Here's what happens: If you
press upwards, X becomes 0 and Y becomes 4; the computer
subtracts 1 from the number of rows. If you press downwards, X
becomes 0 and Y becomes -4; the computer adds 1 to the number
of rows. If you press towards the left, X becomes -4 and Y be-
comes 0; the computer subtracts 1 from the number of columns. If
you press towards the right, X becomes 4 and Y becomes 0; the
computer adds 1 to the number of columns.
560 Makes a square at the new row and column location.
600 Goes back to get the next joystick reading.
```

Add these lines to keep your drawing on the screen:

```
500-552 There are 24 rows and 32 columns on a graphics screen. If you
push the joystick too far, it will get to 25 or 33 (too high) or it will get
down to 0 (too low). If you try to make a box at a location that
doesn't exist, you will get an error. To avoid errors, if the rows get to
be more than 24, we keep R at 24. If the columns get to be more
than 32, we keep C at 32. If R or C get to 0, we keep them at 1. This
is what this section of the program does!
```

Add these lines for the "strange light":

```
570-585 Changes the color of the drawing at random:
570 Checks to see if you've pressed the fire button.
575 If you pressed the fire button, KEY becomes 18 and you continue to
line 580. If the button hasn't been pressed, you skip to line 600.
580-581 Picks a number from 1 to 16 for the drawing and the screen color.
585 Goes to line 135 to change the color of HUE and SCR.
```

Space Sketch

```

100 CALL CLEAR
101 PRINT "PICK A COLOR(1-16)"
102 INPUT HUE
103 PRINT "WHAT COLOR BACKGROUND?"
104 INPUT SCR
110 R=1
120 C=1
130 CALL CLEAR
135 CALL SCREEN(SCR)
140 CALL COLOR(2,HUE,HUE)
200 CALL JOYST(1,X,Y)
205 IF X=0 THEN 210
206 GOTO 300
210 IF Y=4 THEN 250
215 IF Y=0 THEN 560
220 R=R+1
221 GOTO 500
250 R=R-1
251 GOTO 500
300 IF X=-4 THEN 310
305 GOTO 400
310 C=C-1
351 GOTO 500
400 C=C+1
450 GOTO 500
500 REM
560 CALL HCHAR(R,C,42)
600 GOTO 200

```

As you near your destination, you see something strange. Sketch what you see, using your video sketch panel and the drawing mechanism (the joystick!).

Be sure to keep your drawing on the screen:

```

500 REM KEEPING THE DRAWING
    ON THE SCREEN
510 IF R>24 THEN 535
515 IF R<1 THEN 537
520 GOTO 540
535 R=24
536 GOTO 540
537 R=1
540 IF C>32 THEN 550
541 IF C<1 THEN 552
545 GOTO 560
550 C=32
551 GOTO 560
552 C=1

```

What was that strange light?
(Press your button to see it):

```

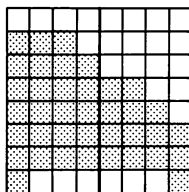
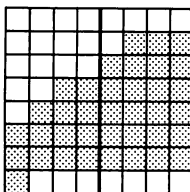
570 CALL KEY(1,KEY,STATUS)
575 IF KEY<>18 THEN 600
580 HUE=INT(16*RND)+1
581 SCR=INT(16*RND)+1
585 GOTO 135

```

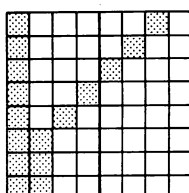
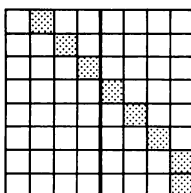
Main ideas: Designing your own characters
Random placement on the screen

Each character that the computer makes contains 64 tiny dots arranged in an 8×8 square. Each row contains 8 dots, divided into a left block of 4 dots and a right block of 4 dots. To design a character, you must tell how many dots to light up and how many to leave off. The chart on page 49 will help you to find the hexadecimal code for each row in your design.

You will need to design 4 characters to make the parachute. Here they are separately:



A\$ = "00070F3F7FFFFF80" **B\$ = "00E0F0FCFEFFFF81"**



C\$ = "4020100804020101" **D\$ = "82848890A0C0C0C0"**

You can assign your characters any number from 128 to 159.

128–135 are in set #13. (All of ours are in set #13.)

136–143 are in set #14.

144–151 are in set #15.

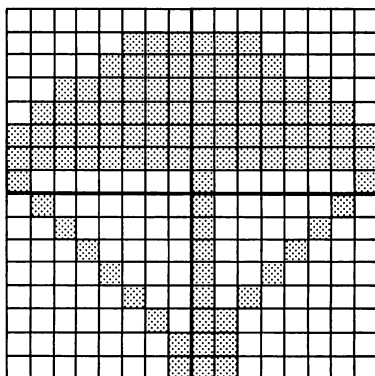
152–159 are in set #16.

The lines in the program accomplish the following:

- 5–80 Designs the 4 characters that make up the parachute.
- 90–100 Makes the parachute green (color 13) on a light blue background (color 6).
- 215–220 Picks a random location and draws character 128.
- 221 Draws character 129 one column to the right.
- 222 Draws character 130 one row down.
- 223 Draws character 131 one column over *and* one row down.

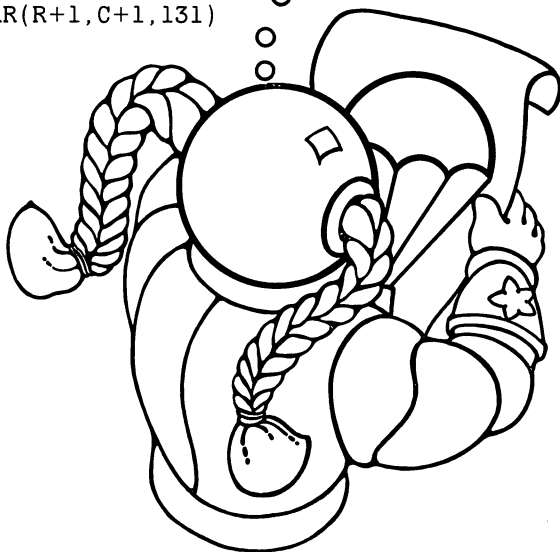
Design Your Parachute

Before leaving Earth,
you were asked to design
your own parachute.
Here it is:



```
5 CALL CLEAR
10 A$="00070F3F7FFFFFF80"
20 B$="00E0F0FCFEFFFF81"
30 C$="4020100804020101"
40 D$="82848890A0C0C0C0"
50 CALL CHAR(128,A$)
60 CALL CHAR(129,B$)
70 CALL CHAR(130,C$)
80 CALL CHAR(131,D$)
90 CALL SCREEN(6)
100 CALL COLOR(13,10,6)

215 RANDOMIZE
218 R=INT(19*RND)+1
219 C=INT(31*RND)+1
220 CALL HCHAR(R,C,128)
221 CALL HCHAR(R,C+1,129)
222 CALL HCHAR(R+1,C,130)
223 CALL HCHAR(R+1,C+1,131)
```



Main ideas: Counting backwards

Using CALL KEY to interrupt a loop

- 110 Prints the directions.
- 120–180 Makes a loop:
 - 120 You are going to count backwards from 700 to 200 by twenties.
 - 130 Makes the sound associated with the number you're up to.
 - 140 The computer checks to see if you've pressed a key.
 - 150 If not, you go to line 180 and back to the beginning of the loop.
 - 160 80 is the character number of "P". If you pressed something other than a "P", you go back to line 140.
 - 170 If you pressed "P", you escape from the loop to line 200.
 - 180 Ends the loop and returns to line 120 for another loop.
- 200–261 Checks to see if you are between 300 and 400 in your descent to Earth:
 - 200 If you are still more than 400, then you go to line 250, which prints that you were "too early" in opening your parachute.
 - 210 If you're less than 300, then you go to line 260, which prints that you were "too late" in opening your parachute.
- 215–230 If you're between 300 and 400, the computer tells you that you were "just in time." It also makes the parachute you designed in the last program. **Note:** You don't get the parachute unless you press "P" at just the right time.

To make the game easier, widen the range in lines 200 and 210. For example:

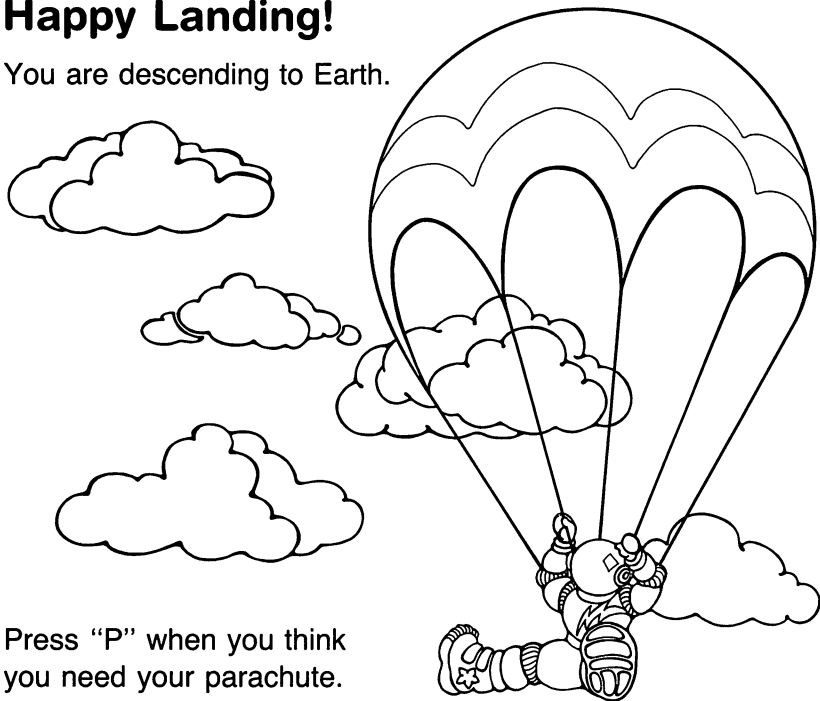
```
200 IF N>500 THEN 250 }  
210 IF N<300 THEN 260 } (a range of 200)
```

To make the game harder, narrow the range in lines 200 and 210. For example:

```
200 IF N>250 THEN 250 }  
210 IF N<200 THEN 260 } (a range of 50)
```

Happy Landing!

You are descending to Earth.



Press "P" when you think
you need your parachute.
Timing is critical!

(Add these lines to your last program!)

```
110 PRINT "    PRESS P FOR PARACHUTE."
120 FOR N=700 TO 200 STEP -20
130 CALL SOUND(50,N,1)
140 CALL KEY(0,X,STATUS)
150 IF STATUS=0 THEN 180
160 IF X<>80 THEN 140
170 IF X=80 THEN 200
180 NEXT N
200 IF N>400 THEN 250
210 IF N<300 THEN 260

230 PRINT "    JUST IN TIME!"
240 GOTO 240
250 PRINT "    TOO EARLY!"
251 END
260 PRINT "    TOO LATE!"
261 END
```

Main ideas: Simple word processing
Substituting variables in a text

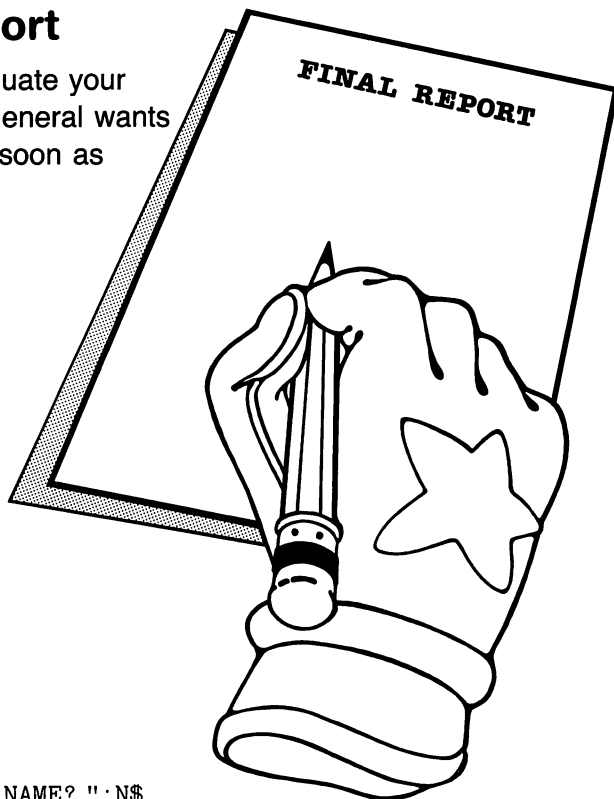
- 5 Clears the screen.
- 10–60 A chance for you to input variables.
- 100 A remark.
- 110 Clears the screen again.
- 120–190 Prints the report using the variables input in lines 10 to 60.

Note the use of semicolons and spaces in getting the report to print out correctly. Semicolons make the next word printed appear immediately after the last word printed. You must put the spaces in when you want them!

Also note the difference between string variables and numeric variables: In lines 10 to 50, you will put in words—hence N\$, G\$, PL\$, L\$, D\$ (string variables). In line 60, you will put in a number—hence N (no dollar sign, since N is a numeric variable).

Final Report

It's time to evaluate your mission. The General wants your report as soon as possible.



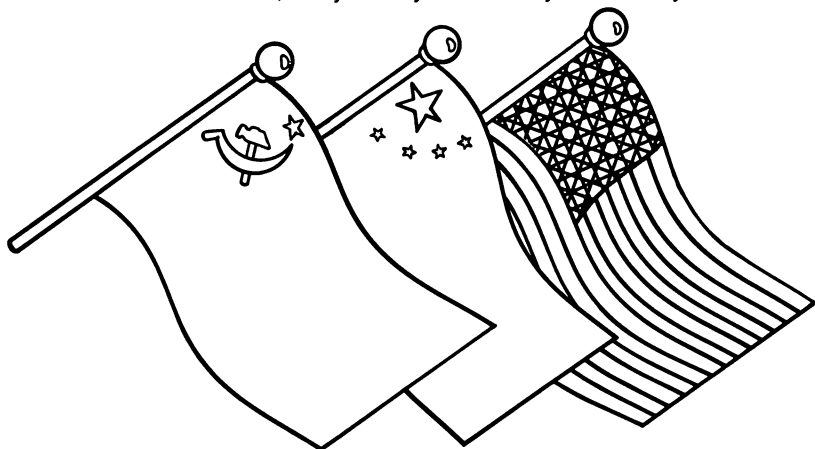
```
5 CALL CLEAR
10 INPUT "YOUR NAME? ":N$
20 INPUT "NAME OF GENERAL IS GEN. ":G$
30 INPUT "PLANET VISITED? ":PL$
40 INPUT "LENGTH OF TRIP? ":L$
50 INPUT "DAY OF WEEK? ":D$
60 INPUT "A NUMBER? ":N
100 REM PRINT REPORT
110 CALL CLEAR
120 PRINT "REPORT OF ";N$
130 PRINT "TO GEN. ";G$
140 PRINT
150 PRINT "MY MISSION TO ";PL$
160 PRINT "IS OVER."
170 PRINT "THE MISSION LASTED ";L$
180 PRINT "I WILL SEND YOU A ";N;"PAGE"
190 PRINT "REPORT NEXT ";D$
```


Main ideas: Staging a race

Using ON . . . GOTO for branching

- 5 Clears the screen.
- 10–40 Sets up the screen by printing the titles and advancing them up the screen with colons.
- 50–70 U stands for the United States, R stands for Russia, and C stands for China. All three will start the “race” at box 3, right under the names of their respective countries.
- 80–90 The computer randomly picks a number from 1 to 3.
 - 95 If it picked a 1, it goes to line 100. If it picked a 2, it goes to line 200. If it picked a 3, it goes to line 300.
- 100–130 If it picked a 1, a star is placed in row 10, box U. “*” is character 42. U is increased by 1, so that the next star will be printed 1 box over, just next to the last star printed. There are 32 boxes in the row. If U goes up to 33, you have reached the end of the row. This is good news for the United States, so you are sent to line 400 where it prints “CONGRATULATIONS.” If U is not up to 33 yet, you go back to line 90 where the computer chooses again who will get the next star.
- 200–230 This is what happens if the computer picked a 2: Russia gets the vote, and a star is placed in row 10, box R. R is increased by 1 for next time. If R is up to 33, the United States has lost and you are sent to line 500 where it prints the bad news.
- 300–330 China gets the vote if the computer picked a 3.
- 400–510 Prints the results of the race and ends the program.

If you like, you may choose different characters for each country. Look at the Character Sets in the Appendix, and change the 42 in lines 100, 200, and 300. Character 42 is the “*”, but you may choose any character you like!



The Race for Space

Your voyage is over, Captain. Have you done the very best for your country? Find out who has won the race for space.

```
5 CALL CLEAR
10 PRINT "    THE RACE FOR SPACE"::::
20 PRINT "U.S.A.":::::
30 PRINT "U.S.S.R.":::::
40 PRINT "CHINA":::::
50 U=3
60 R=3
70 C=3
80 RANDOMIZE
90 X=INT(3*RND)+1
95 ON X GOTO 100,200,300
100 CALL HCHAR(10,U,42)
110 U=U+1
120 IF U=33 THEN 400
130 GOTO 90
200 CALL HCHAR(15,R,42)
210 R=R+1
220 IF R=33 THEN 500
230 GOTO 90
300 CALL HCHAR(20,C,42)
310 C=C+1
320 IF C=33 THEN 500
330 GOTO 90
400 PRINT "CONGRATULATIONS, CAPTAIN!"
410 END
500 PRINT "TRY HARDER, CAPTAIN!"
510 END
```

Main ideas: Using ASCII codes for characters
Using CHR\$(N), ASC(A\$), and TAB

- 10 Clears the screen.
- 20 L is used to position each character on the screen (see line 50). At the beginning of the program, L is 0.
- 30 Reads a number (from the data lines).
- 40 If it reads a -1, this is a signal to skip to line 100.
- 50 CHR\$(N) converts the number to its corresponding character. TAB makes the computer count over to 8 + L to print the character. The first time through the loop, 8 + L = 8, so column 8 is used to print.
- 60 L is increased by 1. Next time L will be 1, so 8 + L = 9 and the next character will be printed at box 9 (see line 50 above).
- 70 Goes back to read the next number.
- 80 Data for the characters.
- 90 "Flag" data.
- 100 Advances the entire message to mid-screen.
- 110 Freezes the image and prevents the "DONE" message. Press FCTN-4 to break the program.

The second program works just backwards from the one above. ASC(A\$) will give you the code number for whatever character you insert as the A\$. For example: "PRINT ASC(R)" will give you "82."

- 10 Clears the screen.
- 20-30 You input the character whose code you want.
- 40 Converts the character (A\$) to a number (A).
- 50 Prints the code number.
- 60 Goes back to line 20 to ask you for the next one.

Break the Code

The General has sent you this final message. Run the program to break the code.

```

10 CALL CLEAR
20 L=0
30 READ N
40 IF N=-1 THEN 100
50 PRINT TAB(8+L);CHR$(N);
60 L=L+1
70 GOTO 30
80 DATA 79,86,69,82,32,65,78,68,32,79,85,84,33
90 DATA -1
100 PRINT:.....
110 GOTO 110

```

To write a return message, put your own numbers into line 50. Use the chart or run this program to get the numbers you need.

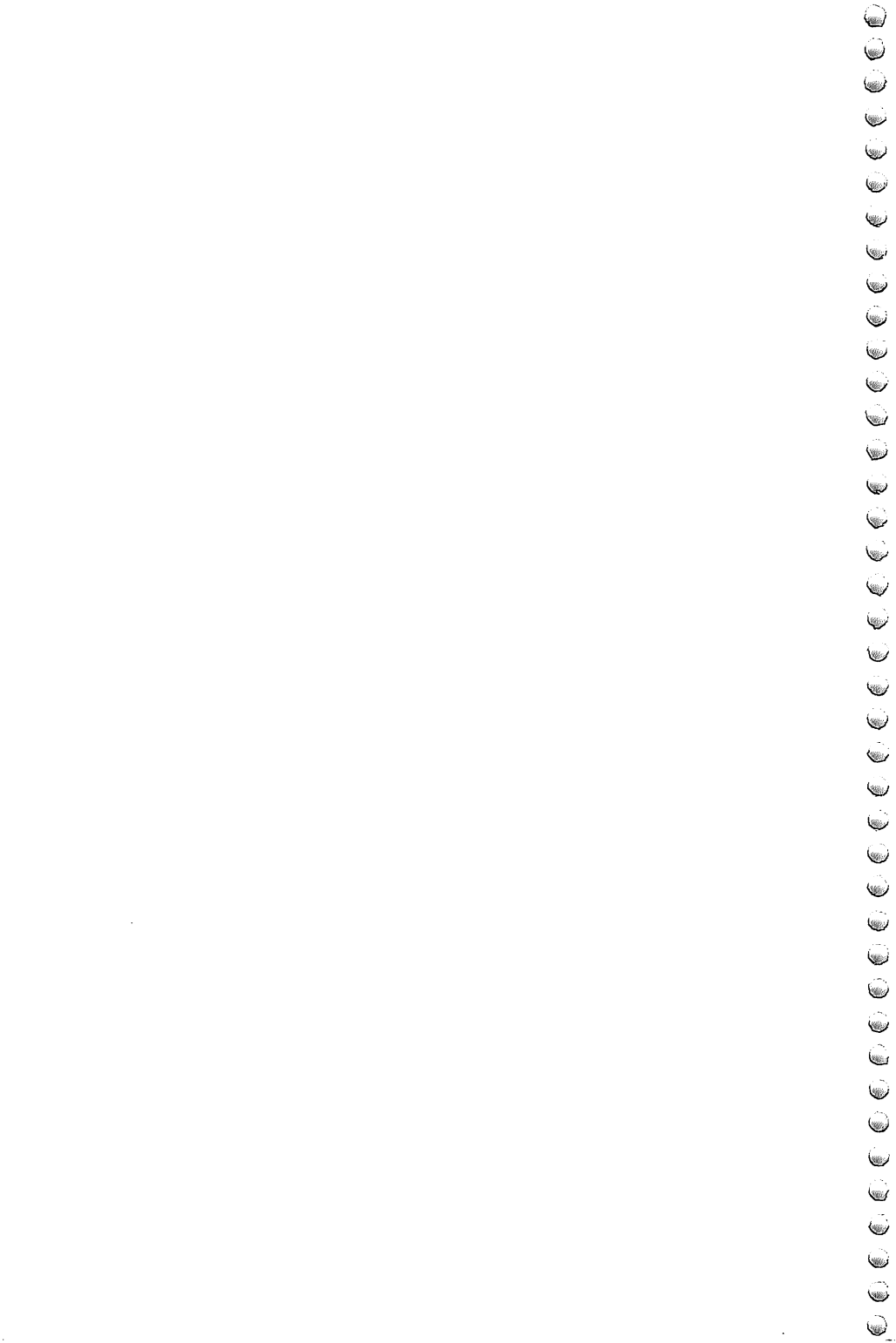
```

10 CALL CLEAR
20 PRINT "WHAT CHARACTER DO YOU WANT"
30 INPUT "TO KNOW? ":A$
40 A=ASC(A$)
50 PRINT "ITS CODE IS ";A
60 GOTO 20

```

THE END

84 72 69 32 69 78 68



APPENDIX

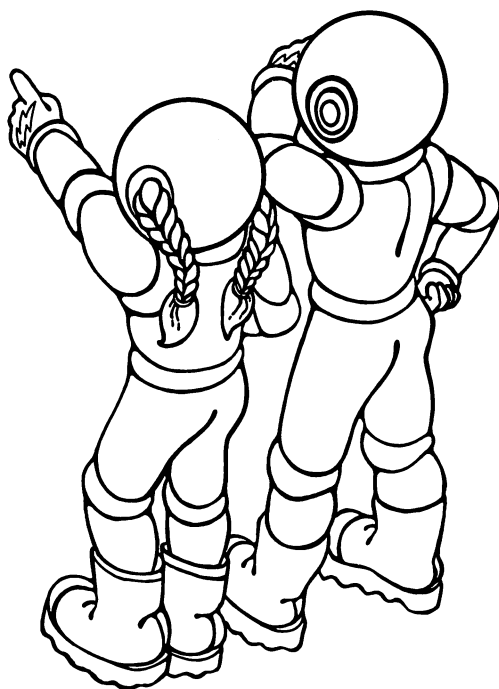
Character Sets, 45

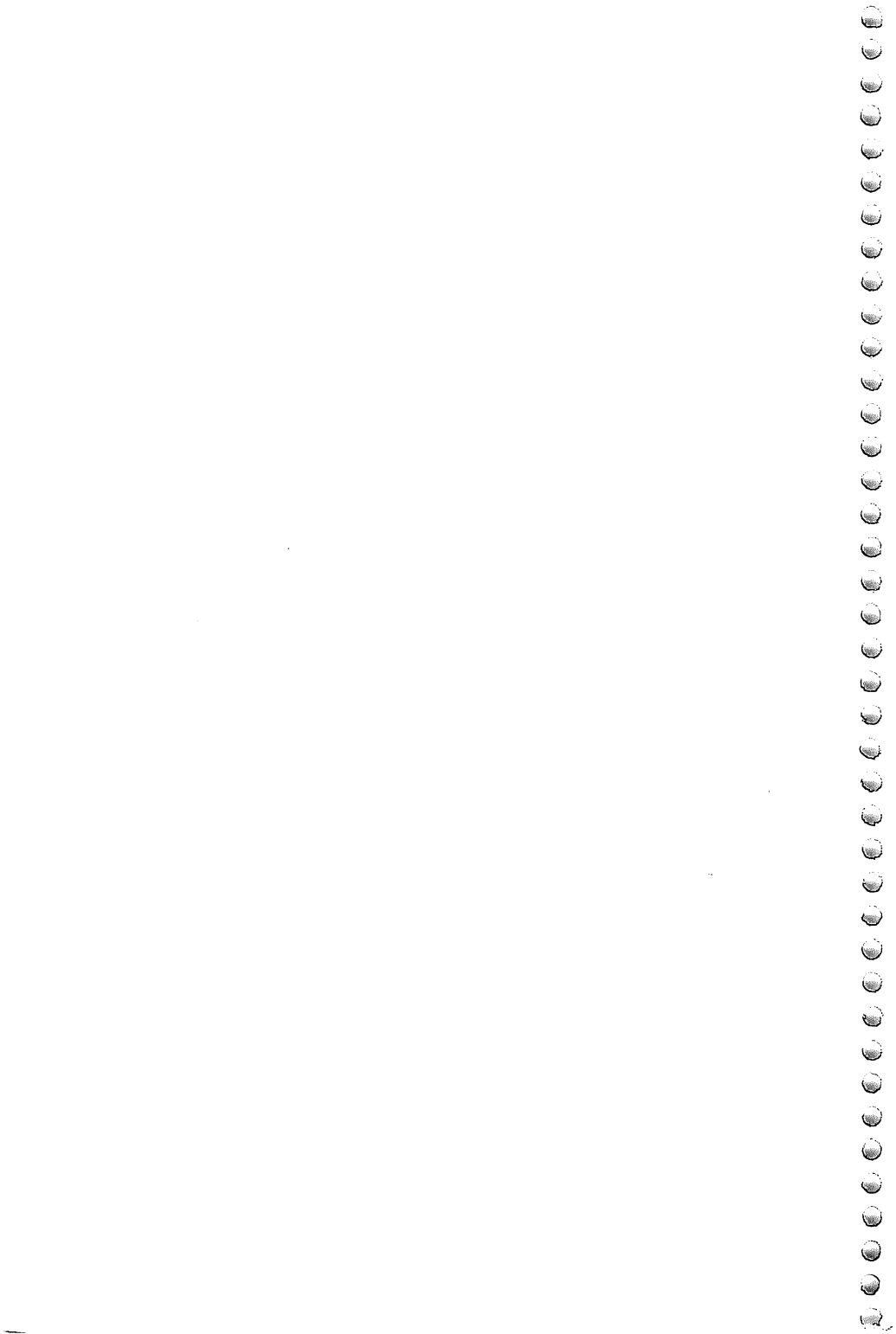
Colors, 47

Design Your Own Character, 49

Graphics Sheet, 50–51

Musical Notes, 53

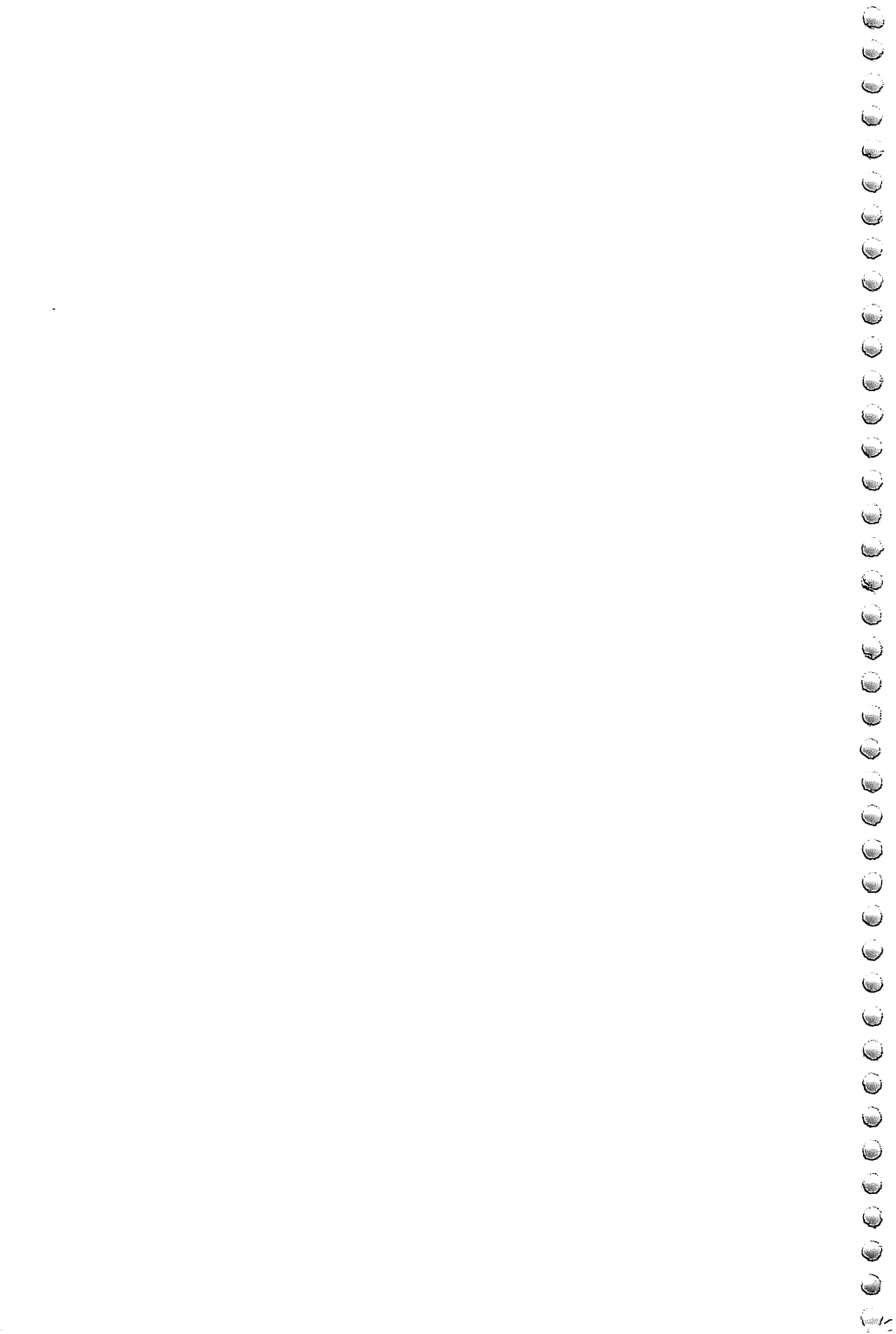




Character Sets

Set #1		Set #2		Set #3		Set #4		Set #5		Set #6	
Code	Character	Code	Character	Code	Character	Code	Character	Code	Character	Code	Character
32	(space)	40	(48	0	56	8	64	@	72	H
33	!	41)	49	1	57	9	65	A	73	I
34	"	42	*	50	2	58	:	66	B	74	J
35	#	43	+	51	3	59	;	67	C	75	K
36	\$	44	,	52	4	60	<	68	D	76	L
37	%	45	-	53	5	61	=	69	E	77	M
38	&	46	.	54	6	62	>	70	F	78	N
39	,	47	/	55	7	63	?	71	G	79	O

Set #7		Set #8		Set #9		Set #10		Set #11		Set #12	
Code	Character	Code	Character	Code	Character	Code	Character	Code	Character	Code	Character
80	P	88	X	96	,	104	H	112	P	120	X
81	Q	89	Y	97	A	105	I	113	Q	121	Y
82	R	90	Z	98	B	106	J	114	R	122	Z
83	S	91	[99	C	107	K	115	S	123	{
84	T	92	\	100	D	108	L	116	T	124	
85	U	93]	101	E	109	M	117	U	125	}
86	V	94	^	102	F	110	N	118	V	126	~
87	W	95	_	103	G	111	O	119	W	127	DEL



Colors

<i>Color</i>	<i>Code #</i>
Transparent	1
Black	2
Medium Green	3
Light Green	4
Dark Blue	5
Light Blue	6
Dark Red	7
Cyan	8
Medium Red	9
Light Red	10
Dark Yellow	11
Light Yellow	12
Dark Green	13
Magenta	14
Gray	15
White	16

You need to know the numbers for the colors when you want to change the color of the screen. For example, CALL SCREEN(2) will give you a black screen! You also need to know the colors to change the colors of the characters and to do graphics. For example, CALL COLOR (3,14,6) will set all characters in set #3 magenta on a light blue background. CALL COLOR (3,6,6) will give you plain blue squares.

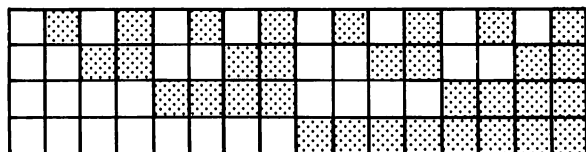


Design Your Own Character

To design your own character, you will have to convert patterns of dots to code. See "Design Your Own Parachute" for an example of how to do this.

Binary Code Hexadecimal
(0=off; 1=on)

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F



Graphics Sheet

There are 24 rows and 32 columns on the screen. When printing messages, the computer uses only 28 printing positions in each line, boxes 3 through 30. For graphics, though, it utilizes all 32 columns.

To print or draw at a certain position on the screen, you have to do a lot of counting! PRINT TAB(3); "LOOK AT THIS!" counts across to the 3rd position before printing your message.

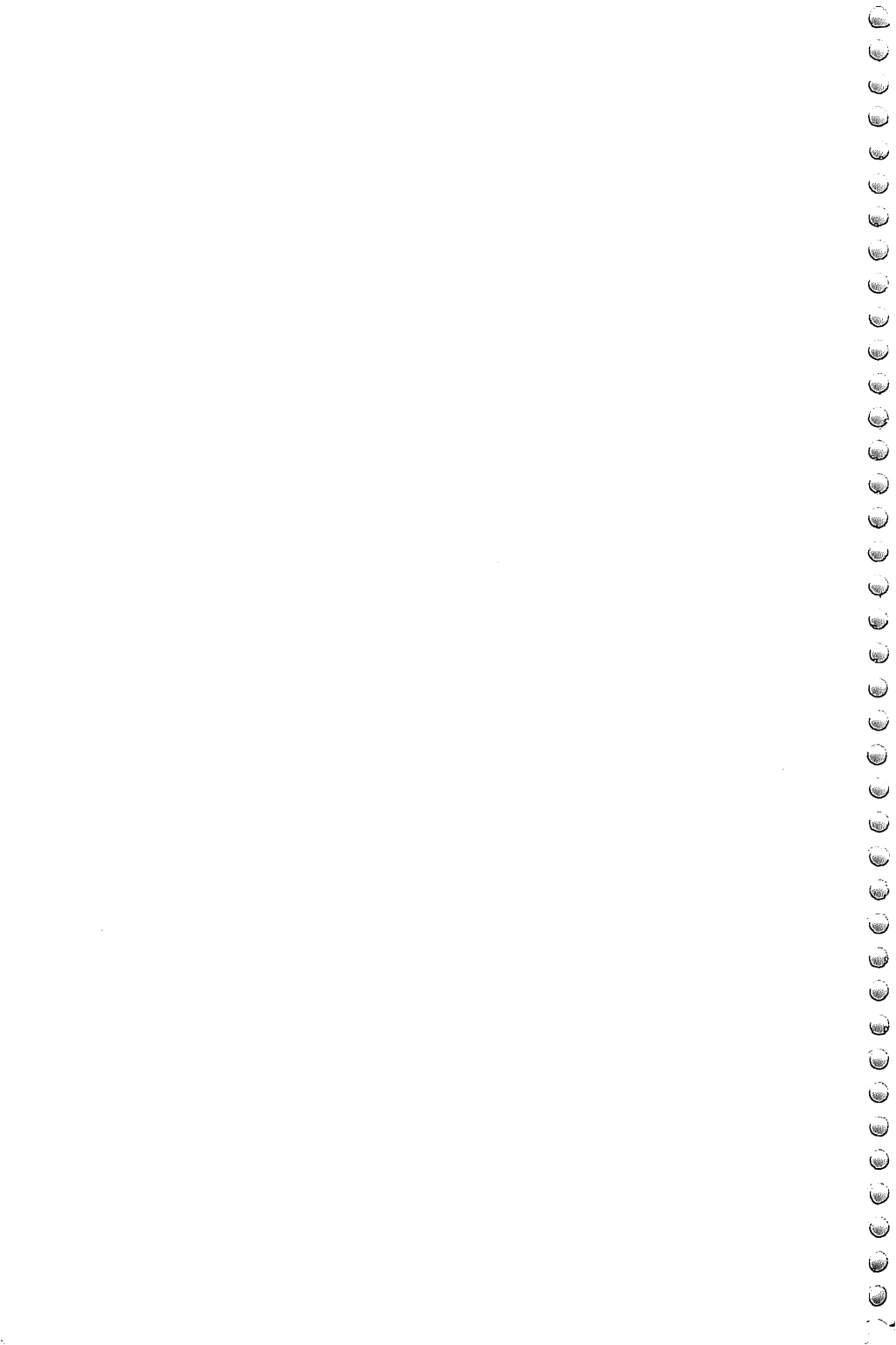
CALL HCHAR and CALL VCHAR are used with graphics. To draw something at a certain position on the screen, you find the position by counting *rows* first, then *columns*. For example, the position 5,3 is down 5 rows and across 3 columns. Or, put another way, it is the place where row 5 meets column 3.

To do well with graphics, you have to be able to find positions on the screen very easily. So practice!

COLUMNS

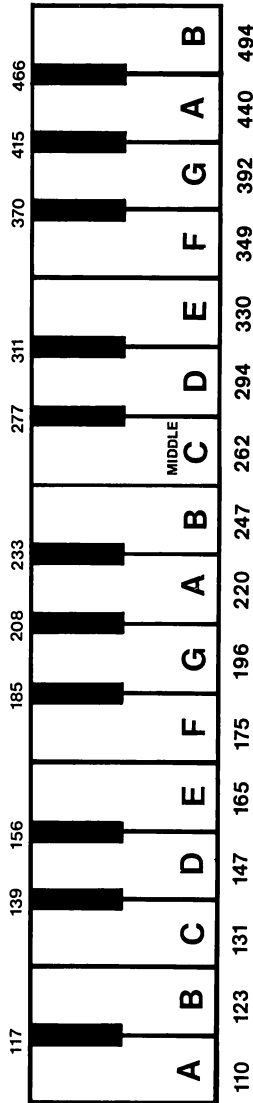
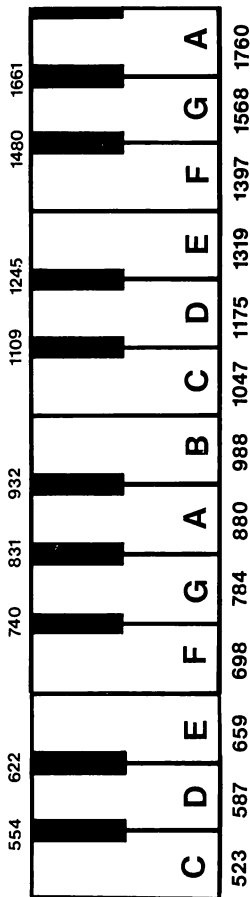
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1																															
2																															
3																															
4																															
5																															
6																															
7																															
8																															
9																															
10																															
11																															
12																															
13																															
14																															
15																															
16																															
17																															
18																															
19																															
20																															
21																															
22																															
23																															
24																															

ROWS



Musical Notes

Use the numeric equivalents of the notes to create your own computer songs!



\$6.95

CLAIRE BAILEY PASSANTINO

ITTY BITTY BYTES OF SPACE



SPACE—an endless source of fascination for us all. Children, however, have the imagination to project themselves into orbit as effortlessly as they learn to talk. Now, ITTY BITTY BYTES OF SPACE encourages your child to explore the magic of space with the help of your home computer. Young readers are presented with bite-size programs and activities designed to excite their eyes, ears, and minds. Each program is accompanied by programming hints and suggestions for parents and teachers. Help your children learn, grow, and play—the computer way!

Table of Contents: A Note to Parents and Teachers • Passing the Test • Gravity • Test Your Spacecraft • Take Off • Count the Stars • Music from Mother Earth • Make Your Own Kind of Music • What Is It? • Who Is It? • Hit or Miss? • Nighty Night! • Good Morning! • Meteors • How Much Fuel? • Space Sketch • Design Your Parachute • Happy Landing! • Final Report • The Race for Space • Break the Code

Other **Itty Bitty Bytes** Books: *Matilda the Computer Cat* • *School Days*

For more information on Creative Pastimes Books write to:

Reston Computer Group
11480 Sunset Hills Road
Reston, VA 22090

Illustration by Bethann Thornburgh



Reston Publishing Company, Inc.

A Prentice-Hall Company
Reston, Virginia

0-8359-3318-0