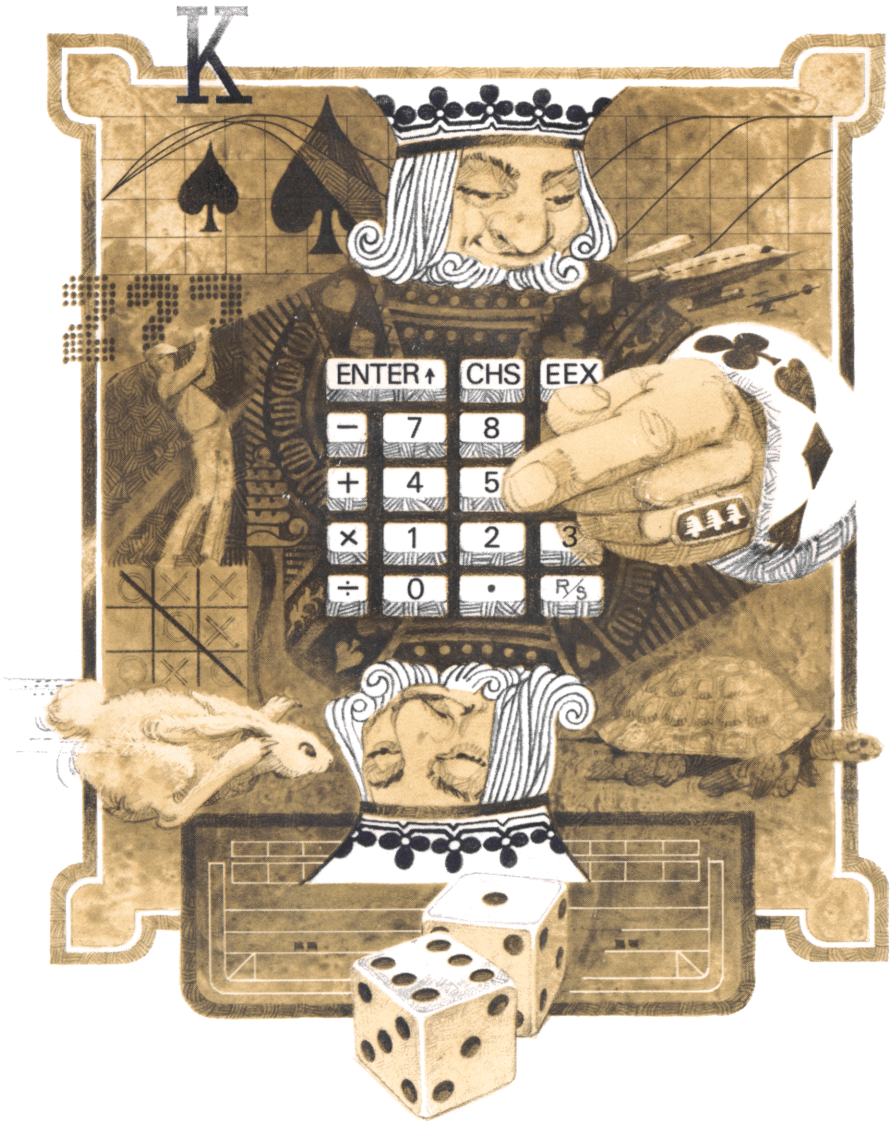


HEWLETT-PACKARD

HP-67/HP-97

Games Pac I



The program material contained herein is supplied without representation or warranty of any kind. Hewlett-Packard Company therefore assumes no responsibility and shall have no liability, consequential or otherwise, of any kind arising from the use of this program material or any part thereof.

WE NEED YOUR HELP

To provide better calculator support for people like you, we need your help. Your timely inputs will enable us to provide high quality software in the future and improve the existing application pacs for your calculator. Your early reply will be extremely helpful in this effort.

1. Pac name: 67/97 Games
2. How important was the availability of this pac in making your decision to buy a Hewlett-Packard calculator? ☐ Would not buy without it. ☐ Important
☐ Not important
3. Did you buy this pac and your calculator at the same time? ☐ Yes ☐ No
4. In deciding to buy this Games Pac, which three programs seemed most fun or useful to you? Program numbers 1. _____ 2. _____ 3. _____
5. Which three programs in this application pac seemed least fun or useful to you? Program numbers 1. _____ 2. _____ 3. _____
6. In the list below, please select up to three application areas for which you use your calculator between games. Please indicate the order of importance by 1, 2, 3 (1 represents the most important area).

Engineering	Business
___01 Chemical	___51 Accounting
___02 Civil/Structural	___52 Banking
___03 Electrical/Electronic	___53 Insurance
___04 Industrial	___54 Investment Analysis
___05 Mechanical	___55 Real Estate
___06 Surveying	___56 Securities
___10 Other (Specify) _____	___57 Sales
	___58 Marketing
Science	___59 Other (Specify) _____
___31 Biology	
___32 Chemistry	Other
___33 Earth Sciences	___71 Architecture
___34 Mathematics	___72 Aviation
___35 Medical Sciences	___73 Computer Science
___36 Physics	___74 Education
___37 Statistics	___75 Navigation
___39 Other (Specify) _____	___79 Other (Specify) _____

Thank you for your time and cooperation.

Name _____	Date _____
Address _____	
City _____	State _____
Zip _____	Phone _____

Please fold and staple for mailing.

Additional Comments:

FIRST CLASS

Permit No.
33

Corvallis,
Oregon

BUSINESS REPLY MAIL

No postage stamp is necessary if mailed in United States

Postage will be paid by:

Hewlett-Packard

1000 N.E. Circle Blvd.
Corvallis, OR 97330

ATTENTION: APPLICATIONS

Introduction

The 19 programs of Games Pac I are designed primarily to provide fun, and also to help teach principles of math, physics and logic. Included are card games, dice games, mathematical puzzles, outdoor sports and war games. Characteristics of each program are shown below.

We hope that Games Pac I will provide you with pleasure and education. We would very much appreciate knowing your reactions to the programs in this pac, and to this end we have provided a questionnaire inside the front cover of this manual. Would you please take a few minutes to give us your comments on these programs? It is in the comments we receive from you that we learn how best to increase the usefulness of programs like these.

PROGRAM CHARACTERISTICS											
		SOLITAIRE	2 PLAYERS	2 OR MORE PLAYERS	EASY TO WIN	HARD TO WIN	VARIABLE DIFFICULTY	CHANCE	SOME SKILL	TWO CALCULATORS CAN PLAY EACH OTHER	NOT A GAME
1	GAME OF 21	●	●	●				●			
2	DICE	●		●			●				
3	SLOT MACHINE	●		●			●				
4	SUBMARINE HUNT	●				●		●			
5	ARTILLERY	●				●		●			
6	SPACE WAR	●			●			●			
7	SUPER BAGELS	●				●		●			
8	NIM _k	●				●		●	●		
9	QUEEN BOARD	●		●				●			
10	HEXAPAWN	●				●		●	●		
11	TIC-TAC-TOE	●			●			●			
12	WARI	●	●		●			●	●		
13	RACETRACK	●		●				●			
14	TEASER	●			●			●			
15	GOLF	●	●			●		●			
16	THE DEALER	●	●								●
17	BOWLING SCOREKEEPER	●	●								●
18	BIORHYTHMS	●									●
19	TIMER	●									●

CONTENTS

Program	Page
Introduction	i
A Word about Program Usage	iv
GAMES	
1. Game of 21	01-01
This card game is also known as blackjack.	
2. Dice	02-01
This includes the game of "Craps" as well as a dice roller.	
3. Slot machine	03-01
The familiar one armed bandit.	
4. Submarine Hunt	04-01
Find and then sink the moving submarine with your depth charges.	
5. Artillery	05-01
Can you locate and destroy the moving target before it destroys you?	
6. Space War	06-01
Your mission: Search out and annihilate the 3 evil Alglogs before time and energy are gone.	
7. Super Bagels	07-01
Based on "Mastermind." How fast can you guess the secret number?	
8. Nim _k	08-01
Who will pick the last object from the last pile, you or the machine?	
9. Queen Board	09-01
You and the calculator take turns moving a chess queen to its target. The one who moves last, wins.	
10. Hexapawn	10-01
You and the 67/97 command armies of 3 chess pawns each. Caution: The calculator learns from its mistakes.	
11. Tic-Tac-Toe	11-01
Your best hope is to play the machine to a draw.	
12. Wari	12-01
You have a reasonable chance of beating the HP-67/97, but beware of a smart human. This ancient game is also known as Man-Kalah.	
13. Racetrack	13-01
Up to 5 players can race. Be alert to the differences between velocity and acceleration.	
14. Teaser	14-01
Changing from one pattern to the other looks easy, but ...	
15. Golf	15-01
The HP Country Club course is challenging, but a duffer with his handicap can beat a champion.	
16. The Dealer	16-01
This shuffles and deals a deck of cards to 4 people; it also calls Bingo.	
17. Bowling Scorekeeper	17-01
Tired of keeping score and missing the game? Here's your answer for up to 10 bowlers.	

18.	Biorhythms	18-01
	Calculates cycle values for any date, and tells which of the next 33 days are critical, maximum or minimum days.	
19.	Timer	19-01
	Offers 2 visible timers, a count-up and count-down timer, and allows splits to be taken.	
	Program Listings	L00-01
	Appendix A: Magnetic Card Symbols and Conventions	A-1

A WORD ABOUT PROGRAM USAGE

Each program in this pac is represented by one or more magnetic cards and a section in this manual. The manual provides a description of the program, a set of instructions for using the program, and one or more example problems, each of which includes a list of the actual keystrokes required for its solution. Program listings for all the programs in the pac appear at the back of this manual. Explanatory comments have been incorporated in the listings to facilitate your understanding of the actual working of each program. Thorough study of a commented listing can help you to expand your programming repertoire since interesting techniques can often be found in this way.

On the face of each magnetic card are various mnemonic symbols which provide shorthand instructions to the use of the program. You should first familiarize yourself with a program by running it once or twice while following the complete User Instructions in the manual. Thereafter, the mnemonics on the cards themselves should provide the necessary instructions, including what variables are to be input, which user-definable keys are to be pressed, and what values will be output. A full explanation of the mnemonic symbols for magnetic cards may be found in Appendix A.

This application pac has been designed for both the HP-97 Programmable Printing Calculator and the HP-67 Programmable Pocket Calculator. The most significant difference between the HP-67 and the HP-97 calculators is the printing capability of the HP-97. The two calculators also differ in a few minor ways.

Most of the computed results in this pac are output by PRINT statements: either by the statement PRINTx or by the command PRINT STACK. On the HP-97 these results will be output on the printer. On the HP-67 each PRINT command will be interpreted as a PAUSE: the program will halt, display the result for up to five seconds, then continue execution. The term "PRINT/PAUSE" is used to describe this output condition.

The lists of keystrokes required to solve example problems indicate the resulting outputs. Those outputs indicated by *** are printed by the HP-97 with the printer in MANUAL mode. These *** outputs are shown by PAUSE on the HP-67. Outputs without stars are displayed on both the HP-97 and HP-67.

If you own an HP-67, you may want more time to copy down the number displayed by a PRINT/PAUSE. All you need to do is press any key on the keyboard. If the command being executed is PRINTx (eight rapid blinks of the decimal point), pressing a key will cause the program to halt. If the command being executed is PRINT STACK (two slow blinks of the decimal per value), the number in the display will remain there until the depressed key is released;

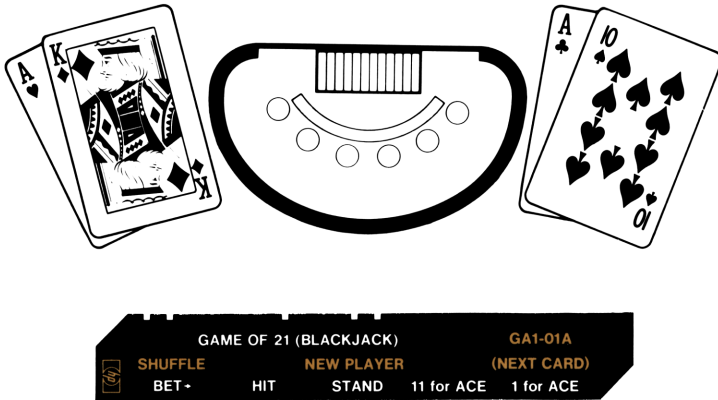
then the next register in the stack will be displayed, and so on. After display of all four registers, the program will halt execution if a key was pressed at any time during the display of the stack contents. In both cases execution of the halted program may be re-initiated by pressing **R/S**.

HP-97 users may also want to keep a permanent record of the values input to a certain program. A convenient way to do this is to set the Print Mode switch to NORMAL before running the program. In this mode all input values and their corresponding user-definable keys will be listed on the printer, thus providing a record of the entire operation of the program.

Another area that could reflect differences between the HP-67 and the HP-97 is in the keystroke solutions to example problems. It is sometimes necessary in these solutions to include operations that involve prefix keys, namely, **f** on the HP-97 and **f**, **g**, and **h** on the HP-67. For example, the operation 10^x is performed on the HP-97 as **f** 10^x and on the HP-67 as **g** 10^x . In such cases, the keystroke solution omits the prefix key and indicates only the operation (as here, 10^x). As you work through the example problems, take care to press appropriate prefix keys (if any) for your calculator.

If you have already worked through a few programs in the Standard Pac, you will understand how to load a program and how to interpret the User Instructions form. If these procedures are not clear to you, take a few minutes to review the sections, Loading a Program and Format of User Instructions, in your Standard Pac.

Game of 21 (Blackjack)



You make your bet, and the calculator, as dealer, deals two cards to you and two to itself. You see the dealer's first card face up, then his second face down (the calculator shows 0). Your two cards are then shown face up, one at a time. Next, you see a number in the form XXX.YY, where XXX is your bet and YY is the number of points in your hand.

You and the dealer may draw additional cards. Your goal is to finish with a hand whose total count is 21 or below, but closer to 21 than the dealer's. If your hand totals over 21, you lose (you're "busted").

The King is indicated by 13, the Queen by 12, and the Jack by 11, but all count 10 points each. The other cards always count their face values except the Ace. The Ace counts 1 unless you decide to change it to an 11 (press **D**). (If you have chosen 11 for an Ace, and want it counted 1, press **E**). If the next card you draw makes your score over 21, the calculator will automatically check for an Ace and make it count 1 if you have chosen 11 for its value. Note that an Ace always counts 1 for the dealer, except for a blackjack. A blackjack is a 2 card hand totalling 21, made up of a 10, Jack, Queen, or King plus an Ace with a value of 11. Your best win is a blackjack, since you win 1.5 times your bet rather than the bet itself. If both you and the dealer get blackjack, you neither win nor lose, it's a "push."

After the initial deal, and provided neither you nor the dealer have blackjack, you may ask the dealer to give you another card ("hit") by pressing **B**. If you don't want to draw, you may "stand" by pressing **C**. The calculator will then show the dealer's two cards. If the dealer's (calculator's) hand counts 16 or less, it draws. It continues to draw until its hand totals 17 or more, then it stands.

The calculator then determines if you've won or lost, and blinks your winnings (XXX) or losses (-XXX) and your final score (YY) in the form XXX.YY.

Next it shows your total account (the sum of your winnings and losses for all the games you’ve played this session).

Terms for HP-67/97 Game of 21 (Blackjack)

- 1. BLACKJACK: Any Ace (1) with 10, Jack (11), Queen (12) or King (13).
- 2. BUST: When your points are 22 or more.
- 3. HIT: Signify that you wish another card by pressing **B**.
- 4. STAND: You wish no more cards. Press **C**.
- 5. PUSH: You have the same points as HP-67/97. Blinking 0.000000000 when both have blackjack; blinking 0.00 for ordinary push.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load sides 1 and 2.			
2	Shuffle cards.		I A	
3	Stop shuffle and cut deck.		R/S	Ignore output
4	Input your bet (whole <i>even</i>			
	dollars only—no cents nor odd			
	dollars). Please do not bet over			
	\$100,000,000.	Bet	A	Dealer's
				cards;
				your cards;
				Bet.points
5	Hit (draw a card).		B	Bet.points
6	Repeat step 5 until you are			
	busted or you want to stand.			
7	If you want your Ace scored			
	as 11:		D	Bet.points
8	If you want your Ace changed			
	from 11 to 1:		E	Bet.points
9	Stand.		C	Win or loss
10	If you had not chosen to stand,			
	what would have been your			
	next card?		I E	Card

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
11	For a new game, go to step 2.			
12	For a new player: Reset			
	account to zero,		f C	0.00
	and go to step 11.			

Example:

Load sides 1 and 2.

Keystrokes:

f **A** →

Wait 20 or 30 seconds.

R/S →

To reproduce the example below,
store .9103987 in register **E** .

100 **A** →

B →

C →

Outputs:

Display will not stabilize until **R/S** is pressed.

Ignore output.

9. *** Dealer's 1st card
0. *** Dealer's 2nd card
(face down)
12. *** Your 1st card is a
Queen, worth
10 points.
7. *** Your 2nd card
- 100.17 100 is your bet,
17 is your score.
4. *** Your 3rd card
- 100.21 Bet. Score
9. *** Dealer's 1st card
7. *** Dealer's 2nd card
12. *** Dealer's 3rd card

100 **A** →

D →

Your Ace is now counted as 11.
You decide to stand.

C →

100.21 Your win. score
Display blinks to
indicate end of
game. Positive
number means you
win.

100.00 Your account

6. *** Dealer's 1st card

0. *** Dealer's 2nd card

1. *** Your 1st card

7. *** Your 2nd card

100.08 Your bet. score

100.18 Your bet. score

6. *** Dealer's 1st card

12. *** Dealer's 2nd card

4. *** Dealer's 3rd card

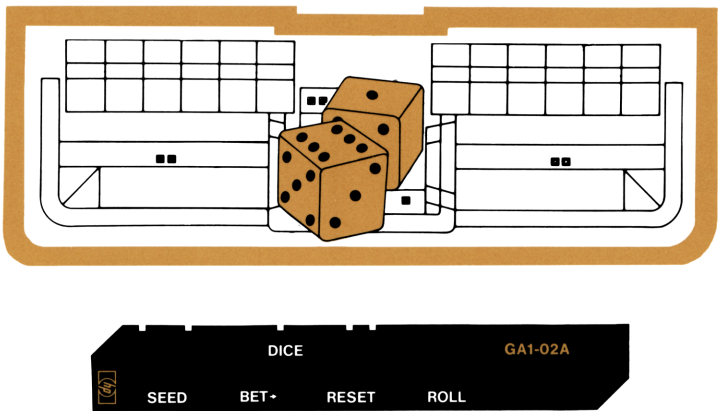
-100.18 Your loss. score

0.00 Your account

Dealer's 20 beats your 18, so you lose the \$100 you won the first game.
Care to try your luck again?

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Dice



There are many games that can be played with dice. One of the most popular of these games is “CRAPS.”

Two dice are used. One player, by general consent, becomes the first “shooter.”

A bet is placed. The shooter then throws the dice. If on the first roll the total is a 7 or 11, this is called a “natural” and the shooter wins. If the throw is a 2, 3, or 12, it is a “crap” and all that is bet is lost.

If any other number appears, it is called a “point.” The shooter then continues to throw the dice until the point is matched, in which case all that is bet is won; but if a 7 appears first, all that is bet is lost. Another player then becomes the shooter.

To play craps using this program, a seed (any number between 0 and 100) is input to key **A**. Then a bet is placed (key **B**). The program will then display generated rolls of the dice until the shooter wins or loses. The shooter’s winnings are updated and appear on the display. If another player is to become the shooter, press key **C**.

Another feature of this program is as a dice roller. A seed (any number between 0 and 100) is input to key **A**. The roll of the dice is then generated by pressing key **D**. After each roll the result is displayed. This process can be repeated as many times as you like.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 only.			
2	Key in seed (any number between 0 and 100).		A	0.00
3	For dice roll, go to step 6.			
	For craps, go to step 4.			
	CRAPS			
4	Input bet, and roll dice until you win or lose.		B	Display*
5	For new player, reset; then go to step 4.		C	0.00
	DICE ROLL			
6	Roll dice.		D	x.y zz†
7	Repeat step 6 as often as you wish.			
	*Each roll of the dice is displayed in succession as x.y zz†. When the player wins or loses, his updated winnings (or losses) are then displayed. A minus sign is used for losses.			
	†x = value of first die.			
	y = value of second die.			
	zz = sum of both dice.			

02-03

Example 1:

Load sides 1 and 2.

Keystrokes:

Set seed.

9 **A** →

Place bet.

10 **B** →

Place bet.

10 **B** →

Place bet.

10 **B** →

Place bet.

10 **B** →

Place bet.

10 **B** →

Another shooter

C →

Outputs:

0.00

5.1 06
6.3 09
3.6 09
1.4 05
6.4 10
6.6 12
2.6 08
2.4 06
10.00

WIN!

3.2 05
3.2 05
20.00

WIN!

6.6 12
10.00

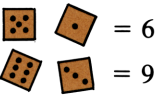
LOSE!

5.6 11
20.00

WIN!

1.4 05
3.3 06
4.5 09
6.6 12
3.1 04
5.2 07
10.00

LOSE!



Example 2:

Keystrokes:

Set seed.

1 **A** _____→

Roll dice.

D _____→

D _____→

D _____→

D _____→

D _____→

etc.

Outputs:

0.00

5.3 08

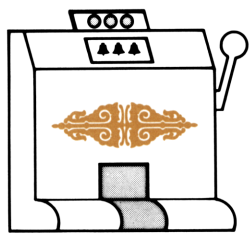
1.2 03

1.1 02

3.1 04

2.6 08

Slot Machine



This electronic slot machine deducts one dollar from your bank account with each “spin” and pays up to one hundred dollars for a jackpot. To begin, input a seed consisting of a decimal point followed by a string of digits using the **E** key.* Then merely press the **A** key time-after-time to spin the wheels. Your bank account may be seen at any time by pressing **B**.

Any combination of three digits may be seen in the display in the format 0.D₁D₂D₃. Only the following combinations, however, result in a payoff:

COMBINATIONS	PAYOFF
0.1XY	\$ 2.00
0.11X	\$ 5.00
0.ZZZ (Z ≥ 2)	\$ 10.00
0.ZZ0 (Z ≥ 2)	\$ 10.00
0.000	\$100.00

Good luck!

*The string of digits should be long and should contain an assortment of values.

Reference:

This program is based on an HP-65 Users’ Library program by Craig A. Pearce.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1.			
2	Enter seed (a many digit number between 0 and 1).	Seed	E	0.
3	Play.		A	Combination
	Winning combinations:			
	0.1XY \$ 2.00			
	0.11X \$ 5.00			
	0.ZZZ \$10.00			
	(where Z is 2 or more)			
	0.ZZ0 \$10.00			
	(where Z is 2 or more)			
	0.000 \$100.00			
4	Recall winnings or losses at any time (optional).		B	Winnings
5	Repeat step 3 any number of times.			
6	To start over, go to step 2.			

Example:

Load side 1.

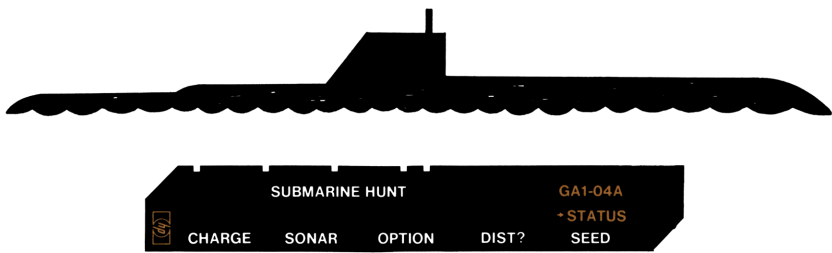
Keystrokes:

.963258741 **E** _____>
A _____>
A _____>
A _____>
B _____>
A _____>
B _____>
A _____>
A _____>
A _____>
A _____>
B _____>

Outputs:

0.
0.450
0.001
0.000 JACKPOT!
97.00
0.173 A \$2 WINNER
98.00
0.991
0.026
0.902
0.999 A \$10 WINNER
104.00

Submarine Hunt



Using your destroyer, you try to locate the position of the enemy submarine in a 10×10 grid, and then destroy it with a depth charge.

You input a seed (1–100) and the calculator will position the submarine in the center of one of the 100 squares (R, C), where R = row and C = column, and where R and C can each be 0, 1, 2, ..., 9.

You make guesses as to where you think the submarine is hiding by taking sonar readings. Input the location of your destroyer (R, C) and press **B**. If the submarine is in one of the 8 adjacent squares (or directly under your destroyer), the calculator will display “1.” Otherwise, a “0” will be shown.

When you think you’ve located the submarine, move your destroyer directly over it (move to the same square) and drop a depth charge. Blinking “1’s” indicate a hit, while a “0” shows a miss. If you miss, the submarine will move randomly to one of the 4 adjacent squares in the same row or column.

You can make the hunt easier or more difficult. For an easier game, press **D**. This increases the sensitivity of your sonar, allowing you to detect the submarine as far away as 2 squares in any direction (you cover a square region of the ocean 5 squares on a side). **D** is a toggle switch—you can switch from 1 to 2 square sensitivity or from 2 to 1 square sensitivity as often as you like during the game.

To make a more challenging game, press **C** immediately after inputting the seed. This allows the submarine to move after each sonar echo as well as after each depth charge miss. The submarine always moves randomly to an adjacent square in the same row or column.

A depth charge has a range of 0.9. When you position your destroyer for a depth charge drop, you may move anywhere on the board, not just to the center of a square. For instance, a depth charge dropped from a (2.5, 6.5) location would destroy any submarine in the center of squares (2, 6) (2, 7) (3, 6) and (3, 7).

Try to destroy the submarine using no more than 10 sonar readings and 1 depth charge, playing a regular game with regular sensitivity. You can check your status any time the display is steady by pressing **fE**.

Status format is XX.YY

where: XX = Number of depth charges fired.
 YY = Number of sonar readings.

Reference:

This program is based on an HP-65 Users' Library program written by Moshe Breiner.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load sides 1 and 2.			
2	Input seed (any number between 0 and 100).		E	0.
3	For regular game, go to step 5.			
4	Select difficult game (sub- marine always moving).		C	1.
5	To change sonar sensitivity: "2" means sensitivity distance is 2 squares. "1" means sensitivity distance is 1 square.		D	2. or 1.
6	SONAR	Row	ENTER	
	"0" means no echo.	Col.	B	0. or 1.
	"1" means echo received.			
	<i>or</i>			
	DEPTH CHARGE	Row	ENTER	
	"0" means miss.	Col.	A	0. or
	Blinking "1's" means HIT!			blink
7	Repeat step 6 until sub- marine is hit.			
8	To review status at any time: XX = number of depth charges fired. YY = number of sonar readings.		F E	XX.YY
9	For a new game, go to step 2.			

	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9											9											9
8											8											8
7											7											7
6											6											6
5											5											5
4											4											4
3											3											3
2											2											2
1											1											1
0											0											0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9											9											9
8											8											8
7											7											7
6											6											6
5											5											5
4											4											4
3											3											3
2											2											2
1											1											1
0											0											0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9											9											9
8											8											8
7											7											7
6											6											6
5											5											5
4											4											4
3											3											3
2											2											2
1											1											1
0											0											0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	

Playing boards for Submarine Hunt and Space War.
You might wish to use copies of this page for your games.

Example 1:

Load sides 1 and 2.

Keystrokes:

Outputs:

58 **E** →

0.

First move:

2 **ENTER** 1 **B** →

1. Echo

You now know your enemy is in one of the “x” squares below.

	0	1	2	3	4	5	6	7	8	9	
9											9
8											8
7											7
6											6
5											5
4											4
3	x	x	x								3
2	x	■	x								2
1	x	x	x								1
0											0
	0	1	2	3	4	5	6	7	8	9	

Diagram of 1st move

Second move:

1 **ENTER** 2 **B** →

0. No echo

The submarine cannot be in the ⊗ squares below.

	0	1	2	3	4	5	6	7	8	9	
9											9
8											8
7											7
6											6
5											5
4											4
3	x	x	x								3
2	x	⊗	⊗								2
1	x	⊗	⊗								1
0											0
	0	1	2	3	4	5	6	7	8	9	

Diagram of 2nd move

Third move:

3 **ENTER** 0 **B** → 0. No echo

You’ve narrowed down the submarine’s location to just 2 squares, those containing an “x” with no circle.

	0	1	2	3	4	5	6	7	8	9	
9											9
8											8
7											7
6											6
5											5
4											4
3	⊗	⊗	x								3
2	⊗	⊗	⊗								2
1	x	⊗	⊗								1
0											0
	0	1	2	3	4	5	6	7	8	9	

Diagram of 3rd move

Fourth move:

1 **ENTER** 0 **B** → 1. Echo

This eliminates (3, 2) as a submarine location, so you’ve found it!

	0	1	2	3	4	5	6	7	8	9	
9											9
8											8
7											7
6											6
5											5
4											4
3	⊗	⊗	⊗								3
2	⊗	⊗	⊗								2
1	⊠	⊗	⊗								1
0											0
	0	1	2	3	4	5	6	7	8	9	

Diagram of 4th move

Fifth move:

1 **ENTER** 0 **A** $\longrightarrow$ 0.111
0.11111
0.1111111
0.11111111
0.11111111 A hit!

Example 2:

Keystrokes:

Outputs:

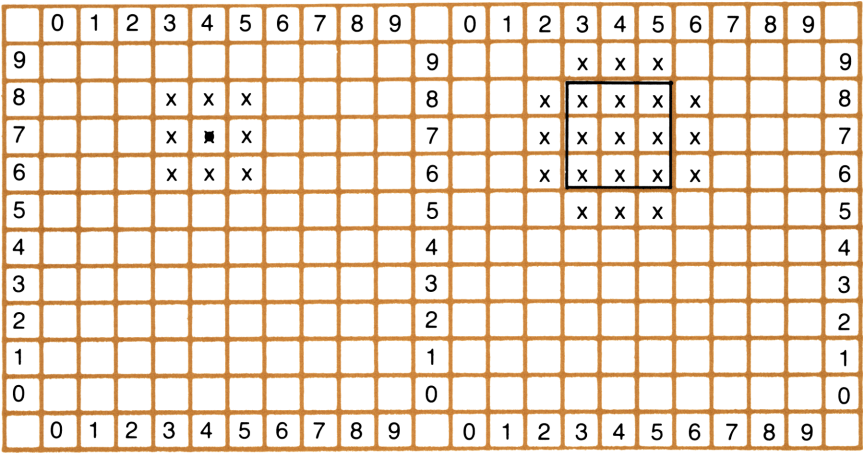
60 **E** $\longrightarrow$ 0.
C $\longrightarrow$ 1.

Submarine will now move on sonar echos as well as on depth charge misses.

First move:

7 **ENTER** 4 **B** $\longrightarrow$ 1. Echo

The submarine is in one of the “x” squares in the left diagram below. But the submarine moves, so now it could be in any of the “x” squares in the right diagram below.



Diagrams of 1st move

04-07

Second move:

8 **ENTER** 4 **B** → 0. No echo

You’ve eliminated some positions (left diagram: ⊗), but new possible positions have been created by the enemy’s random move (right diagram).

	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9				⊗	⊗	⊗					9			x				x				9
8			x	⊗	⊗	⊗	x				8		x	x	x		x	x	x			8
7			x	⊗	⊗	⊗	x				7		x	x	x	x	x	x	x			7
6			x	x	x	x	x				6		x	x	x	x	x	x	x			6
5				x	x	x					5			x	x	x	x	x				5
4											4				x	x	x					4
3											3											3
2											2											2
1											1											1
0											0											0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	

Diagrams of 2nd move

Third move:

7 **ENTER** 5 **B** → 1. Echo

This eliminates many possible positions (left diagram), but again, new ones are created (right diagram).

	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9			⊗				⊗				9						x	x				9
8		⊗	⊗	⊗		x	x	⊗			8					x	x	x	x			8
7		⊗	⊗	⊗	x	■	x	⊗			7			x	x	x	x	x	x			7
6		⊗	⊗	⊗	x	x	x	⊗			6			x	x	x	x	x	x			6
5			⊗	⊗	⊗	⊗	⊗				5					x	x	x				5
4				⊗	⊗	⊗					4											4
3											3											3
2											2											2
1											1											1
0											0											0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	

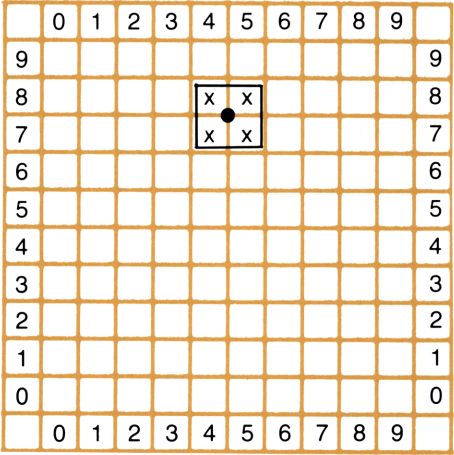
Diagrams of 3rd move

Fourth move: You try a depth charge.

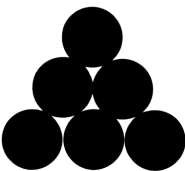
7.5 **ENTER** 4.5 **A** $\longrightarrow$ 0.111
0.11111
0.1111111
0.11111111
0.11111111 A hit!

It pays to be lucky.

The submarine used to be in one of these 4 squares:



Artillery



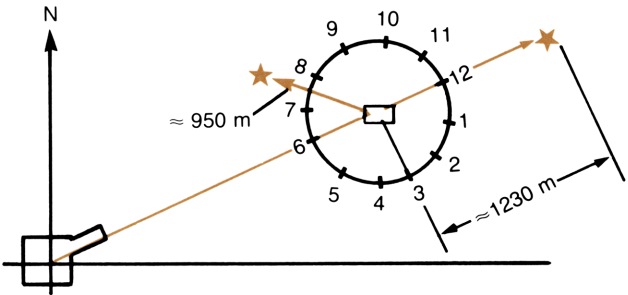
This program simulates the firing of an artillery round at a moving target whose initial position has been randomly selected. Feedback to the gunner is via a spotter plane weaving in and out of clouds over the battle area.

- (1) Initial display for each new battle is one of the 8 main compass directions: 0 (North), 45, 90 (East), 135, 180 (South), 225, 270 (West), or 315. The target lies in that *general* direction from the hidden gun and is 5 to 10 kilometers away.
- (2) The gunner fires by bearing and elevation parameters. A spotter (poor to perfect) relays information by displaying hh.DDDD :

hh (0 to 12) = the shell hit as an hour position on a relative clock face with the target at center and 6 o'clock in line with the gun.

DDDD = the *estimated* range from target to shell hit.

Thus: 8.0950 = shell was a bit short, left, and 950m away.
12.1230 = shell passed over target and hit 1230m beyond.



- (3) If shell lands within KILL range of target, the gunner wins. If not destroyed, and target has closed within 500 meters, target blasts gun to pieces—player loses!

Win is indicated by PRINT/PAUSE display of stack:

T = spotter rating

Z = maximum axial movement of target (SPEED)

Y = KILL range

X = proximity of shell hit to target

Loss is indicated by flashing 500.0.

- (4) Between one-minute shots, target randomly moves (up to maximum SPEED both N-S and E-W) but usually closes toward gun. Accurate information on where target *was* when last shell was fired can be obtained via **C** in format bbb.DDDD (true bearing and distance from gun).
- (5) After loading program, initialization (**I** **A**) sets the following parameters:
- Maximum gun range = 10,000 meters. (Thus, the formula for any shell's range = $10,000 \times \sin(2 \times \text{elevation})$.)
 - Target KILL range = 100 meters.
 - Target SPEED = 500 meters/minute (along each axis).
 - Spotter rating = 3.0 (1 = poor, 4 = perfect).
 - Random number seed = 0.5284163

Remarks:

Remember, information from a less than perfect spotter may be incorrect by as much as 3 hours in either circular direction and as much as $\pm 60\%$ of the "miss" distance. A perfect (4.0) spotter will report to the nearest hour and within 10 meters. Target movement along axes (SPEED), spotter rating (SPOTR), and lethal radius of a shell blast (KILL) may all be altered by the player even during a battle. Distinguished indeed will be the first player to win a battle under the following conditions: SPEED = 1000, SPOTR = 1.0, and KILL = 0!

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load sides 1 and 2 of card.			
2	Initialize parameters:		f A	10000.00
	(Random seed = 0.5284163)			
	(Axial movement of target,			
	SPEED = 500)			
	(Spotter rating, SPOTR = 3.0)			
	(KILL range = 100)			
	(Maximum gun range			
	= 10,000)			
3	Optional—			
	Set target SPEED.	Speed	f B	Speed
	Set spotter rating (1 = poor,			
	4 = perfect).	Spot	f C	Spot
	Set KILL range (1000 = easy,			
	10 = tough).	Kill	f D	Kill
4	Reset target for new battle.		A	BBB.*
5	FIRE!	Bearing	ENTER *	
		Elevation	E	
	or	or	f E	hh.DDDD*
6	Repeat step 5 until decision:			
	WIN—Stack is output.			
	LOSS—Flashing 500.0			
7	Display bearing & distance to			
	last target (Opt).		C	bbb.DDDD*
8	Recall rounds fired.		RCL I	Rounds

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
9	Go to step 3.			
	*Outputs:			
	BBB: Bearing in			
	degrees.			
	hh.DDDD:			
	hh = Estimate of hour			
	position of hit relative to			
	target. 6 is nearest gun.			
	DDDD = Estimate of dis-			
	tance from hit to target.			
	bbb.DDDD = Correct			
	bearing and distance to			
	last target position.			

Example 1:

Load sides 1 and 2.

Keystrokes:**Outputs:**

f **A** → 1000.00

(This initialization need be done only on a freshly loaded program. SPEED = 500, SPOTR = 3.0, and KILL = 100 meters.)

A (Starts the battle) → 180. (Approximate target bearing)

201 **ENTER** 33 **E** → 201.0000000 ***
 33.0 ***
 11.0130 ***

(Spotter has reported that round landed left, long and missed by 130m.)

201.5 **ENTER** 32 **E** → 201.5000000 ***
 32.0 ***
 2.0670 ***

05-05

199

ENTER

30

f

E

→

199.0000000

30.0

1.0570

196

ENTER

27

E

→

196.0000000

27.0

Target destroyed!

3.0

T Spotter rating

500.0

Z Target max.

movement

100.0

Y Kill range

79.1

X “Miss” distance

Example 2:

Select SPEED = 100, SPOTR = 4.0 (perfect), and KILL = 20.

100

f

B

→

100.0

4

f

C

→

4.0

20

f

D

→

20.0

A

→

315.

Approximate target bearing

315

ENTER

30

E

→

315.0000000

30.0

3.3230

295

ENTER

25

E

→

295.0000000

25.0

1.0530

293

ENTER

22.5

f

E

→

293.0000000

22.5

4.0050

292.5

ENTER

22.2

E

→

292.5000000

22.2

Gotcha!

4.0

T Spotter rating

100.0

Z Target max.

movement

20.0

Y Kill range

10.4

X “Miss” distance

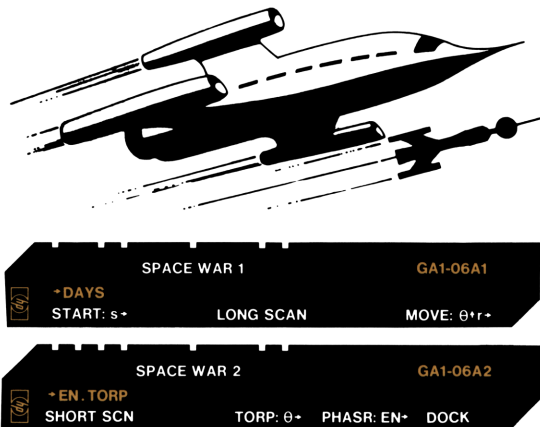
Example 3:

Select 1000 meter SPEED and KILL range.

1000.0	f	B	f	D	→	1000.0	
A					→	90.	
90	ENTER	30	E	→		90.0000000	***
						30.0	***
						1.2400	***
85	ENTER	20	E	→		85.0000000	***
						20.0	***
Ridiculously easy!						4.0	*** T
						1000.0	*** Z
						1000.0	*** Y
						881.0	*** X

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Space War



You are the commander of the Nuclear Powered Reconnoiterer (NPR) Kittyhawk. The NPR Kittyhawk is the sole guardian of justice in a vast galaxy that measures 10 quadrants by 10 quadrants. Within the galaxy somewhere, anchored among the blazing stars, lie three agents of evil: the vile Alglogs, known throughout space as interstellar thieves and creators of cosmic mischief. Your mission as commander of the NPR Kittyhawk is to search out and destroy the fearsome Alglogs within 18 stardays.

Also within the galaxy somewhere is a Base, a haven to which your ship may return in order to resupply itself. The weapons carried by the Kittyhawk are torpedos, which are fired in a straight line, and phasers, which send out an omnidirectional burst of energy. In addition, your starship is equipped with short- and long-range sensors which can detect the presence of Alglogs or the Base in nearby space. One starday is used whenever the Kittyhawk changes its position, i.e., when a move is made. Details of the operation of the NPR Kittyhawk are given below.

POWER ON (Card 1)

The Kittyhawk is started by supplying a seed s ($0 \leq s \leq 1$) to the routine START (Card 1). This routine positions the three Alglogs, the Base, and the Kittyhawk randomly in the galaxy. For best results, the seed s should contain all the digits but 0 and end in a 1, 3, 7, or 9. Remember that the galaxy is a 10×10 grid of quadrants; within each quadrant is a 10×10 grid of smaller areas called sectors. The quadrants are numbered 00 through 99, as are the sectors. The position of an object, then, may be specified by giving its quadrant and sector (QQ.SS). Examples of allowable positions are 23.68, 10.99, 7.01, and 85.00. No two objects may occupy the same position. At the end of the routine START, the calculator displays the starting position of the NPR Kittyhawk.

LONG-RANGE SCAN (Card 1)

The long-range scan covers all quadrants adjacent to and including that of the Kittyhawk itself. This scan will detect and report the presence of Alglogs or the Base in those quadrants. Suppose objects are located in the quadrants as below.

62	63	64	65
		B	A
52	53	54	55
A	KH		
42	43	44	45

The long-range scan would include the quadrants adjacent to quadrant 53 (Q53). The output of the scan would be three lines as follows:

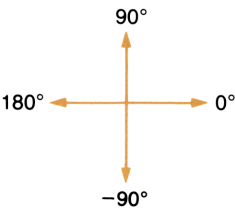
63.00400401
53.10400400
43.00400400

The first line shows the contents of quadrants 62, 63, and 64. Two digits are allocated to each quadrant, with the “4’s” merely indicating separation of the quadrants. The Base would appear in the right-hand of the two digits, an Alglog in the left-hand digit. Notice that the Base appears in the top line (Q64) and an Alglog in the second line (Q52). The Alglog in quadrant 65 is beyond the range of the sensors and does not show up. The numbers 63, 53, and 43 refer to the middle quadrant of each line.

The contents of nine squares are displayed with each long-range scan. If the Kittyhawk is at or near the edge of the galaxy, some of this information may be meaningless.

MOVE (Card 1)

If the Kittyhawk were in the position shown in the long-range scan above, a logical move would be to go to quadrant 52 to attack the Alglog. To make a move, one specifies the angle θ and distance r to be covered. Orientation of angles is shown in the diagram below. Angles must be input in degrees.



The distance is specified in terms of quadrants. To move exactly one quadrant's width, specify an r of 1. To move from Q53 to Q52, then, select $\theta = 180^\circ$ and $r = 1$. Suppose one wished to move from Q53 to Q64. This would require an angle of 45° and a distance of $\sqrt{2}$. The output at the end of the move routine is the Kittyhawk's new position.

Each move uses 1 starday. If a move is taken when no stardays remain, the display will flash zeros to indicate that the mission has failed.

Caution must be observed near the edges of the galaxy. Moving beyond an edge can result in the Kittyhawk's being lost in space.

SHORT-RANGE SCAN (Card 2)

The short-range scan gives a detailed picture of the quadrant the Kittyhawk is presently in. The output is 10 lines of information output by PRINTx commands, each line representing a row of the quadrant. The rows are output in the order 9, 8, 7, ..., 0. Each line consists of 10 digits that represent the ten sectors in the row. A "0" in a line means that that sector is unoccupied; a "3" marks the location of the Kittyhawk, a "4" an Alglog, and a "7" the Base. Suppose the output of a short-range scan were as shown below:

Row 9	0.000000000
Row 8	0.040000000
Row 7	0.000000000
Row 6	0.000000000
Row 5	0.000000003
Row 4	0.000000000
Row 3	0.000000000
Row 2	7.000000000
Row 1	0.000000000
Row 0	0.000400000

This scan indicates the presence of Alglogs in sectors 04 and 82, the Kittyhawk in sector 59, and the Base in sector 20.

TORPEDO (Card 2)

The Kittyhawk begins its mission with 3 torpedos. A torpedo may be fired at an Alglog within the same quadrant. If the torpedo passes within 1° of the Alglog, the Alglog is destroyed and the torpedo is spent. To fire a torpedo, simply specify the angle of fire in degrees.

If no torpedos remain and you attempt to fire a torpedo, the display will show "Error."

PHASERS (Card 2)

At the start of the mission, 1000 units of energy are available for firing phasers. Unlike torpedos, phasers fire equally in all directions and can destroy as many Alglogs as are within range. Only Alglogs within the same quadrant as the Kittyhawk may be fired on. The closer the Alglog, the less energy is required to destroy it. A minimum of 105 units and a maximum of 275 units may be needed to destroy an Alglog. To fire phasers, simply specify the amount of energy to be used.

Whenever an Alglog is fired on with phasers, there is a danger that the fire will be returned. Accordingly, the Kittyhawk uses another 100 units of energy to maintain shields against each Alglog within the quadrant.

If more energy is needed than is available, flashing zeros will be displayed to indicate that the mission has failed.

DOCK (Card 2)

The Kittyhawk may dock at the Base by moving into a sector adjacent to that of the base and executing the routine DOCK. If the docking is successful, the Kittyhawk's supply of torpedos and energy are replenished to their initial level: 3 torpedos, 1000 units of energy. The display at the end of this routine shows the current supply of energy and torpedos.

STATUS

Two routines are available for providing information on the status of the mission. Either of these routines may be executed at any time.

The first, on card 1, shows the number of days remaining in the mission. Simply press **f** **A** and the number of days will be displayed.

The second, on card 2, shows the remaining energy and torpedos. Both values are output in a single display as Energy.Torpedos. For example, a supply of 500 energy units and 2 torpedos would be displayed as 500.2. This information is available by pressing **f** **A** on card 2.

MAP OF GALAXY

A map of the galaxy (playing board) is located on page 04-03 in the game of Submarine Hunt.

Reference:

This program is based on an HP-65 Users' Library program written by Lee Gregory, Jr.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of			
	<i>Space War 1.</i>			
2	To initialize, key in a seed s			
	($0 \leq s \leq 1$). Output is the			
	starting position of Kittyhawk.	s	A	QQ.SS
3	You have 18 stardays. Your			
	options are outlined below.			
	Be sure the appropriate card			
	is loaded for each option.			
	LONG RANGE SCAN (Card 1)			
4	Output 3 lines of scan. KH is			
	in center. QQ refers to middle			
	quadrant of each line. Digits			
	AB refer to Alglogs and Base.		C	QQ.AB4AB4AB
	MOVE (Card 1)			
5	Key in direction and distance			
	(in quadrants) of desired			
	move; output new position of			
	Kittyhawk. (Flashing zeros			
	means all stardays used;			
	mission failed.)	θ	ENTER*	
		r	E	QQ.SS
	SHORT RANGE SCAN (Card 2)			
6	Output 10 rows of present			
	quadrant in order 9, 8, ..., 0.			
	Kittyhawk = 3, Alglog = 4,			
	Base = 7.		A	X.XXXXXXXXXX
	TORPEDO (Card 2)			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
7	To fire a torpedo, key in angle			
	or fire; output number of			
	Alglogs remaining. ("Error "			
	means no torpedos remain;			
	use phasers.)	θ	C	# Alglogs
	PHASER (Card 2)			
8	To fire phasers, key in units			
	of energy expended in fire;			
	output number of Alglogs			
	remaining. (Flashing zeros			
	mean all energy used; mis-			
	ion failed.)	Energy	D	# Alglogs
	DOCK (Card 2)			
9	You may dock from any			
	square adjacent to Base.			
	Output is present			
	Energy.Torpedos.		E	En.Torp
	STATUS			
10	With Card 1, display number			
	of days remaining.	(Card 1)	F A	Days
11	With card 2, display present			
	Energy. Torpedos.	(Card 2)	F A	En.Torp

Example:

From the log of the NPR Kittyhawk:

Load side 1 and side 2 of Space War 1.

Keystrokes:

.63154897 **A** →
C →

Outputs:

50.53 (KH position)
60.00400400 ***
50.00400400 *** (Long scan)
40.00400400 ***

Current map of galaxy:

	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9											9											9
8											8											8
7											7											7
6	●	●									6											6
5	KH	●									5				3							5
4	●	●									4											4
3											3											3
2											2											2
1											1											1
0											0											0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	

Quadrants

Sectors

Dots indicate quadrants known to contain neither Alglogs nor Base.

Move in direction -45° .

45

CHS

ENTER

5.5

E

C

14.62

(New position)

24.00400400

14.00400401

*** (Long scan)

4.00400410

Current map of galaxy:

	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9											9											9
8											8											8
7											7											7
6	●	●									6				3							6
5	●	●									5											5
4	●	●									4											4
3											3											3
2				●	●	●					2											2
1				●	KH	B					1											1
0				●	●	A					0											0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	

Quadrants

Sectors

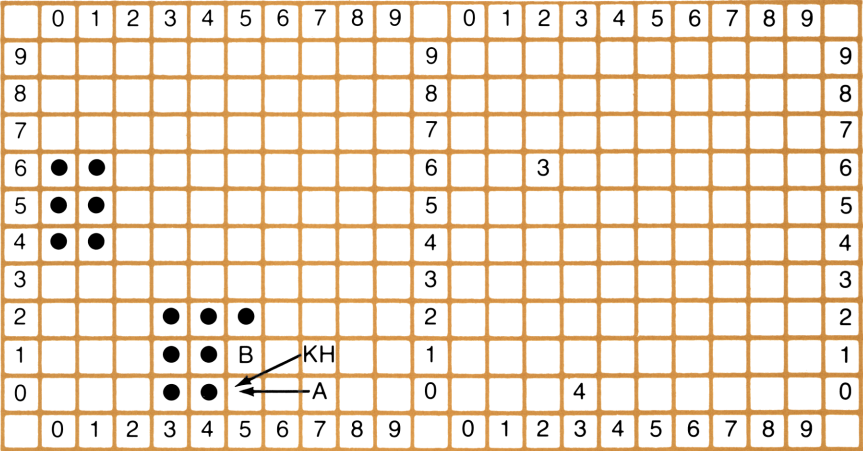
Move to Q05 to attack.

45 **CHS** **ENTER** **2** **√x** **E** —————→ 5.62 (Now in Q05)

Load side 1 and side 2 of Space War 2.

A —————→ 0.000000000 *** (Short scan)
0.000000000 ***
0.000000000 ***
0.030000000 *** (KH in S62)
0.000000000 ***
0.000000000 ***
0.000000000 ***
0.000000000 ***
0.000000000 ***
0.004000000 *** (Alglog in S03)

Current map of galaxy:



Quadrants

Sectors

Fire a torpedo in direction -80°.

80 **CHS** **C** —————→ 2.000000000 (2 Alglogs left)

Return to Base and dock.

Load side 1 and side 2 of SW1.

90 **ENTER** **1** **E** —————→ 15.62 (New position)

Load side 1 and side 2 of SW2.

A

0.000000000 ***

0.000000000 ***

0.000000000 ***

0.030000007 *** (KH in S62, Base in S69)

0.000000000 ***

0.000000000 ***

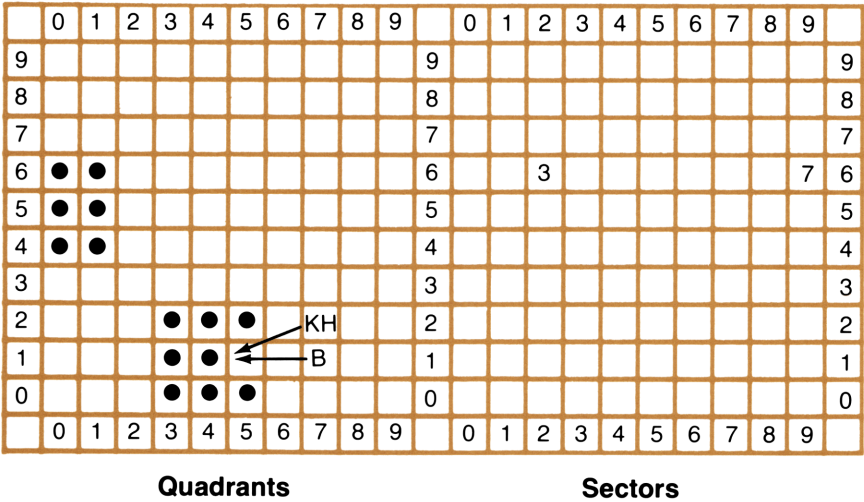
0.000000000 ***

0.000000000 ***

0.000000000 ***

0.000000000 ***

Current map of galaxy:



Move to S68.

Load side 1 and side 2 of SW1.

0

ENTER

.6

E

15.68

C →

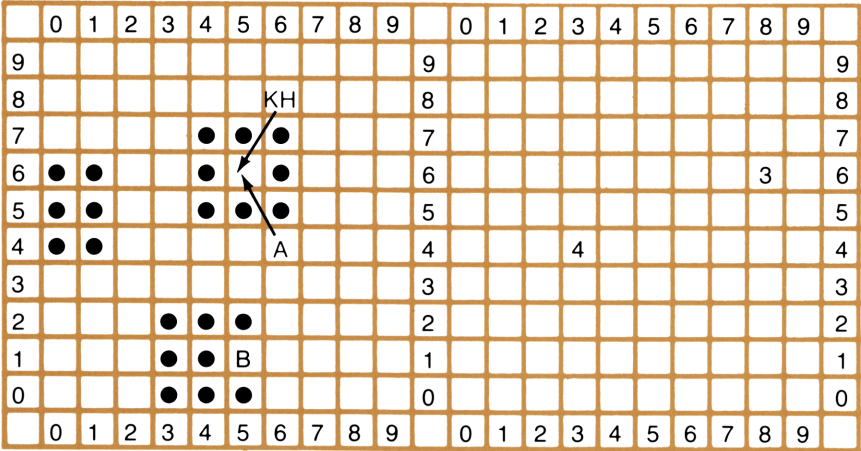
75.00400400 ***
65.00410400 *** (Note Alglog in Q65)
55.00400400 ***

Load side 1 and side 2 of SW2.

A →

0.000000000 ***
0.000000000 ***
0.000000000 ***
0.000000030 *** (KH in S68)
0.000000000 ***
0.004000000 *** (Alglog in S43)
0.000000000 ***
0.000000000 ***
0.000000000 ***
0.000000000 ***

Current map of galaxy:



Quadrants

Sectors

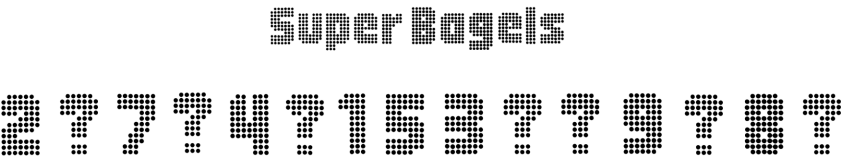
Use phasers. Try 200 energy units.

200 D → 1.000000000 (1 Alglog left)

The rest of the mission will be left as an exercise for the cadet.

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Notes



The object of the game of Super Bagels is for the player to guess an integer number which the calculator has chosen. Clues are given after each guess to tell the player how close his guess is to the hidden number. To make the game more interesting, the hidden number can be specified by the user to be from 1 to 8 digits, where each digit can range from zero to a maximum specified by the user. When the proper number is finally entered, the number of guesses required to discover the hidden number is displayed.

Play begins by first keying in a seed (any number) and pressing **A**. The display will return with a 2-digit code as shown: 4.5. This output shows the type of number which the machine will pick for the player to guess unless requested otherwise. The 4.5 game is a game of moderate difficulty where the number is a 4 digit number with each digit having any value from 0 to 5. Thus the minimum possible number is 0000, and the maximum 5555.

After the game has been started by pressing **A**, the number to be guessed can be changed by pressing **B** or **C**. To change the number of digits in the hidden number, key in the number of digits and press **B**. The number of digits must be at least 1 and not greater than 8. To change the maximum digit found in the hidden number, the maximum digit is keyed in and **C** is pressed. The range of legal maximum digits is from 0 to 9. After **B** or **C** is pressed, the display returns with a display in the same format as after pressing **A**. This makes sure the user knows the type of number he is trying to guess.

Once the hidden number has been specified, the game begins with the player entering his first guess and pressing **E**. The returned output is of the form CW.Guess where C is the number of digits of the guess that exactly match digits in the hidden number both in value and location. W is the number of digits of the guess that match digits of the hidden number in value, but not in location. Digits are not counted twice; that is, digits counted as C digits are not counted again as W digits. If C is 0 there will be a blank in place of C.

For example, if the hidden number is 12251, a guess of 12345 would yield 21.12345, meaning that 2 numbers (the 1 and 2) match exactly the hidden number, but that 1 number (5) is out of place.

A guess of 21125 would give 5.21125 meaning all of the digits of the guess are in the hidden number, but none of them are in the right place.

When the guess finally matches the hidden number, the number of guesses used is output. At any time during the game, the current number of guesses taken can be displayed by pressing **D**.

To play again, enter a new seed and press **A**. Super Bagels is written to run as fast as possible, but when numbers with many digits are selected, the time to analyze a guess will be significantly slower than with a simpler number. For those who want to discover the hidden number in some manner other than by using the clues given, you are invited to try. Super Bagels has been written to hide the number as much as possible. Good Luck.

Reference:

This game was inspired by the popular game *Mastermind*.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	To start game, key in a seed			
	(any number); output is in the			
	form D.M.*	Seed	A	4.5
3	To change number of digits to			
	be guessed in the number,			
	key in the number ($1 \leq D \leq 8$).	D	B	D.M*
4	To change the maximum digit			
	found in the number to be			
	guessed, key in the number			
	($0 \leq M \leq 9$).	M	C	D.M *
5	To display current number of			
	guesses already taken		D	Guesses
6	Guess a number containing D			
	digits with the maximum			
	digit $\leq M$. The output returns			
	the guess and also codes C			
	and W which signify the			
	number of digits of the guess			
	correct and in the right location			
	(C), and the number of other			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	digits of the guess correct but			
	in the wrong location (W).	Guess	E	CW.Guess
7	Repeat step 6 until a match is			
	made between the guess and			
	the number stored in the			
	machine. The output shows the			
	number of guesses required			
	to make the match.			no. of guesses
8	To play again return to 2.			
	*D = number of digits in			
	hidden number.			
	M = maximum allowed value			
	of any digit in hidden			
	number.			

Example 1:

Play a game with 4 digits, each in the range 0-5.
Load sides 1 and 2.

Keystrokes:

1.23456987 **EEX** **CHS** 12 **A** →
3214 **E** →

3015 **E** →
5234 **E** →

5203 **E** →
5223 **E** →
5253 **E** →

Outputs:

4.5
11.3214 *** (1 right, 1 in right place)
2.3015 *** (2 right)
21.5234 *** (1 right, 2 in right place)
30.5203 *** (3 in right place)
30.5223 ***
40.5253 *** (Correct)!
6. *** (6 guesses)

Example 2:

The original game of Bagels uses a hidden number of 3 digits, each in the range of 0-9. Play a game of Bagels.

Keystrokes:

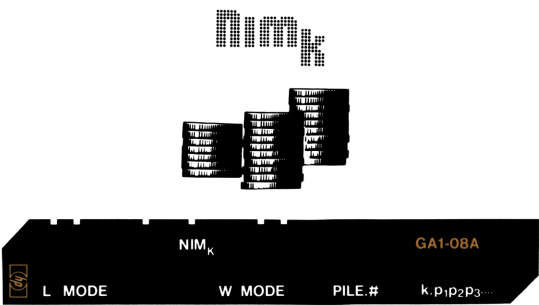
1.23456987 **A** →
 3 **B** →
 9 **C** →
 512 **E** →
 702 **E** →
 217 **E** →
 236 **E** →
 235 **E** →
 239 **E** →
 238 **E** →

Outputs:

4.5
 3.5
 3.9
 1.512 ***
 1.702 ***
 10.217 ***
 20.236 ***
 20.235 ***
 20.239 ***
 30.238 ***

7. *** (Number of
 guesses)

***Shown by PRINT on HP-97 and by PAUSE on HP-67.



According to Martin Gardner, one of the oldest and most engaging of all two person mathematical games is known today as Nim. The game, perhaps Chinese in origin, is played with counters (often pennies) arranged in piles, and two players alternate in removing one or more objects from one pile at a time. The player taking the last counter or counters wins or loses according to the mode of play.

Nim_k is a generalization of this Nim, first proposed by Prof. E. H. Moore of the University of Chicago in 1910. Nimb is an abbreviated version of Nim provided for the HP-65 and HP-25.

A number of objects or counters (from one to nine) is placed in a desired number of separate piles (from one to nine). You and the calculator take turns removing any number of counters (but at least one) from up to k piles. You choose whether the player taking the last counter wins (W mode) or loses (L mode).

The number k may be from one to eight. It is set at the beginning of the game and does not change during the game. You select both k and the initial arrangement of counters and piles.

To illustrate, choose k = 1 and let 2, 4, 5, and 6 counters be placed in pile numbers 1, 2, 3, 4 as shown below:

k = 1

Pile number:	1	2	3	4
Number of counters in each pile:	2	4	5	6

This will be shown in the calculator display as

1.2456

After the calculator's move, the display

1.2453

tells how many piles you may reduce and how many counters remain. The left-most digit keeps track of how many piles may yet be reduced.

You respond with a pile number, and the number of counters to be removed. Your response is displayed as:

3.1

That is, from pile 3 take 1 counter. The display then shows 0.2443, the zero indicating your move is finished (there are no more piles left from which to remove counters until your next move).

The moves continue back and forth until the last counter or counters are taken by the winner, or in the L mode of play, by the loser.

The game's complete mathematical analysis for $k = 1$ was first published by C.L. Bouton in 1901. In 1910 E.H. Moore described this generalization which he named Nim_k . As Moore described it, for n piles containing respectively

$c_1, c_2, \dots, c_n$ counters,

represent these numbers

c_i ($i = 1, \dots, n$)

in the binary scale of notation with

$c_{ij} \begin{pmatrix} i = 1, \dots, n \\ j = 0, 1, \dots \end{pmatrix}$ each 0 or 1

where $c_i = c_{i0} + c_{i1} 2^1 + c_{i2} 2^2 + \dots + c_{ij} 2^j$
($i = 1, 2, \dots, n$).

The combination is safe when

$$\sum_{i=1}^n c_{ij} = 0 \pmod{k+1} \quad (j = 0, 1, 2, \dots).$$

That is, for every place j the sum of the n digits c_{ij} ($i = 1, \dots, n$) is exactly divisible by $k + 1$. The L mode in which the player taking the last counter or counters loses, requires a change in strategy toward the end of the game. When the number of piles with two or more counters is from 1 to k inclusive, make

$$\sum_{i=1}^n c_{i0} = 1 \pmod{k+1} \text{ and } \sum_{i=1}^n c_{ij} = 0 \pmod{k+1} \quad (j = 1, 2, \dots).$$

This strategy is continued till the end of the game. A lengthy execution time is required to implement the above expressions.

The present game can be used as a teaching device to achieve an intuitive grasp of the strategy of play since at any time the play can be turned over to the calculator to reveal how it would play. For example, it is clear that if there are fewer than $k + 1$ piles, your opponent can win by taking all the counters. Such an arrangement is an "unsafe" combination (according to the

W mode of play). A “safe” combination would be $k + 1$ piles of one counter each. At least one counter must be taken by the calculator. That leaves k counters that you can take and win. If instead, k counters had been taken there would still be one left for you to take and win. Similarly, for all other options it is a safe situation for you.

References:

Bouton, Charles L., “Nim, a Game with a Complete Mathematical Theory.”, *Annals of Mathematics*, Series 2, Vol. 3, pages 35-39, 1901.

Moore, Eliakim H., “A Generalization of the Game called Nim.”, *Annals of Mathematics*, Series 2, Vol. 11, pages 93-94, 1910.

Redheffer, Raymond M., “A Machine for Playing Generalized Nim.”, B.S. Thesis in Mathematics, 1943, Massachusetts Institute of Technology.

Gardner, Martin, *Mathematical Puzzles & Diversions*, Simon and Schuster, 1959.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of card.			
2	Input choice of k followed by the number of objects in each pile.	$k.p_1p_2p_3...$	E	$k.p_1p_2p_3...$
3	Choose either W or L mode of play: W: player taking last object wins.		C	32.
	L: player taking last object loses.		A	-32.
	Pressing C or A also generates calculator's move.			$k.p_1p_2p_3...$
4	Input choice of pile and number of objects to be removed. m = the number of piles yet available to choose objects from this turn.	pile.#	D	pile.# $m.p_1p_2p_3...$

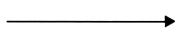
STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
5	Repeat step 4 until k piles have been chosen—(or less than k piles if desired) then go to step 3 for calculator's turn.			
6	Continue playing till the last object or objects are taken by the winner (or loser if in L mode).			k.000...
7	For another game, go to step 2.			
8	To learn good move habits, have the calculator play itself. After step 2, repeat step 3 until all piles are empty.			

Example 1:

Load sides 1 and 2.

Keystrokes:

7.478379895 **E**



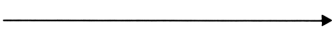
Outputs:

7.478379895 ***

You have chosen 9 piles, where counters from up to 7 piles may be removed each turn. The number of counters in each pile is:

Pile number:	1	2	3	4	5	6	7	8	9
Number of counters:	4	7	8	3	7	9	8	9	5

C



32. ***

You've chosen the W mode, where the player taking the last counter wins.

7.455155555 ***

After the calculator moves (unfortunately for you, it does a lot of thinking), it leaves the above pile arrangement .

08-05

You choose to take 4 counters from pile 2:

2.4 **D** → 2.4 ***
6.415155555 ***

You can still remove counters from 6 more piles this move. You elect to remove counters from all 6 as follows:

3.4 **D** → 3.4 ***
5.411155555 ***
5.4 **D** → 5.4 ***
4.411115555 ***
6.4 **D** → 6.4 ***
3.411115555 ***
7.4 **D** → 7.4 ***
2.411111555 ***
8.4 **D** → 8.4 ***
1.41111115 ***
9.4 **D** → 9.4 ***
0.41111111 ***

Now it's the calculator's turn.

C → 32. ***
7.01111111 ***

The handwriting is on the wall, but you gamely continue:

9.1 **D** → 9.1 ***
6.01111110 ***

It's all over but the last move:

C → 32. ***
7.00000000 ***

The calculator won this one, but maybe if you chose the L mode, you'd do better. Why don't you try?

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

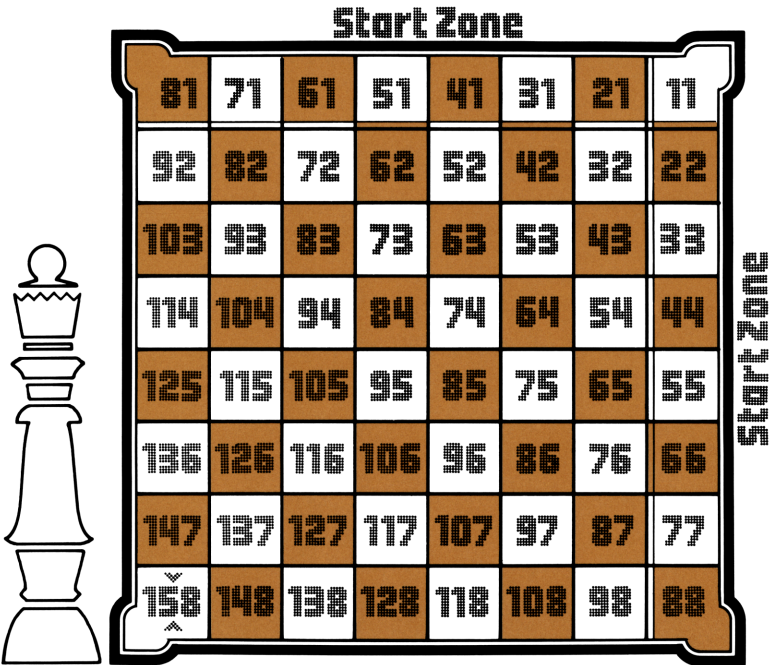
Notes

Queen Board



This game is based on the moves of a chess queen. A queen will be allowed to move only to the left, down, or diagonally to the left. The object of the game is to be the first player to move the queen to the lower left-hand corner of the chess board (square 158), by alternating moves between you and the calculator. You start by placing the queen on any square on the top row or right-hand column. This is your first move. The play then alternates.

The playing board is numbered as follows:



You tell the calculator your moves by keying in the number of the square you start on or move to. Press **A** and the calculator responds with the square it moves to. Square 158 is the winning square.

The program does not check for illegal moves. If you win (by moving to square 158), the program will respond with 168 (the calculator acknowledges the loss by displaying a nonexistent square).

The magnetic card was recorded in DSP 0 mode, so only integers will be displayed.

Reference:

This program is based on an HP-65 Users' Library program by Jacob R. Jacobs.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1.			
2	Key in your starting position			
	(first move).	Move	A	Calculator's
				move
3	Repeat step 2 until someone			
	wins.			

Example:

Load side 1.

Keystrokes:

Outputs:

55 **A** → 75.

(You start on 55, and the calculator, after deep and careful thought, moves to 75).

97 **A** → 127.

(You respond with 97, and the calculator, showing no mercy, moves to 127).

148 **A** → 158.

(You try 148, hoping the calculator's batteries run down before it can respond, but no luck—it wins by moving to 158).

Hexapawn



Hexapawn is a game which is programmed to learn from its mistakes. The game is played with chess pawns on a 3×3 board. Pawns may advance one square at a time or capture the opponent's pawns by moving diagonally one square. The game starts with the pawns positioned as follows:

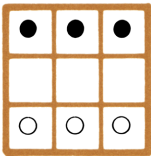


Figure 1. Starting position of pawns

The two allowed opening moves for the first player (in this example, white) are A and B:

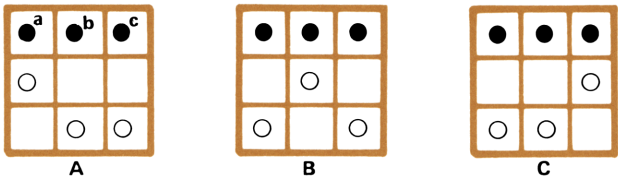


Figure 2. Opening moves

Since position C is a mirror image of A, it is not used. Black's three possible responses to white's A move are D, E and F.

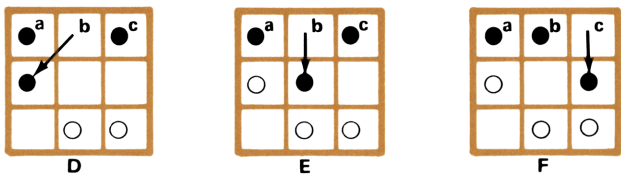


Figure 3. Black's responses to white's A move

Black can move diagonally and capture white (D), or he can move either **b** or **c** straight ahead one square. Black pawn at **a** is blocked. Note that the only way a pawn can move to an open square is straight ahead. Also, the only way a pawn can capture is by moving diagonally.

The game is won by advancing a pawn to the third row, capturing all of the opponent's pawns, or creating a position in which the opponent cannot move.

Moves are made by keying in a board position selected from the set of board positions shown in Figure 4 or Figure 5). The numbered arrows in each diagram indicate the possible choices the machine has for its replies. A reply of "0.00" indicates that the calculator has decided to forfeit the game. The machine selects its move at random, but whenever it is punished, it forgets the previous move. Thus, if the machine makes a poor move and is punished, it will not repeat the mistake.*

Since it is not easy for a human to visualize the changing game board, it is suggested that you follow your game on the playing board on page 10-03. You might use silver colored coins for white and pennies for black.

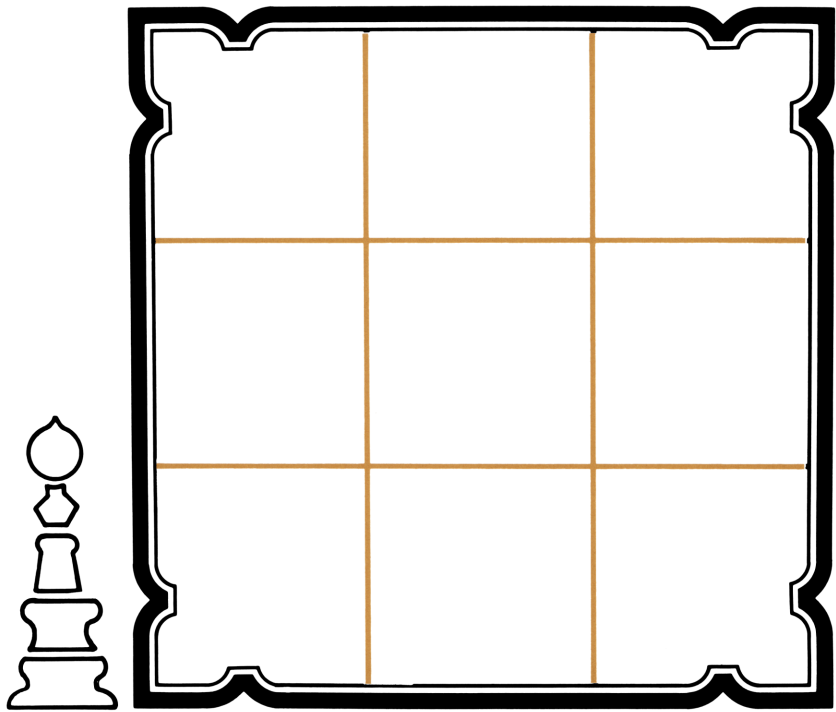
*Similarly, if you punish the machine for making good moves, it will eventually lose consistently—the calculator is your slave.

Reference:

This program was published in *65 Notes*, Vol. 2, No. 3. The HP-65 version was written by John R. Rausch.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Initialize:			
	Human moves first		f C	
	or Calculator moves first.		C	
	Then, after waiting a few			
	seconds, stop the random			
	number generator.		R/S	
3	Input the board position which			
	exists after your move.	Position	A	Response
4	Update your own board and			
	repeat step 3 until the game			
	is over.			
5	If the machine lost, punish it.		B	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
6	To play another game:			
	a) If the same player plays			
	first, go to step 3.			
	b) If the other player plays			
	first, go to step 2.			



Playing Board for Hexapawn

Board Positions and Responses

You move *first*:

You are white (○)

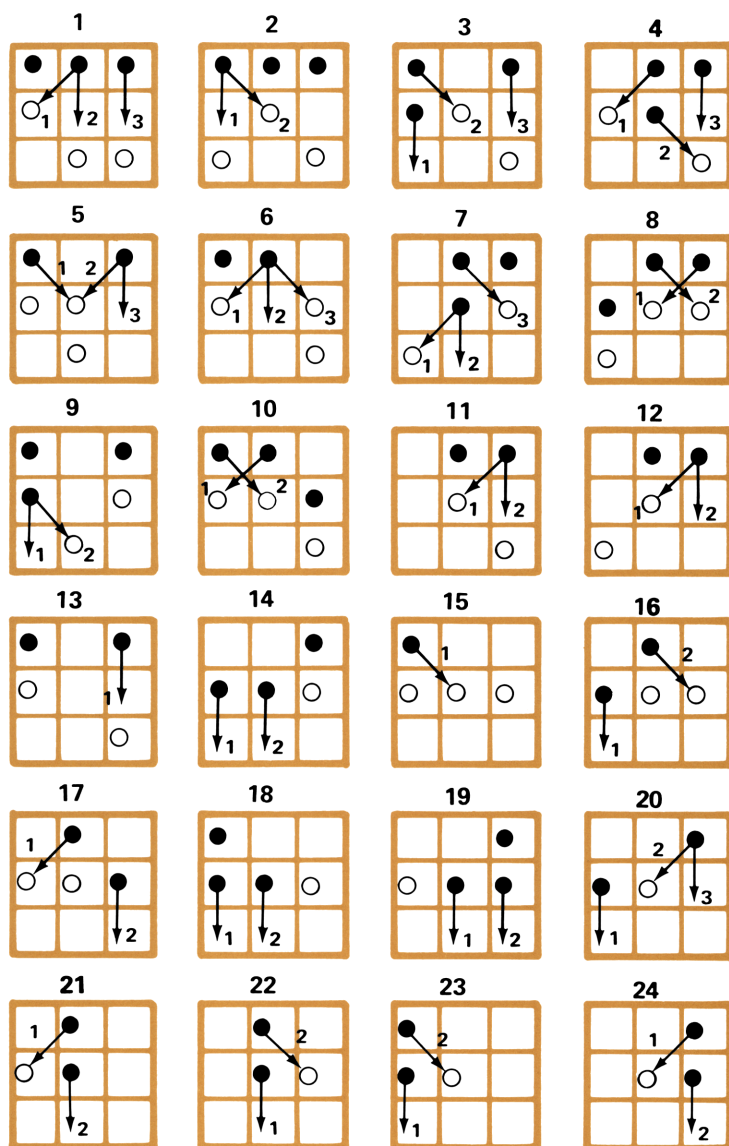


Figure 4

Board Positions and Responses

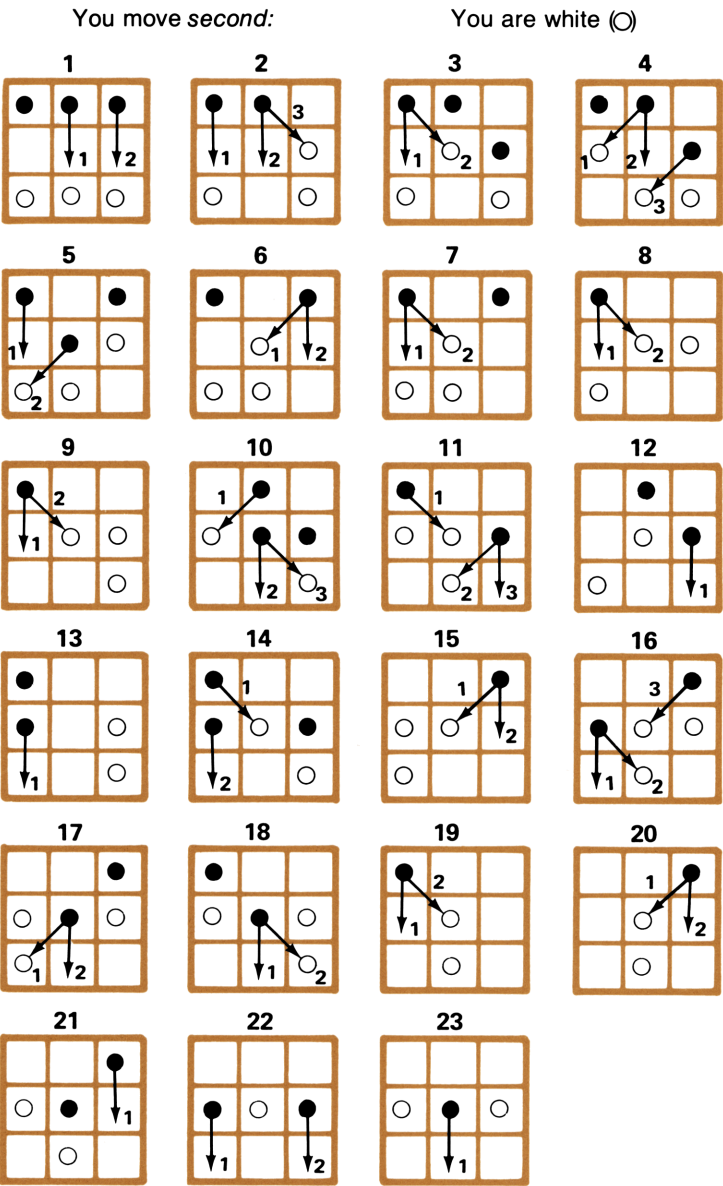


Figure 5

Example 1:

Load sides 1 and 2.

Keystrokes:

Outputs:

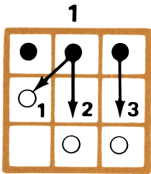
You move first.

f **C** (Wait a few seconds.)

R/S → Ignore display.

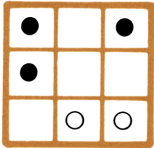
If you store .70879 in register 6, the game will proceed as follows:
You start by moving your left pawn, giving board position 1. The calculator responds with move 1.

1 **A** →



Board position 1

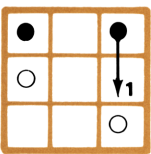
1.00



Board after calculator's response

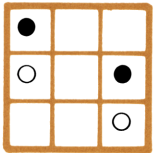
You capture, resulting in board position 13. In response, the calculator makes the only possible move.

13 **A** →



Board position 13

1.00



Board after calculator's response

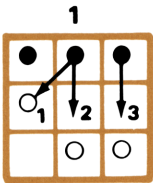
You can't move, so the calculator wins.

Example 2:

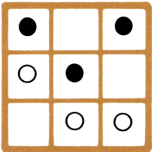
You start again by making the same opening move (board position 1). This time the calculator chooses move 2.

1 **A**

2.00

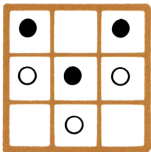


Board position 1



Board after calculator's response

You move your right pawn forward, making it impossible for the calculator to move. You win.



Your winning board position

Since the calculator's move was a foolish one, you punish it.

B Ignore output.

Now the calculator will not respond to board position 1 with move 2, unless you scramble its brain by removing the Hexapawn program from its memory and then reloading it.

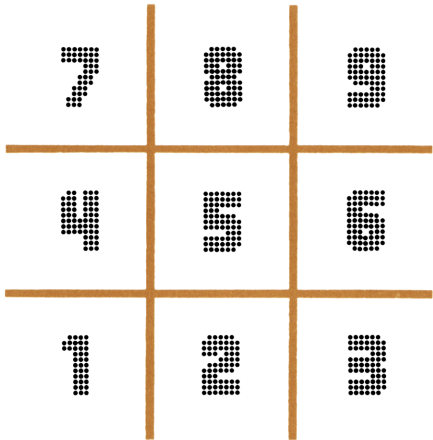
By continuing to punish the calculator for unwise moves, it will soon play without error. Then be prepared for some tough games.

Notes

Tic Tac Toe

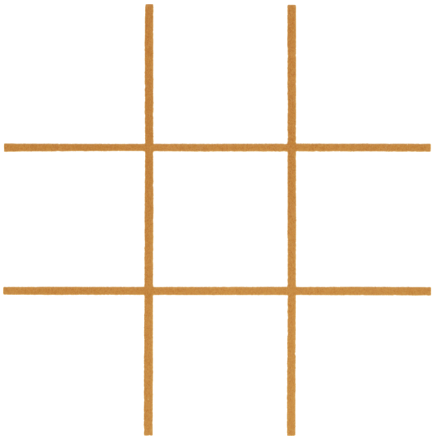


The game of tic-tac-toe hardly needs any introduction. In this one, you play versus the calculator. Moves are entered by keying in the appropriate position number selected from the diagram shown below. The calculator moves first at (2) so you will have a better chance to draw.



If you make any bad moves, the calculator will win; otherwise, the game will end in a tie.

You can use copies of the playing board below to follow your games.



Reference:

This program is based on an HP-65 Users' Library submittal by Delmer D. Hinrichs.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS									
1	Load side 1 and side 2.												
2	Initialize.		 	0.000000000									
3	To start a game			Board*									
4	Repeat step 5 until the end of the game.												
5	Your turn. Enter position number (see diagram above).	$1 \leq x \leq 9$											
	The calculator will move, and the resulting position will be output.			Board*									
6	To start a new game, go to step 3.												
	*For instance, this PRINT/PAUSE output: 4.121, 4.112, 4.212, 7.212112121 corresponds to this board position:												
	<table border="1" data-bbox="248 1066 344 1180"><tr><td>1</td><td>2</td><td>1</td></tr><tr><td>1</td><td>1</td><td>2</td></tr><tr><td>2</td><td>1</td><td>2</td></tr></table>	1	2	1	1	1	2	2	1	2			
1	2	1											
1	1	2											
2	1	2											
	Your moves are shown by 2's, the calculator's moves are shown by 1's. The 4. in each PAUSE display means you have finished your 4 th move.												
	The 7 means the calculator has just moved into position 7 (see diagram above).												

Example:

Load sides 1 and 2.

Keystrokes:

f **A** →

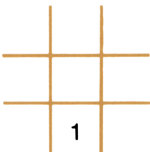
Outputs:

0.000000000

The calculator starts by making the move it will always make when a new game begins:

A →

0.000 ***
0.000 ***
0.010 ***
0.010000000

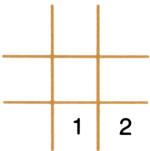


Board position 1A
Calculator's opening move

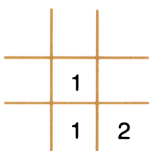
You put your “2” in square 3, and the calculator responds with square 5.

3 **R/S** →

1.000 ***
1.010 ***
1.012 ***
5.012010000



Board position 1B
You move to square 3



Board position 2A
The calculator moves to square 5

8 **R/S** →

2.021 ***

2.010 ***

2.012 ***

9.012010021

	2	
	1	
	1	2

Board position 2B
You move to square 8

	2	1
	1	
	1	2

Board position 3A
The calculator moves
to square 9

1 **R/S** →

3.021 ***

3.110 ***

3.212 ***

4.212110021

	2	1
	1	
2	1	2

Board position 3B
You move to square 1

	2	1
1	1	
2	1	2

Board position 4A
The calculator moves
to square 4

6 **R/S** →

4.121 ***

4.112 ***

4.212 ***

7.212112121

	2	1
1	1	2
2	1	2

Board position 4B
You move to square 6

1	2	1
1	1	2
2	1	2

Board position 5A
The calculator moves
to square 7

11-05

You've battled the calculator to a tie, which gives you a moral victory.

Flushed with success, you decide to play another game:

A $\longrightarrow$ 0.000 ***
0.000 ***
0.010 ***
0.010000000

5 R/S $\longrightarrow$ 1.000 ***
1.020 ***
1.110 ***
1.110020000

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

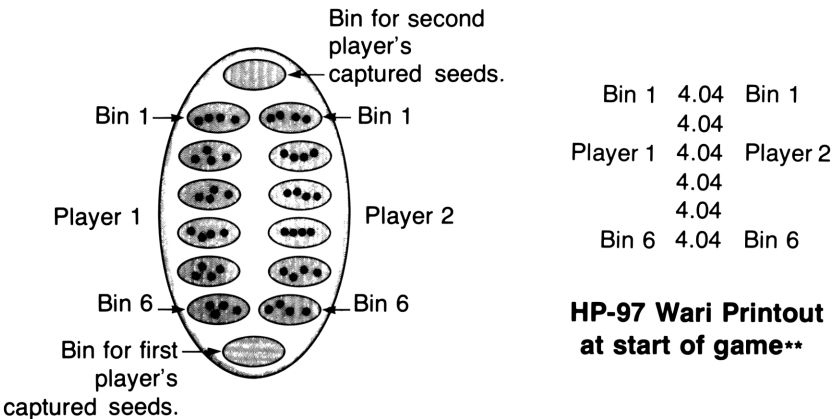
You might wish to continue this game. The calculator is a very good Tic-Tac-Toe player, so be careful!

Notes



Wari* is a board game which has been played for at least several centuries in various forms throughout Africa. The game is played on a board containing (generally) twelve small pits or bins, and two large pits. Forty-eight beads, seeds, or other counters are moved and captured according to certain rules.

The Wari board is shown here set up to begin a game.



**HP-97 Wari Printout
at start of game****

Wari Board at start of game

Each player in turn removes all the counters from one bin on his side and distributes them one-at-a-time into successive bins moving counterclockwise, skipping the two bins which are for storing captured counters. If the last counter drops into an opponent's hole containing one or two counters, the contents of that hole are captured and placed in the player's scoring pit. Counters in an unbroken sequence of two- and three-counter bins on the opponent's side clockwise from the captured bin are also captured. If a bin contains twelve counters or more, that bin is skipped when the counters from that bin are distributed.

The above rules are implemented in the calculator program. Special rules, such as prohibiting moves which remove all of the opponent's counters, were deemed to be variations of the basic game and were not programmed. It is possible to come to a situation where a few counters will circulate forever. In this case each player claims the counters on his side.

* Also known as Man-Kalah, Awari, and many other names.
**The HP-67 displays this information in PRINT/PAUSE mode.

To make a play on the calculator Wari board, the player specifies the bin he wants to move by keying in a number from 1 to 6 and then pushing either **A** or **E**. The machine then moves the counters from the specified bin according to the rules, prints the resulting position and displays the updated score. If you play this game on an HP-67, you must copy down the position as it is print/paused. To review the position, press **B**. To redisplay the score, press **D**.

An elementary strategy is also implemented in this program so that a player may match his wits against the machine. The machine is able to make obvious, though not brilliant, captures. The machine does not attempt to avoid having its counters captured, so it is a relatively poor opponent.

A special situation exists (covered by the Example) when one side (say side 1) has no counters. If player 1 is a human, he or she passes until consecutive moves by player 2 puts counters on side 1. If player 2 is the calculator, player 1 presses **f E** to cause the calculator to move.

If the calculator is playing the side that is temporarily without counters, move your counters in the usual way (Move **A**), and then press **R/S** anytime after the board is print/paused. Next, make your next move (Move **A**), and so on until the board shows a counter on the calculator's side. At that time, let the calculator make its move.

References:

Zaslavski, Claudia, *Africa Counts—Number and Pattern In African Culture*, Prindle, Weber & Schmidt, Inc., Boston, 1973.

Rules for Man-Kalah!, Skor-Mor Corp., 1970.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Start.		C	Board**, 0.00
3	Player 1: Select move (choose bin number from 1 to 6).	Move	A	Board, score
4	If 2 humans are playing, go to step 5.			
	If a human is playing the calculator, go to step 9.			
	**See text and example.			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
5	Player 2: Select move.	Move		Board, Score
6	If one player cannot move (no counters), he skips his turn and the other player plays again.			
7	Repeat steps 3, 5 and 6 until the game is finished.			
8	For another game, go to step 2.			
9	Human: Tell calculator to be your opponent, and instruct it to move automatically after each of your moves.			
	For HP-67:			
				Move,
				Board,
				Score
	For HP-97:			
				Move,
				Board,
				Score
10	Select move (choose bin number from 1 to 6).	Move		Board,
				Move,
				Board,
				Score

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
11	If you cannot move (no			
	counters), tell calculator to			
	move again.		I E	Board,
				Score
12	If the calculator cannot move			
	(no counters), stop calculator's			
	automatic move.		R/S	
	And move your counter(s)			
	again.	Move	A	
13	Repeat steps 10, 11 and 12			
	until the game is finished.			
14	For another human/calculator			
	game, press START,		C	
	and go to step 10.			
15	For a human/human game,			
	tell calculator politely that it is			
	no longer playing:			
	For HP-67		h SF	
			O	
	For HP-97		I STF	
			O	
	and go to step 2.			
16	To review position		B	Board
17	To review score		D	Score

Example:

You have decided to challenge the calculator.

Load sides 1 and 2. To reproduce this example, completely clear all registers.

Keystrokes:

C →

Outputs:

4.04 ***
4.04 ***
4.04 ***
4.04 ***
4.04 ***
4.04 ***

}

Starting board

0.00 Score

Move 1:

You choose to move your counters out of bin 2:

2 **A** →

4.04 ***
0.04 ***
5.04 ***
5.04 ***
5.04 ***
5.04 ***

}

Board after your
1st move

0.00 Score

For HP-67: **h** **CF** 0 **f** **E**
For HP-97: **f** **CLF** 0 **f** **E**

3.00 *** Calculators 1st move

5.05 ***
1.05 ***
5.00 ***
5.04 ***
5.04 ***
5.04 ***

}

Board after Cal's
1st move

0.00 Score

After you dropped each of your 4 counters from your bin 2 into your bins 3, 4, 5 and 6 (moving counterclockwise), the calculator decided to move its counters from its bin 3. Note that bins are numbered from top to bottom for both sides and that both sides move counterclockwise.

Move 2:

5 **A** →

5.05 ***
1.05 ***
5.01 ***
5.05 ***
0.05 ***
6.05 ***

}

Board after your
2nd move

4.00 *** Cal's 2nd move

6.06	***	} Board after Cal's 2 nd move
0.06	***	
5.02	***	
5.00	***	
0.05	***	
6.05	***	
0.02		Score

The calculator has struck the first blow. The 5 counters from its bin 4 went to its bins 3, 2, 1 and your bins 1 and 2. Since the last bin reached (your bin 2) contained only one lonely counter, both your counter and the attacking counter were removed, making the score 0 to 2, the calculator leading. If your bin 2 had contained 2 counters, they would have been captured, and the score would have been 0 to 3. Had your bin 2 contained 3 or more counters, you would have been safe.

You and the calculator continue moving as follows:

Move no.	Your move	Cal's move	Score
3	6	5	0-2
4	4	6	3-4

Here's the board after Cal's move 4:

8.09	***	
0.09	***	
5.05	***	
0.01	***	
1.02	***	
1.00	***	
3.04		Score

Move 5:

1 **A** 

0.09	***	} Board after your 5 th move
1.09	***	
6.05	***	
1.00	***	
2.00	***	
2.01	***	
6.00	***	Cal's 5 th move

0.09	***	} Board after Cal's 5 th move
1.09	***	
6.05	***	
1.00	***	
2.01	***	
2.00	***	
8.04		Score

You moved the 8 counters from your bin 1 around the board counterclockwise, ending up in Cal's bin 4. Looking clockwise from Cal's bin 4, note that Cal's bin 5 has only 2 counters. So you take the single counter in 4 (plus your counter) and the 2 counters in 5 (plus your counter), giving you a take of 5 counters. If Cal's bin 6 had contained 1 or 2 counters (before you added yours), you would have added another 2 or 3 to your score.

The game continues:

Move no.	Your move	Cal's move	Score
6	6	3	10-4
7	5	2	12-4
8	6	1	16-4
9	6	4	18-4
10	3	5	18-4
11	6	3	20-4
12	4	1	20-4
13	5	2	27-4
14	1	1	27-4
15	1	6	27-4
16	3	5	27-4
17	6	6	29-4
18	2	5	29-4
19	6	2	31-4
20	1	3	31-4
21	2	1	31-4
22	3	4	31-4
23	1	3	31-4
24	2	2	31-4
25	1	1	31-4

After Cal's 25th move, the board looks like this:

1.00 ***
2.00 ***
1.00 ***
5.00 ***
4.00 ***
0.00 ***
31.04 Score

} Board

Move 26:

3 **A** →

1.00 ***
2.00 ***
0.00 ***
6.00 ***
4.00 ***
0.00 ***

Since there are no counters on the calculator's side, it cannot move. You therefore stop its otherwise endless search for the nonexistent counter for it to move.

R/S →

Hold key until
display stabilizes.
Ignore output.

You then move again (5 **A**), and the game then proceeds normally. The calculator's succeeding moves depend on how long you wait before pressing **R/S**, since the program's random number generator runs during the calculator's search for a counter. Therefore, the rest of this example shows one of several possible outcomes.

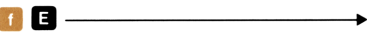
Move no.	Your move	Cal's move	Score
27	5	5	31-4
28	6	4	33-4
29	4	5	35-4
30	6	4	37-4
31	1	2	37-4
32	1	1	37-4
33	1	3	37-4
34	2	6	37-4
35	3	5	37-4
36	5	4	37-4
37	4	3	37-4
38	6	4	37-4
39	5	5	37-4
40	6	4	40-4

The board looks like this after Cal's half of the 40th move:

0.00 ***
0.02 ***
0.02 ***
0.00 ***
0.00 ***
0.00 ***

You have no counters and cannot move. So you ask the calculator to move again:

Move 41:



3.00 *** Cal's move
0.01 ***
0.03 ***
0.00 ***
0.00 ***
0.00 ***
0.00 ***

} Board after Cal's
41st move

Ruthless in victory, you seek even more victims:

Move no.	Your move	Cal's move	Score
42	No move	2	40-4
43	2	1	40-4
44	1	No move	40-4
45	2	No move	40-4
46	4	No move	40-4
47	3	No move	40-4
48	5	6	40-4
49	6	6	42-4

This is the final board following Cal's last move:

0.00 ***
0.00 ***
0.00 ***
1.00 ***
0.01 ***
0.00 ***

Since these two counters would chase each other forever, the game is called. Each remaining counter is counted for the side in whose bin it lies, making the final score:

Human: 43

Machine: 5

The machines have not taken over yet, but don't underestimate this opponent. It can give you trouble.

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Racetrack



This program is based on a pencil-and-paper game published in Martin Gardner’s “Mathematical Games” column in *Scientific American* (May, 1973). Up to five contestants race on a superelliptical track. Players take turns inputting acceleration (direction and magnitude) and the machine updates the velocity and position of each racer and checks for collision, shown by flashing the ID number(s) of the other racer(s) involved. Racers are considered to have collided if they approach within 2 units of each other. All racers involved in a collision are penalized by having their velocities reduced to zero. A racer that leaves the track or is off the track is similarly penalized. Thus, a racer leaving the track at high velocity might require several moves to get back on—so watch out!

The track is bounded by two superellipses:

outer boundary

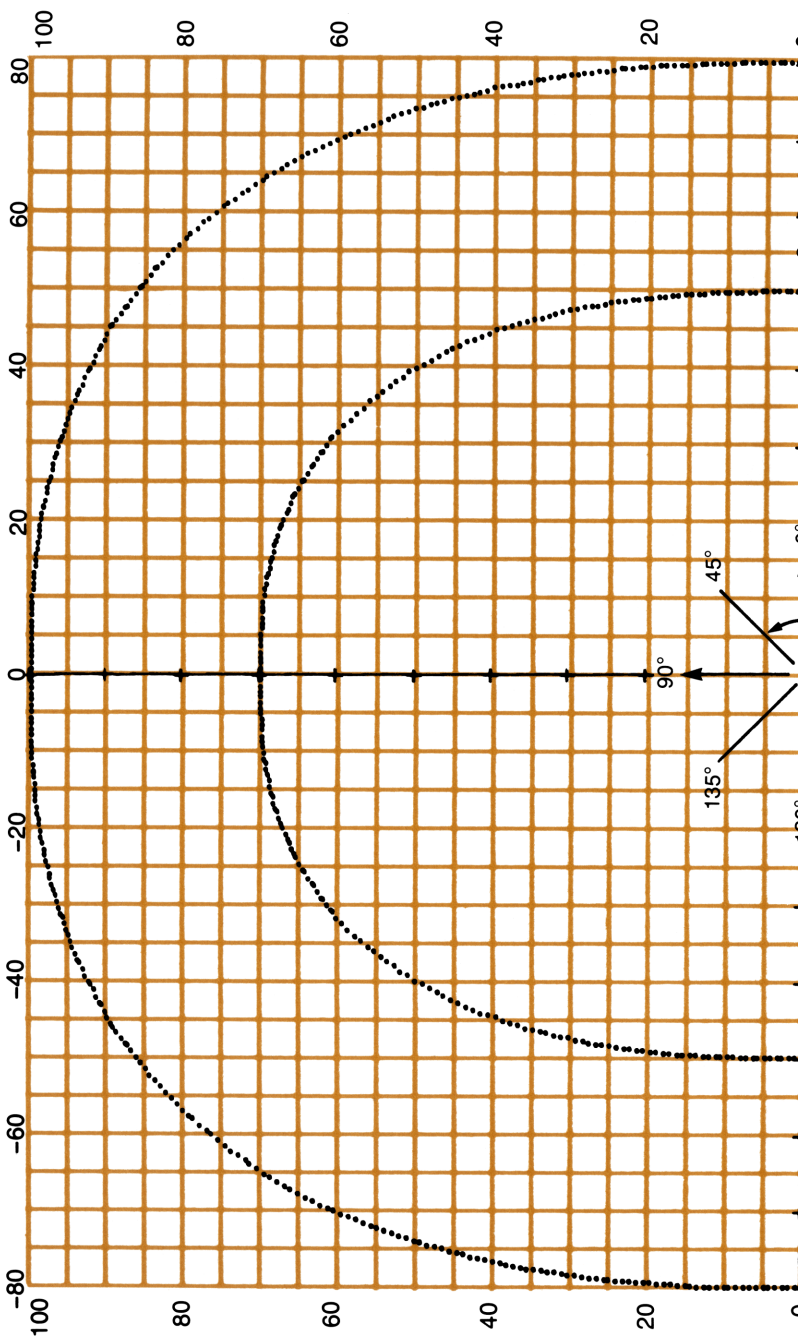
$$\left(\frac{x}{80}\right)^{5/2} + \left(\frac{y}{100}\right)^{5/2} = 1$$

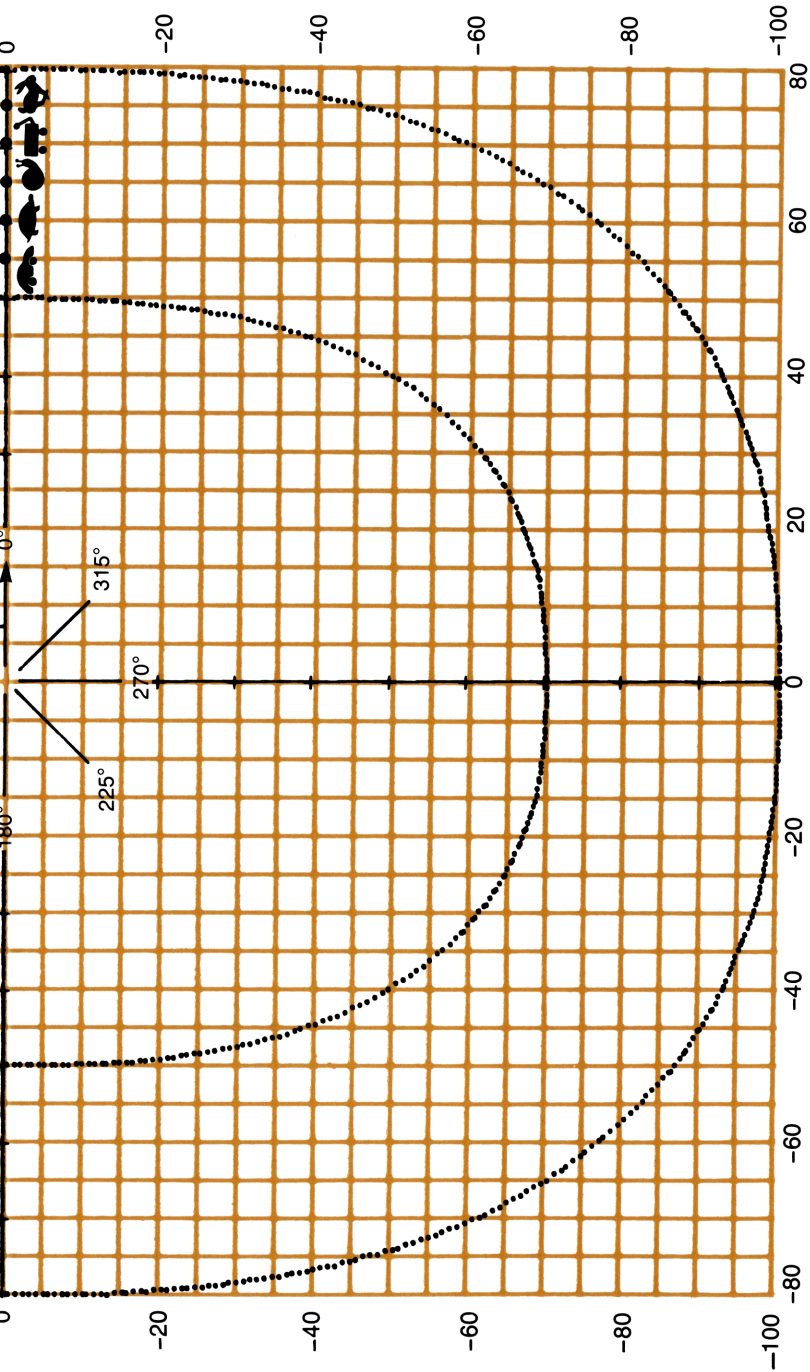
inner boundary

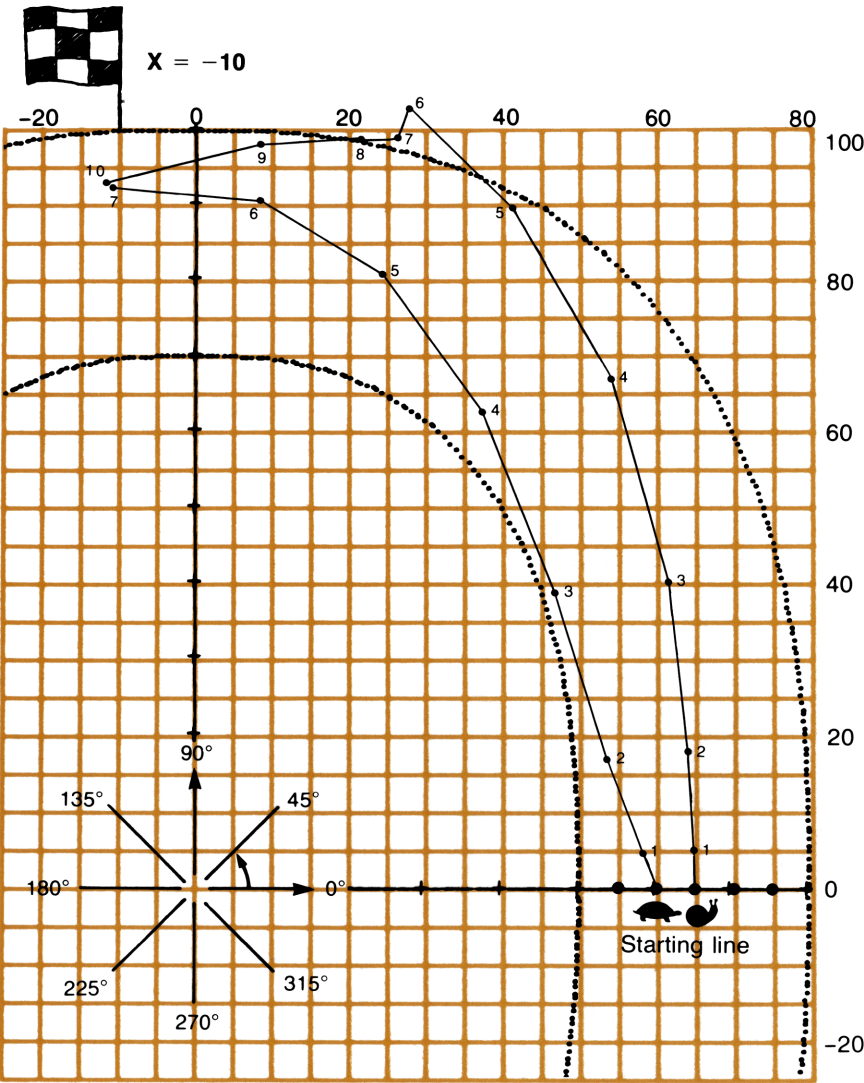
$$\left(\frac{x}{50}\right)^{5/2} + \left(\frac{y}{70}\right)^{5/2} = 1$$

At the start, the five racers are located as shown on the racetrack at the points (55, 0), (60, 0), (65, 0), (70, 0), and (75, 0). Moves are made by placing direction in the y-register, acceleration in the x-register, and pressing **A**, **B**, **C**, **D**, or **E** as appropriate for the desired racer. The direction convention used is shown on the racetrack. To simulate frictional effects, the maximum acceleration is 9 units per second per second.

It is convenient to keep track of the progress of the race by plotting the positions of the racers on a copy of the accompanying racetrack. By observing the changing positions of the racers, it is easier to determine what the next move should be.







Race between snail and turtle

Reference:

This program was first programmed for the HP-65 Users' Library by Delmer D. Hinrichs.

The superellipse is described in the following article:

Gardner, Martin, "The Superellipse: a Curve that lies between the Ellipse and the Rectangle," *Scientific American*, Sept. 1965, 222-234.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Start.		 C	Ignore output.
3	Move racers in turn.			
	Car: Input angle	θ	ENTER 	
	and acceleration.	a	A	1, stack*
	Turtle: Input angle	θ	ENTER 	
	and acceleration.	a	B	2, stack*
	Snail: Input angle	θ	ENTER 	
	and acceleration.	a	C	3, stack*
	Wagon: Input angle	θ	ENTER 	
	and acceleration.	a	D	4, stack*
	Rabbit: Input angle	θ	ENTER 	
	and acceleration.	a	E	5, stack*
4	To check the position of any			
	racer, input ID.	$1 \leq ID \leq 5$	 A	ID y, x
	*The contents of the the			
	stack are			
	angle of velocity T			
	velocity Z			
	y-coordinate Y			
	x-coordinate X			

Example:

A race is proposed between the turtle and the snail. They decide to race around the first turn to the finish line shown. By watching how they race, we might learn some tricks.

Load sides 1 and 2.

Keystrokes:

Initialize.

f **C** —————→

The racers start by accelerating in the directions 110 and 90 at the maximum acceleration.

110 **ENTER** **9** **B** —————→

90 **ENTER** **9** **C** —————→

Outputs:

-3.00 (Ignore this display.)

2.00 *** Turtle's I.D.
110.00 *** T Direction of Turtle
9.00 *** Z Speed of Turtle
4.32 *** Y y-position of Turtle
58.46 *** X x-position of Turtle
3.00 *** Snail's I.D.
90.00 *** T Direction of Snail
9.00 *** Z Speed of Snail
4.50 *** Y y-position of Snail
65.00 *** X x-position of Snail

Turtle now speeds on in the same direction and Snail begins a slow left turn.

110 **ENTER** **9** **B** —————→

100 **ENTER** **9** **C** —————→

2.00 *** Turtle's I.D.
110.00 *** Direction of Turtle
18.00 *** Speed of Turtle
16.91 *** y-position of Turtle
53.84 *** x-position of Turtle
3.00 *** Snail's I.D.
95.00 *** Direction of Snail
17.93 *** Speed of Snail
17.93 *** y-position of Snail
64.22 *** x-position of Snail

100 **ENTER** 9 **B** →

2.00 *** Turtle
 106.67 *** Direction
 26.91 *** Speed
 38.26 *** y
 46.91 *** x
 3.00 *** Snail
 96.67 *** Direction
 26.91 *** Speed
 40.23 *** y
 61.87 *** x

100 **ENTER** 9 **C** →

Turtle now begins to turn left a little more...

225 **ENTER** 5 **B** →

2.00 *** Turtle
 116.84 *** Direction
 24.93 *** Speed
 62.27 *** y
 37.42 *** x

... and so does Snail.

180 **ENTER** 9 **C** →

3.00 *** Snail
 114.40 *** Direction
 29.35 *** Speed
 66.95 *** y
 54.25 *** x

250 **ENTER** 9 **B** →

2.00 *** Turtle
 136.12 *** Direction
 19.89 *** Speed
 80.28 *** y
 24.63 *** x

260 **ENTER** 9 **C** →

3.00 *** Snail
 127.46 *** Direction
 22.50 *** Speed
 89.25 *** y
 41.34 *** x

250 **ENTER** 9 **B** →

2.00 *** Turtle
 162.99 *** Direction
 18.21 *** Speed
 89.84 *** y
 8.75 *** x

13-09

Snail realizes he is going too fast and he turns sharply left, but...

270 ENTER 9 C →

3.00 *** Snail
0.00 *** OOPS! Snail
0.00 *** crashes
102.61 *** through the fence.
27.65 ***

Turtle roars across the finish line.

220 ENTER 9 B →

2.00 *** Turtle
-178.92 *** Direction
24.31 *** Speed
92.28 *** y
-12.11 *** x is less than -10,
so Turtle has won.

Even though he has lost, Snail would like to try to get back on the track and finish the race.

250 ENTER 9 C →

3.00 *** Snail
0.00 *** Still off the track,
0.00 *** but closer
98.38 *** y
26.11 *** x

180 ENTER 9 C →

3.00 *** Snail
180.00 *** Hooray! Snail
9.00 *** made it back to the
98.38 *** racetrack.
21.61 ***

190 ENTER 9 C →

3.00 *** Snail
-175.00 *** Direction
17.93 *** Speed
97.60 *** y
8.18 *** x

Now Snail should be able to cross the finish line only three moves after Turtle.

250 **ENTER** 9 **C**  3.00 *** Snail
 2.00 (flashing) Snail
 crashed into Turtle!
 0.00 *** Direction
 0.00 *** Speed
 91.81 *** y
 -11.22 *** x

Well, folks, that's it from the HP racetrack: Snail came in second and Turtle was next to last.

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Teaser



The object of this game is to convert the pattern

0 0 0
0 1 0
0 0 0

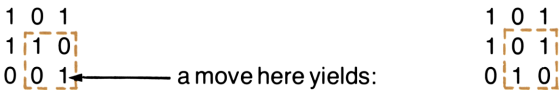
To the pattern

1 1 1
1 0 1
1 1 1

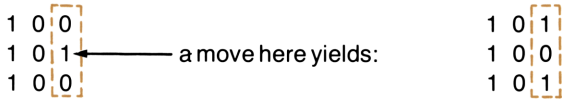


by changing 1's to 0's. The only legal move is changing a 1 to a 0. Consequently, the only allowable opening move is changing the 1 in the center of the board to a 0. When a 1 is changed to a 0, certain other 1's and 0's also change according to these rules:

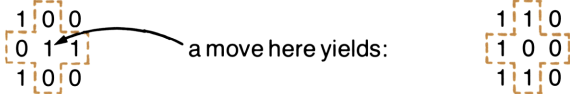
- 1. A move in a corner causes all 1's and 0's in a 2×2 box containing the corner to change state.



- 2. A move in the center of an edge causes all 1's and 0's on that edge to change state.



- 3. A move in the center causes all 1's and 0's in a "+" to change state.



References:

Nico, Willard I., "Shooting Stars," *Byte*, May, 1976, pp. 42-48.
People's Computer Center, *What To Do After You Hit Return*, People's Computer Company, Menlo Park, 1975, p.54.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Initialize.		C	Board*
3	Move.	$1 \leq \text{moves} \leq 9$	A	Board
4	Repeat step 3 until the pattern			
	1 1 1			
	1 0 1			
	1 1 1			
	is reached.			
5	To suppress printing		E	0**
6	To reinstate printing		E	1**
	*The board is both print/paused			
	in the form			
	0.789			
	0.456			
	0.123			
	and displayed as			
	9.123456789 xx			
	where xx is the number of			
	moves you've completed.			
	**If you don't get the desired			
	output, press E again.			

14-03

Example:

Load sides 1 and 2.

Keystrokes:

C →

5 **A** →

This is the only possible move.

6 **A** →

4 **A** →

6 **A** →

5 **A** →

Outputs:

0.000 ***
0.010 ***
0.000 ***

9.000010000 00

0.010 ***
0.101 ***
0.010 ***

9.010101010 01

0.011 ***
0.100 ***
0.011 ***

9.011100011 02

0.100 ***
0.011 ***
0.100 ***

9.100011100 09

0.101 ***
0.010 ***
0.101 ***

9.101010101 10

0.111 ***
0.101 ***
0.111 ***

9.111101111 11

All you have to do is fill in the missing moves and you've got the solution.

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Notes

Golf

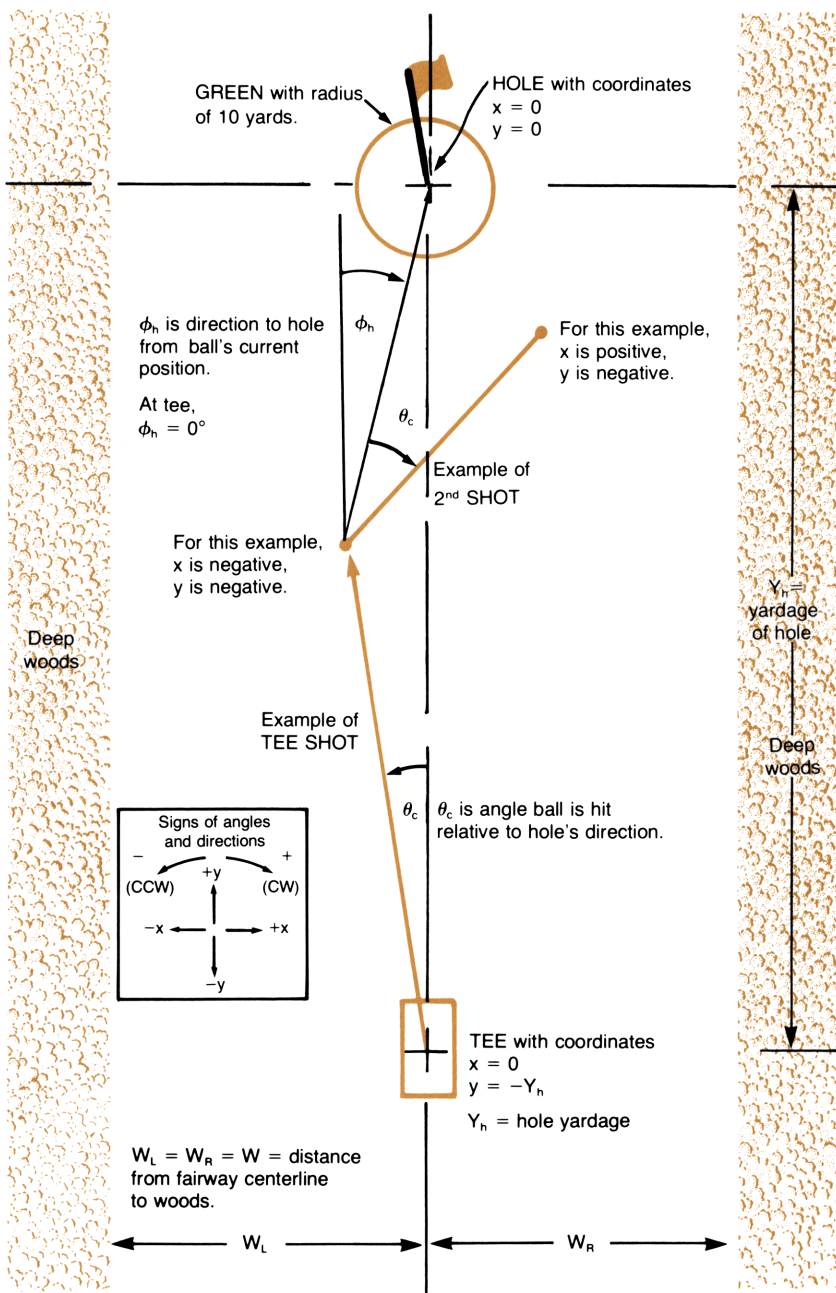


You choose your handicap, design your course, select and swing your club, and hope your ball escapes the woods. One or two people can play.

After you design each hole (actually, your HP-67/97 designs each hole for you), the hole number, hole yardage, par, and distance to the woods are output. After each shot, you are given the distance the ball was hit, the angle the ball was hit relative to the hole's direction, the coordinates of the ball's position, the remaining distance to the hole, and the angle between the ball-hole line and the direction from tee to hole.

The woods are called the deep woods, because that's the kind of trouble you may be in if your ball lands there. If such bad luck befalls you, you have the option of taking a penalty stroke and hitting a second ball from the same spot, or slashing your way through the brambles to find and hit your ball where it lies. The woods are not only thick, but also magic. You can see the flag from every position in the woods, and consequently you always aim directly for the flag. However, if your aim is less than excellent, your ball will hit from one to many trees. Your distance will be sharply reduced, and you may even find yourself moving away from the flag rather than towards it.

Think twice before following your ball into the woods and hitting it from there, since if you do, you have no choice but to club your way out. The example with two players shows what can happen.



General Hole Layout

All clubs may be swung with a full swing or less, allowing you to adjust your swing according to the distance to the hole.

The distance and angle achieved with each hit varies randomly about a mean value. The ball is always aimed directly at the hole, and the deviation from this direction tends to get worse as your handicap increases. The maximum possible distance with each club is independent of handicap, but the means and minimum possible distances decrease as handicap increases. When using a particular club, the number of yards between maximum and minimum possible yardage remains the same, regardless of whether full or partial swing is used. It is generally wise, therefore, to use a full swing whenever possible.

Table 1 gives the mean distances achieved with each club for a 0 handicap player:

TABLE I

Club	Mean Distance	Club	Swing Factor	Mean Distance
1 Wood	260 yds.	10 Iron	1	100
2 Wood	240 yds.	10 Iron	.5	50
3 Wood	220 yds.	11 Iron	1	90
4 Wood	200 yds.	11 Iron	.5	45
1 Iron	190 yds.	Putter	1	10
2 Iron	180 yds.	Putter	.5	5
3 Iron	170 yds.			
4 Iron	160 yds.			
5 Iron	150 yds.			
6 Iron	140 yds.			
7 Iron	130 yds.			
8 Iron	120 yds.			
9 Iron	110 yds.			

The course near each green is treacherous. Tall grass, traps, and other undefined problems will reduce your chipping accuracy. The greens are also challenging. Only the best golfers can do well.

In spite of the championship nature of this course, a duffer can beat the best because of the method of scoring. After each hole is completed, the player's adjusted score is given. This compares the total strokes less handicap against total par for the number of holes completed. While an 18 handicapper may have more trouble with woods, approach shots and green, his score is reduced by 1 stroke per hole before being compared against par. A negative adjusted score means you're beating par.

When playing 2 person golf, it is easy to keep track of whose storage register bank is active (Did I press **f** **P&S** or didn't I?). One way is to recall R7 and see whose handicap is displayed. Another way is to use R6 in each bank to hold an identifying number (such as 1 or 2) for each player. Recalling R6 could then display the player number whose bank was active.

If you want to simplify the game by moving the woods further away for all holes (or make it more difficult by moving the woods closer), change the woods constant as follows:

1. Choose minimum and maximum woods distances ($W_{\max} = 3W_{\min}$). The standard range is from 60 to 20 yards from the fairway centerline.
2. Determine the midpoint of this range (must be a 2 digit integer). This is your new woods constant.
3. Press **GTO** .018.
4. Switch to PRGM. The display will show 018 00.
5. Remove old woods constant by pressing **DEL** two times. The display will show 016-55 (HP-97) or 016 61 (HP-67).
6. Key in new woods constant. This must be a 2 digit integer, such as 70 or 55. The display will show 018 OX (where X is the second digit of your new constant).
7. Switch to RUN.

When you've inserted your woods constant into program memory, you might wish to preserve the revised program by recording it on a magnetic card.

TABLE II

Symbols

u_0	Initial random number seed.
H	18 hole handicap.
h	Hole number.
Y_h	Hole yardage.
W	Distance from centerline of fairway to right and left woods.
C#	Club number.
Y_c	Distance ball hit (club yardage).
θ_c	Angle ball hit relative to hole direction. A 0° angle means the ball is hit directly towards the hole.
y	y coordinate of ball's position after hit.
x	x coordinate of ball's position after hit.
D_h	Distance from ball to hole after hit.
ϕ_h	Direction to hole after hit. If $\phi = 0^\circ$, the ball lies on the fairway's centerline, the line between tee and hole.
S	Adjusted score. This equals the sum of pars for the holes played subtracted from total strokes reduced by adjusted handicap (adjusted for number of holes played).

In symbols:

$$S = \left(\text{Total strokes} - \frac{hH}{18} \right) - \Sigma \text{ par}$$

A negative S means you're beating par.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Clear registers.		1 CL REG	
			1 P±S	
			1 CL REG	
3	Choose and store a six digit random number seed u_0 , between 0 and 1 such as .345762. To change the character of each game, store a different random number.		STO A	
4	For one person golf, go to step 5. For two people golf, go to step 13.			
	ONE PERSON GOLF			
5	Store your 18 hole handicap (handicap may not be negative).	H	STO 7	H
6	Design hole.		A	h, Y_n , Par, W^*
7	Choose club and enter club number:			
	<i>Either:</i>			
	Wood (Club #1, 2, 3 or 4),	C#	B	0
	<i>or:</i>			
	Iron (Club #1, 2, 3, 4, ..., 10			
	or 11),	C#	C	0
	<i>or:</i>			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	Putter (use only if hole distance (D_h) is 10 yds or less).		D	0
8	Choose and enter swing factor s, where s is from 0 to 1.	s	R/S	See below for output.
9	If you want to take a penalty stroke and hit another ball from the same place, go to step 29. Otherwise, go to step 10.			
10	Repeat steps 7, 8 and 9 until ball is holed.			
11	When ball is holed, start next hole at step 6.			
12	For new game, start at step 2 (you may omit step 3).			
	TWO PEOPLE GOLF			
13	Store 18 hole handicaps (must be zero or positive).			
	Player 1:	H_1	STO 7	
			↑ P↔S	H_1
	Player 2:	H_2	STO 7	
			↑ P↔S	H_2
14	Design hole.		A	$h, Y_n \text{ Par}, W^*$
15	Player 1, make your mark.	1	SPACE	
			PRINT X	$1†^*$
16	Player 1, choose club and enter club number:			
	<i>Either:</i>			
	Wood (Club #1, 2, 3 or 4),	C#	B	0
	or:			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	Iron (Club #1, 2, 3, ..., 10			
	or 11),	C#	C	0
	or:			
	Putter (use only if hole dis-			
	tance (D_h) is 10 yds or less).		D	0
17	Player 1, choose and enter			
	swing factor s, where s is			
	from 0 to 1.	s	R/S	See below
				for output.
18	Player 1, if you want to take a			
	penalty stroke and hit another			
	ball from the same place, go to			
	step 29. Otherwise, go to			
	step 19.			
19	Player 1, has player 2 finished			
	this hole?			
	Yes: Repeat steps 16, 17 and			
	18 until you finish this			
	hole. Then go to step 14.			
	No: Allow player 2 to hit his ball.		I P&S	
	Note: Steps 20, 21, 22 and 23			
	are identical to steps 15, 16, 17			
	and 18, except player 2 is			
	playing instead of player 1.			
20	Player 2, make your mark.	2	SPACE	
			PRINT x	2†*
21	Player 2, choose club and			
	enter club number:			
	<i>Either:</i>			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	Wood (Club #1, 2, 3 or 4),	C#	B	0
	or:			
	Iron (Club #1, 2, 3, ..., 10			
	or 11),	C#	C	0
	or:			
	Putter (Use only if $D_h = 10$ yds			
	or less).		D	0
22	Player 2, choose and enter			
	swing factor s, where s is from			
	0 to 1.	s	R/S	See below for
				output.
23	Player 2, if you want to take a			
	penalty stroke and hit another			
	ball from the same place, go to			
	step 29. Otherwise, go to			
	step 24.			
24	Player 2, has player 1 finished			
	this hole?			
	Yes: Repeat steps 21-23 until			
	ball is holed. Then go to			
	step 26.			
	No: Allow player 1 to hit his			
	ball.		I P&S	
25	Player 1, go to step 15.			
26	Player 2, allow player 1 to hit			
	his tee shot.		I P&S	
27	Player 1, go to step 14.			
28	For new game, go to step 2			
	(you may omit step 3).			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
29	To take penalty stroke:		[1] [STO]	
			[+] [1]	
		previous y	[STO] [2]	
		previous x	[STO] [3]	
		previous D _h	[STO] [4]	
		previous ϕ_h	[STO] [5]	
	Then go to next step (10, 19			
	or 24).			
	Output: After choosing and			
	inputting club number and			
	swing factor, the following			
	output is provided:			
	<i>Either:</i>			
	If ball is not sunk in hole:			$Y_c, \theta_c,$
				y, x,
				D_h, ϕ_h^*
	Note: A six decimal output for			
	D _h and ϕ_h show that ball has			
	landed in woods.			
	<i>Or:</i>			
	If ball is holed:			0.000000000,
				S*
	*After obtaining this output,			
	player may review his situation			
	as follows:			
	18 hole handicap		[RCL] [7]	H
	Total strokes (not reduced			
	by handicap)		[RCL] [1]	S'

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	Total par of holes played or			
	started		RCL 9	Σ par
	†These steps are optional.			

Example 1:

One player game.

This records the first hole fortune of Ken Brambles, a moderately accomplished Sunday afternoon golfer carrying an 18 hole handicap of 18.

Load side 1 and side 2.

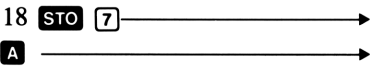
Keystrokes:



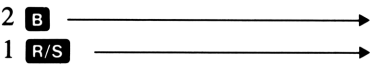
Outputs:

1. u_0 Seed

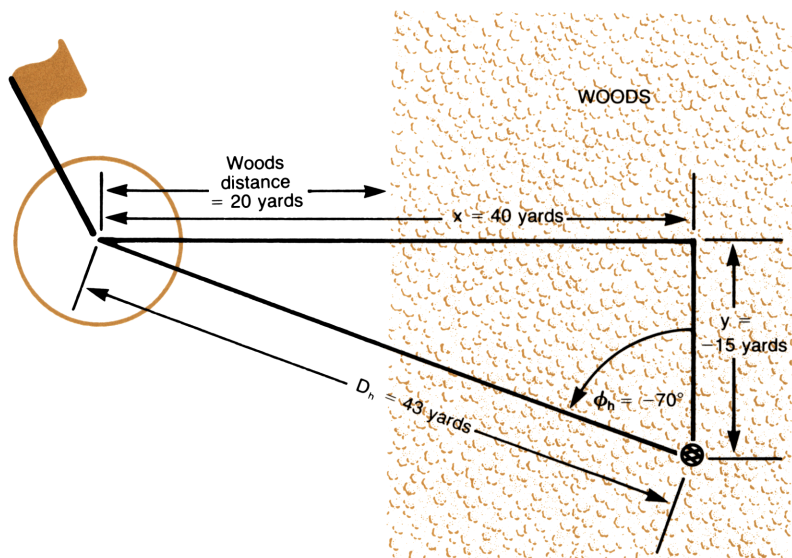
Since the program is run in DSP 0 format, numbers are rounded to the nearest whole number.



18.
1. *** h Hole no.
241. *** Y_h Hole yd.
4. *** Par
20. *** W Woods dist.
0.
229. *** Y_c Club yd.
10. *** θ_c Club angle
-15. *** y }
40. *** x } Ball location
42.981363 *** D_h Hole dist.
-69.671474 *** ϕ_h Hole angle



A six decimal place output for D_h and ϕ_h shows ball has landed in woods.
Ken's tee shot covered 229 yards with a very acceptable 10° slice. His ball is now located as follows:



Ken, a cautious man, elects to take a penalty stroke rather than fight the woods, especially since he's 20 yards from the fairway at the nearest point. Note that -229 and 0 are input as previous y and x , since these are the coordinates of the tee (see drawing on page 15-02).

1 STO + 1 229 CHS STO 2

0 STO 3 229 STO 4 0

STO 5 2 B →

1 R/S →

11 C →

.2 R/S →

0.

211. *** Y_c Club yd.

-2. *** θ_c Club angle

-19. *** y } Ball location

-6. *** x }

19. *** D_h Hole dist.

18. *** ϕ_h Hole angle

0.

4. *** Y_c Club yd.

-11. *** θ_c Club angle

-15. *** y } Ball location

-5. *** x }

16. *** D_h Hole dist.

20. *** ϕ_h Hole angle

11 **C** →
.2 **R/S** →

- 0.
- 20. *** Y_c Club yd.
- 1. *** θ_c Club angle
- 4. *** y } Ball location
- 2. *** x }
- 5. *** D_h Hole dist.
- 154. *** ϕ_h Hole angle

Since D_h is less than 10 yards, the distance from the edge of the green to the hole, Ken has made the green. A D_h of 10 yards also means “on the green.”

D →
.6 **R/S** →

- 0.
- 5. *** Y_c Club yd.
- 3. *** θ_c Club angle
- 1.613743000-02 *** y } Ball location
- 2.725378030-01 *** x }

In DSP 0 format, a number smaller than .5 is presented by the 67/97 in scientific notation. Look for these scientific notation numbers. They mean your hit has excellent directional accuracy, or (as here) you’re very close to the hole, or you’re very near the center of the fairway.

D →
.1 **R/S** →

- 2.730151474-01 *** D_h Hole dist.
- 87. *** ϕ_h Hole angle
- 0.
- 0.000000000 *** Holed!
- 2. *** S Score

Because of bad luck, Ken is 2 over par, even including his 1 stroke/hole handicap.

A →

- 2. *** h Hole no.
- 130. *** Y_h Hole yd.
- 3. *** Par
- 33. *** W Woods dist.

A poor start, but perhaps if we stop looking over Ken’s shoulder, he’ll do better.

Example 2:

Two player game.

This match pits D.C. Divot against Janet Birdie, one of the leading lady golfers of our time. Janet carries a 0 handicap, while D.C. stumbles around the course under a 34. As you will see, however, D.C.’s spirit is as high as his handicap. He is a charger and a scrambler.

Load side 1 and side 2.

Keystrokes:

f **CL REG** f **PzS** f **CL REG**

.385246 **STO** **A** → 3.852460000-01

0 **STO** **7** f **PzS** →

34 **STO** **7** f **PzS** →

A →

1 f **SPACE** **PRINT x** →

This step is optional.

1 **B** →

1 **R/S** →

f **PzS**

2 f **SPACE** **PRINT x** →

1 **B** →

1 **R/S** →

f **PzS**

Janet outdrove D.C. by 42 yards, and she has a good fairway lie compared to D.C.'s 5 yard penetration of the left woods.

1 f **SPACE** **PRINT x** →

1 **B** →

1 **R/S** →

Outputs:

u₀ Seed
0. Janet's handicap
34. D.C.'s handicap
1. *** h Hole no.
571. *** Y_h Hole yd.
5. *** Par
24. *** W Woods dist.
1. Janet's I.D. no.

0.
246. *** Y_c Club yd.
2. *** θ_c Club angle
-325. *** y } Ball location
11. *** x }
325. *** D_h Hole dist.
-2. *** ϕ_h Hole angle

2. D.C.'s I.D. no.

0.

204. *** Y_c Club yd.

-8. *** θ_c Club angle

-369. *** y } Ball location
-29. *** x }

370.509734 *** D_h Hole dist.

4.455967 *** ϕ_h Hole angle

1. Janet

0.

276. *** Y_c Club yd.

1. *** θ_c Club angle

-49. *** y } Ball location
9. *** x }

50. *** D_h Hole dist.

-10. *** ϕ_h Hole angle

15-15

f PzS
2 f SPACE PRINT x →
1 B →
1 R/S →

2. D.C.
0.
25. *** Y_c Club yd.
-65. *** θ_c Club angle
-357. *** y } Ball location
-51. *** x }
360.811319 *** D_h Hole dist.
8.046678 *** ϕ_h Hole angle

f PzS

Since D.C. could see the flag from his lie in the woods, he decided to make a try for it rather than take a penalty shot. The window out of the woods proved too narrow, however, and D.C.'s gutsy #1 wood shot hit many trees, and finally came to rest only 10 yards closer to the hole and 22 yards further into the woods.

1 f SPACE PRINT x →
10 C →
.5 R/S →

1. Janet
0.
48. *** Y_c Club yd.
1. *** θ_c Club angle
-1. *** y } Ball location
1. *** x }
2. *** D_h Hole dist.
-52. *** ϕ_h Hole angle

f PzS

Janet now finds herself only 2 yards from the hole. An excellent approach shot from 50 yards out.

2 f SPACE PRINT x →
1 B →
1 R/S →

2. D.C.
0.
30. *** Y_c }
-53. *** θ_c } Result of
-337. *** y } D.C.'s swing
-72. *** x }
344.037744 *** D_h } Hole location
12.002730 *** ϕ_h } from ball

f PzS

D.C. gained another 16 yards on the hole, but he's now buried 48 yards into the woods.

1 f [SPACE] [PRINT x] →
D →
.3 R/S →

1. Janet
0.
3. *** Y_c
-7. *** θ_c } Result of
1. *** y } Janet's swing
-1. *** x }
1. *** D_h } Hole location
114. *** ϕ_h } from ball

f [PzS]

Too bad, Janet missed a good birdie chance.

2 f [SPACE] [PRINT x] →
1 B →
1 R/S →

2. D.C.
0.
32. *** Y_c
100. *** θ_c
-348. *** y
-42. *** x
350.755487 *** D_h
6.898699 *** ϕ_h

f [PzS]

1 f [SPACE] [PRINT x] →
D →
.2 R/S →

1. Janet
0.
0.000000000 ***
0. S

f [PzS]

Janet made her par.

2 f [SPACE] [PRINT x] →
1 B →
1 R/S →

2. D.C.
0.
246. *** Y_c
-1.803671685-01 *** θ_c
-104. *** y
-13. *** x
105. *** D_h
7. *** ϕ_h

The rewards of try, try again. D.C. found the window.

9 **C** →
1 **R/S** →

- 0.
- 98. *** Y_c
- 1. *** θ_c
- 7. *** y
- 1. *** x
- 7. *** D_h
- 10. *** ϕ_h

D.C. is on the green.

D →
.8 **R/S** →

- 0.
- 8. *** Y_c
- 12. *** θ_c
- 2.430006000-03 *** y
- 2. *** x
- 2. *** D_h
- 90. *** ϕ_h

D →
.3 **R/S** →

- 0.
- 0.000000000 ***
- 1. S

f **P&S**

D.C.'s struggles in the woods gave him an 8 on this hole, but considering his nearly 2 strokes/hole handicap, he stands only 1 over par, 1 stroke behind Janet. It looks like a close match.

A →

- 2. *** h
 - 399. *** Y_h
 - 4. *** Par
 - 22. *** W
- } Hole no. 2 layout

You might wish to continue to see who wins.

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Notes

The Dealer



The Dealer is capable of drawing from a numerical deck or bin without replacement. That is, once a card or number has been selected, it will not be selected again until a shuffle is performed.

If the “Cards” mode is selected, numerical cards of 4 suits, containing 13 cards each, are dealt as follows:

S.	CC
Digit position	Exponent position

In this display format, S is the suit (digit from 1-4) and CC is the card of the indicated suit (digit from 1-13). The following convention is used for a standard deck of 52 cards:

Suit Convention	Card Convention
Spade = 1.	Ace = 01
Heart = 2.	2-10 = 02-10
Diamond = 3.	Jack = 11
Club = 4.	Queen = 12
	King = 13

The bingo mode simply selects numbers between 1. and 75. without replacement.

In both card mode and bingo mode, it is possible to draw one value at a time using **A** or many values automatically using **C**. To use the automatic feature, key in the number of values wanted before pressing **C**.

A shuffle may be performed at any time by pressing **f** **C**. An automatic shuffle is performed after all cards or numbers have been dealt.

The Dealer will start the same sequence of cards or numbers each time the card or bingo mode is selected unless the seed used to start the sequence is keyed in by the player(s). A seed is any number between 0 and 1. The seed must be keyed in after selecting card or bingo mode and **R/S** is used to store it. A fair way to select a seed in a multiplayer game is to have the dealer key a decimal point and the first digit and have each player key in a subsequent digit until the display is full. Press **R/S** to store the seed.

Remarks:

It is possible to modify the dealer to deal up to 100 numbers in bingo mode. Steps 109 and 110 control the number of objects dealt. If you wish to deal 85 numbers instead of 75 numbers replace the 23 (steps 109 and 110) with 33 ($23 + (85 - 75)$). Similarly a joker could be added to the card dealer by replacing the 51 at steps 112-113 with 52. The Joker would be displayed as the first card of the fifth suit.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	For bingo-type dealer (deals numbers without repetition), go to step 10.			
3	Select card dealer.		I A	0.
4	Optional: input seed ($0 < \text{seed} < 1$).	seed	R/S	seed
5	Deal one card, or deal a specified number of cards.	n	A C	"suit. card" "suit. card"
6	Optional: Review cards dealt.		E	"suit. card"
7	Go back to step 5 for more cards or go to step 8 for shuffle.			
8	Shuffle.		I C	0. 00
9	Go to step 5 and deal cards.			
10	Select bingo dealer.		I E	0.
11	Optional: input seed ($0 < \text{seed} < 1$).	seed	R/S	seed

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
12	Draw one number		A	"#."
	or draw a specified number			
	of numbers.	n	C	"#."
13	Optional: Review numbers			
	drawn.		E	"#."
14	Go back to step 12 for more			
	numbers or go to step 15 for			
	a new game.			
15	Shuffle.		f C	0.
16	Go to step 12 for more			
	numbers.			

Example 1:

Deal two poker hands of five cards. First use the program’s seed, then use a seed of .896348.

Load sides 1 and 2.

Keystrokes:

f A →
5 **C** →

5 **C** →

Outputs:

0.
4. 04 *** (4C)
3. 02 *** (2D)
2. 01 *** (AH)
1. 02 *** (2S)
1. 10 *** (10S)
0. 00
3. 09 *** (9D)
3. 12 *** (QD)
4. 03 *** (3C)
3. 04 *** (4D)
3. 01 *** (AD)
0. 00

f A →
.896348 R/S →
5 C →

0. 00
9. -01
3. 08 ***
4. 10 ***
1. 10 ***
1. 01 ***
1. 06 ***
0. 00
1. 03 ***
2. 11 ***
2. 02 ***
1. 11 ***
2. 13 ***
0. 00

5 C →

Example 2:

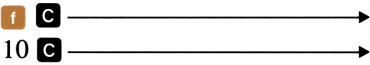
Generate 10 bingo numbers, one at a time and then review the numbers automatically. Then shuffle, and deal 10 more numbers automatically.

Keystrokes:

f E →
A →
A →
A →
A →
A →
A →
A →
A →
A →
E →

Outputs:

0. 00
62.
41.
20.
3.
14.
52.
57.
64.
45.
74.
62. ***
41. ***
20. ***
3. ***
14. ***
52. ***



- 57. ***
- 64. ***
- 45. ***
- 74. ***
- 0.
- 17. ***
- 54. ***
- 36. ***
- 55. ***
- 6. ***
- 21. ***
- 33. ***
- 3. ***
- 61. ***
- 64. ***
- 0.

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Notes

Bowling Scorekeeper



This program keeps score for up to 10 bowlers. Each bowler is identified by a number (ID). Using his identification number, a player may input the pin count for each ball, and access his score.

To score a frame, key in the bowler's ID number, a decimal point, and a one digit pin count (P) for each ball. For instance, if bowler 6 knocked down three pins on the first ball and 5 pins on the second ball, the keystrokes would be as follows:

6.3 **A**

6.5 **A**.

A strike for player 3 (10 pins on the first ball) is indicated by:

3 **C**.

Similarly, if player 4 knocked down 9 pins on the first ball and then picked up the spare, the score would be indicated by:

4 **B**.

The score is displayed after each player's pin count is input. The format is shown below:

S.FS NF

or

-S.FS NF

where:

– (if present) = bowl another ball this frame.

S = Score calculated through frame FS.

FS = Frame number containing most recent score (frames scored).

NF = Frame in which next ball will be bowled (next frame).

By pressing **[DSP]** 6, the pin count (PC) for the last ball is displayed in the last 2 digits:

S.FS NF PC

A player's score may be displayed at any time by keying in the player's ID number and pressing **D**.

Remarks:

Players need not bowl in order. It is not necessary to complete a particular bowler's frame before input of another bowler's score. This allows two lanes to be scored simultaneously.

Other variables used in listing (pages L17-01, -02)

SSS = Score

CS = Current sum within a frame.

S1 = Strike one frame earlier.

S2 = Strike two frames earlier.

PS = Spare previous frame.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Initialize.		I A	0.0000
3	Input each ball's score:			
	a) For frame's first ball (no			
	strike), key in player's ID			
	number, decimal point, and			
	single digit pin count.	ID.P	A	-S.FSNF*
	b) For frame's second ball			
	leaving open frame	ID.P	A	S.FSNF
	c) For spare	ID	B	S.FSNF
	d) For strike	ID	C	S.FSNF
4	For next ball or another			
	bowler, go to step 3.			
5	Display score (at any time).	ID	D	
6	Optional: Increase display to			
	show pin count of last ball.			
	Step 3 output will become			
	S.FSNFPC (PC = last ball pin			
	count).		DSP 6	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
7	For a new game, go to step 2.			
	*S = Score			
	FS = Frame associated with			
	score			
	NF = Frame in which next			
	ball will be bowled			
	Minus sign means a second			
	ball should be bowled this			
	frame.			

Example:

Score the games below using the calculator.

Player 1

8	7	2			8	1		8			7	1		8
17	26	54	73	82	102	122	140	148	168					
7	1					9			6	2	8		9	-
8	38	68	97	117	137	155	163	182	191					

Player 2

Load sides 1 and 2.

Keystrokes:

f A →
1.8 A →

Outputs:

0.0000
-0.0001 ***

The minus sign means a second ball should be bowled this frame. There is no score, and the next ball will be bowled in frame 1.

1 B → 0.0002 ***

Player 1’s next ball will be in frame 2.

2.7 A → -0.0001 ***
2.1 A → 8.0102 ***

Player 2's score is 8 in frame 1, and his next ball will be bowled in frame 2.

1.7 **A** → -17.0102 ***

Player 1's score in frame 1 is 17, and he has another ball to bowl in frame 2.

1.2 **A** → 26.0203 ***

Since frame 2 is open, player 1's score (26) can be calculated up to the current frame (2). His next ball will be in frame 3.

2 **C** → 8.0103 ***

1 **C** → 26.0204 ***

2 **C** → 8.0104 ***

1 **C** → 26.0205 ***

2 **C** → 38.0205 ***

The strike in frame 2 can now be scored, since two more balls have been bowled.

1.8 **A** → -54.0305 ***

1.1 **A** → 82.0506 ***

2 **C** → 68.0306 ***

1 **C** → 82.0507 ***

2.9 **A** → -97.0406 ***

2 **B** → 117.0507 ***

Recall score of player 1.

1 **D** → 82.0507

1.8 **A** → -82.0507 ***

1 **B** → 102.0608 ***

2 **C** → 137.0608 ***

1 **C** → 122.0709 ***

2.6 **A** → -137.0608 ***

2.2 **A** → 163.0809 ***

1.7 **A** → -122.0709 ***

1.1 **A** → 148.0910 ***

2.8 **A** → -163.0809 ***

2 **B** → 163.0810 ***

The players would like to have each ball's pin count displayed:

DSP 6 → 163.081010

17-05

For a spare and a strike, the pin count is shown as 10.

1 **c** $\longrightarrow$ 148.091110

Here, the “11” (xxx.xx11xx) means another ball should be bowled in the 10th frame.

1.8 **A** $\longrightarrow$ -148.091108

Now a minus sign calls for the 3rd 10th frame ball.

```
1 B → 168.100000 ***
```

Player 1's final score

2.9 **A** $\longrightarrow$ -182.091009 ***

2.0 **A** $\longrightarrow$ 191.100000 ***

Player 2 wins easily.

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Notes

Biorhythms



BIORHYTHMS

GA1-18A

BIRTH

BIODATE

+ +1

+ MAX

+ MIN

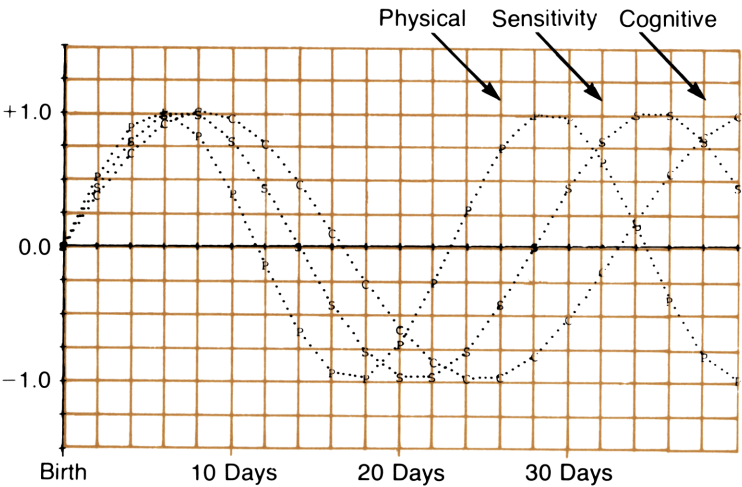
PRINT?

+ CRITICAL

From the ancient of days, philosophers and sages have taught that human happiness lies in the harmonious integration of body, mind, and heart. Now a twentieth-century theory claims to be able to quantitatively gauge the functioning of these three aspects of ourselves: the physical, sensitive, and cognitive.

The biorhythm theory is based on the assumption that the human body has inner clocks or metabolic rhythms with constant cycle times. Currently, three cycles starting at birth in a positive direction are postulated. The 23-day or physical cycle relates with physical vitality, endurance and energy. The 28-day or sensitivity cycle relates with sensitivity, intuition and cheerfulness. The 33-day or cognitive cycle relates with mental alertness and judgement.

For each cycle a day is considered either high, low, or critical. The high ($0 < x \leq 1$) times are regarded as energetic times, you are your most dynamic in the cycle. The low ($-1 \leq x < 0$) times are regarded as the recuperative periods. The critical days ($x = 0$) are regarded as your accident prone days especially for the physical and sensitivity cycles.



Operating Limits and Warnings:

The birthdate and biodate must occur between January 1, 1901 and December 31, 2099.

The date format for input is MM.DDYYYY (March 3, 1976 is keyed in as 3.031976). The program does not check input data. Thus, if an improper format or an invalid date (e.g., February 30) is keyed in, erroneous answers will result.

Reference:

This program is based on an HP-65 Users' Library program by Grant Munsey.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	(Optional) To cancel PRINT/			
	PAUSE mode		 	0*
	Later for automatic output of			
	results set PRINT/PAUSE			
	mode.		 	1*
3	Key in the following:			
	Birthdate	MM.DDYYYY		
	Biodate	MM.DDYYYY		P**
				S**
				C**
4	To calculate the cycles for			
	Biodate + 1, 2, ...			P (day+1)
				S (day+1)
				C (day+1)
				P (day+2)
				S (day+2)
				C (day+2)
	To stop cycle			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
5	To calculate which of the next			
	33 days after biodate are			
	critical days		E	# days P
				# days S
				# days C
6	To calculate which of the next			
	33 days after biodate are			
	max days		I D	# days P
				# days S
				# days C
7	To calculate which of the next			
	33 days after biodate are min			
	days		D	# days P
				# days S
				# days C
	*If you don't get the desired			
	output, press I E again.			
	**Physical, Sensitivity,			
	Cognitive			

Example:

Calculate the Bio values for the month of June 1976 if birthdate is June 7, 1948.
Load sides 1 and 2.

Keystrokes:

6.071948 **A** →

By using May 31 for biodate (5.311976 **B**), instruction steps 5, 6 and 7 give correct day numbers for month of June.

5.311976 **B** →

Outputs:

6.07

0.82 *** P
0.00 *** S
-0.95 *** C

These are the values of the three cycles for May 31, 1976.

C	→																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
----------	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Listing stops.

f D	→	20.75 *** P
		7.00 *** S
		18.25 *** C

The one maximum for the physical cycle during June (and the first 3 days of July) is June 20. Similarly, the sensitivity and cognitive cycles have one maximum each in this period, June 7 and June 18.

E	→	3.50 ***	} P
		15.00 ***	
		26.50 ***	

14.00 *** } S
28.00 *** }

10.00 *** } C
26.50 *** }

There are 3 critical days in June for the physical cycle, June 3, 15, and 26. The other two cycles have 2 critical days each.

D 

9.25 *** } P
32.25 *** }

21.00 *** S

1.75 *** C

Only the physical cycle has more than one minimum day during this 33 day period.

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

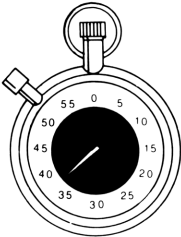
Notes

Timer



This program contains five different timers:

- 1. Five-second interval visible timer
- 2. Minimum interval visible timer
- 3. Count-Up timer
- 4. Count-Down timer
- 5. Splits



Calibration routines are also included to accomodate differences between calculators and different environments. The clock circuits in the HP-67 and HP-97 are designed for calculator use, not for accurate time keeping. Consequently, highly stable performance should not be expected.

In addition to the primary calibration constants whose adjustments are covered below, a secondary constant K_a , is also used for the 5 second timer. This determines how fast the calibration routine “zeros in” on the best value for the 5 second primary constant. K_a generally should require no user adjustment. These constants are named in Table I and are indicated in the listings.

TABLE I

CALIBRATION CONSTANTS

C_a	Primary for 5-Second Timer
C_b	Primary for Minimum Interval Timer
C_c	Primary for Count-Up Timer
C_d	Primary for Count-Down Timer
C_e	Primary for Splits Timer
K_a	Secondary for 5-Second Timer

A split is a time measurement which is preserved without stopping the clock. The split routine allows up to 16 splits to be taken during one continuous running of the count-up timer. If the count-up timer is properly calibrated, 10 splits taken over a few minutes should introduce no more than a few seconds error.

CALIBRATION ROUTINES

5-Second Timer

- 1. Initialize (**A**), and note sweep second hand time **R/S** is pressed.
- 2. When timer reaches chosen time T_p (program time), note sweep second hand finish time.

3. Stop timer (**RTN**), and enter sweep second hand finish time (H.MS **ENTER**↵).
4. Enter sweep second hand start time (H.MS **ENTER**↵).
5. Key in T_p and calculate new C_a (H.MS **f** **A**).
6. Enter the displayed C_a into program as follows:
 - a. Press **GTO** .016.
 - b. Switch to PRGM. Display should show 016 0X (X = last digit of old C_a).
 - c. Remove old C_a by pressing **DEL** 6 times. Display should show 010 35 15 (HP-97) or 010 33 15 (HP-67).
 - d. Key in new C_a . Display should show 016 0X (X = last digit of new C_a).
 - e. Switch to RUN.
7. Repeat steps 1-2 to check calibration, and if necessary, repeat steps 3-7.

Minimum Interval Timer

1. Initialize (**B**), and note sweep second hand time **R/S** is pressed.
2. When timer reaches chosen time T_p (program time), note sweep second hand finish time and SIMULTANEOUSLY stop timer (**RTN**). Timer will not stop unless display is steady when **RTN** is pressed.
3. Enter sweep second hand finish time (H.MS **ENTER**↵).
4. Enter sweep second hand start time (H.MS **ENTER**↵).
5. Key in T_p and calculate new C_b (H.MS **f** **B**).
6. Enter the displayed C_b into program as follows:
 - a. Press **GTO** .008.
 - b. Switch to PRGM. Display should show 008 0X (X = last digit of old C_b).
 - c. Remove old C_b by pressing **DEL** 4 times. Display should show 004 16-53 (HP-97) or 004 31 43 (HP-67).
 - d. Key in integer portion (4 digits) of new C_b . Display should show 008 0X (X = last digit of new C_b).
 - e. Switch to RUN.
7. Repeat steps 1-2 to check calibration, and if necessary, repeat steps 3-7.

Count-Up Timer

1. Initialize (**C**), and note sweep second hand time **R/S** is pressed.

2. When sweep second hand reaches desired finish time, stop timer by pressing any key until display stabilizes.
3. Enter sweep second hand finish time (H.MS **ENTER**).
4. Key in sweep second hand start time and display elapsed time T_p (program time) measured by count-up timer (H.MS **f** **D**).
5. Calculate new C_c (**R/S**).
6. Enter the displayed C_c into program as follows:
 - a. Press **GTO** .027.
 - b. Switch to PRGM. Display should show 027 0X (X = last digit of old C_c).
 - c. Remove old C_c by pressing **DEL** 4 times. Display should show 023 35 13 (HP-97) or 023 33 13 (HP-67).
 - d. Key in integer portion (4 digits) of new C_c . Display should now show 027 0X (X = last digit of new C_c).
 - e. Switch to RUN.
7. Repeat steps 1-4 to check calibration, and if necessary, repeat steps 5-7.

Count-Down Timer

1. After initialization (**D**), enter start time, which equals elapsed time T_p (program time) as measured by timer (H.MS **R/S**).
2. Start timer, and note sweep second hand time **R/S** is pressed.
3. Note sweep second hand finish time the instant timer displays 0.0000.
4. Enter sweep second hand finish time (H.MS **ENTER**).
5. Key in sweep second hand start time and calculate new C_d (H.MS **f** **E**).
6. Enter displayed C_d into program as follows:
 - a. Press **GTO** .033.
 - b. Switch to PRGM. Display should show 033 0X (X = last digit of old C_d).
 - c. Remove old C_d by pressing **DEL** 4 times. Display should show 029 35 14 (HP-97) or 029 33 14 (HP-67).
 - d. Key in integer portion (4 digits) of new C_d . Display should now read 033 0X (X = last digit of new C_d).
 - e. Switch to RUN.
7. Repeat steps 1-3 to check calibration, and if necessary, repeat steps 4-7.

Splits

1. The following procedure should be followed only after the count-up calibration constant has been adjusted to give acceptable timer performance.
2. Initialize (**C**), and note sweep second hand time **R/S** is pressed.
3. At chosen sweep second hand time intervals, take a series of from 10 to 16 splits (**E E**—see instruction steps 19 and 20).
4. When sweep second hand reaches desired finish time, stop timer by pressing any key until display stabilizes.
5. Display and note total elapsed time (**f C**).
6. Display and note splits (press **R/S** repeatedly).
7. Program steps 158, 159, 160 and 161 contain the split calibration constant, whose form is X.XX. If the split times are too small, this constant is too small, and visa versa. Key a new split constant into program memory and repeat steps 2-6. Increasing the split constant 0.10 will increase the displayed time after 16 splits roughly one second or so.

Normally, several runs through this calibration procedure should allow you to “zero in” on a value for this constant which gives an accuracy over 16 splits of one or two seconds (plus whatever error might be introduced by the count-up timer itself over the total time interval measured).

After you have entered your new calibration constants into program memory, you may wish to record the timer program on a different card to preserve your new constants.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
	For 5-second interval visible			
	timer:			
	Go to step 2.			
	For minimum interval			
	visible timer:			
	Go to step 6.			
	For count-up timer:			
	Go to step 10.			
	For count-down timer:			
	Go to step 14.			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	For splits:			
	Go to step 17.			
	5-SECOND VISIBLE TIMER			
2	Initialize.		A	0.0000
3	If you want timer to start at a time other than zero, key in time.	H.MMSS		H.MMSS
4	Start 5-second timer.		R/S	H.MMSS
	Each PAUSE begins at the time displayed.			
5	Stop 5-second timer. Press RTN during time display.		RTN	H.MMSS
	MINIMUM INTERVAL VISIBLE TIMER			
6	Initialize.		B	0.0000
7	If you want timer to start at a time other than zero, key in time.	H.MMSS		H.MMSS
8	Start minimum interval timer.		R/S	H.MMSS
	Even when accurately calibrated, displayed time can vary ± 1 second from correct time.			
9	Stop minimum interval timer.			
	Press RTN during time display.		RTN	H.MMSS
	COUNT-UP TIMER			
10	Initialize.		C	0.0000
11	Start count-up timer.		R/S	
12	Stop count-up timer: Depress any key until display stabilizes.			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
13	Display elapsed time.		t C	H.MMSS
	COUNT-DOWN TIMER			
14	Initialize.		D	0.0000
15	Enter starting time.	t_s H.MMSS	R/S	t_s H.MMSS
16	Start count-down timer.		R/S	
	When starting time interval			
	has elapsed, flashing display			
	is replaced by 0.0000.			
	SPLITS			
17	Initialize.		C	0.0000
18	Start count-up timer.		R/S	
19	Take a split: For HP-97, press			
	E twice, in rapid succession.			
	For HP-67, press E and hold			
	it down just long enough to			
	obtain a steady display. Then			
	press E again quickly.			
20	Take another split: Repeat			
	step 19. Up to 16 splits may			
	be taken. At least 4 seconds			
	must separate adjacent splits.			
21	Stop count-up timer:			
	Depress any key until display			
	stabilizes.			
22	Display total elapsed time.		t C	H.MMSS
23	Display last split.		R/S	H.MMSS
24	Display next-to-last split.		R/S	H.MMSS
25	Display each earlier split in			
	turn:			
	Repeatedly press		R/S	H.MMSS

Example 1:

Operate the 5-second interval visible timer.

Load side 1 and side 2.

Keystrokes:	Outputs:
A →	0.0000
R/S →	0.0000
	0.0005 H.MS
	0.0010
	0.0015
	⋮
	0.0055
	0.0100
RTN →	0.0100

Example 2:

Calibrate the 5-second timer. (You'll probably generate a different calibration constant C_a with your calculator.)

Keystrokes:	Outputs:
A →	0.0000
R/S →	0.0000
Start time $t_s = 9:28:45$.	0.0005 H.MS
	⋮
	0.0155
Finish time $t_f = 9:30:48$.	0.0200
RTN →	0.0200
9.3048 ENTER 9.2845 ENTER	
.02 f A →	4.4347 *** C_a
GTO .016	
Switch to PRGM. →	016 00
DEL DEL DEL DEL DEL DEL →	010 35 15 (HP-97)
	010 33 15 (HP-67)
4.4347 →	016 07
Switch to RUN. →	4.4347
A →	0.0000
R/S →	0.0000

Start time $t_s = 9:36:35.$ →	0.0005	H.MS
	⋮	
Finish time $t_f = 9:38:34.$ →	0.0155	
	0.0200	
RTN →	0.0200	

***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Example 3:

Take 6 splits, 10 seconds apart, and stop the count-up timer at 70 seconds.

Keystrokes:

C →

Start count-up timer.

R/S

On 10 second mark: **E E**

On 20 second mark: **E E**

⋮

On 60 second mark: **E E**

On 70 second mark: **f** →

f C →

R/S →

R/S →

⋮

R/S →

R/S →

Outputs:

0.0000

Ignore display.

0.0110 Elapsed time, H.MS

0.0100 Last split

0.0050 Previous split

⋮

0.0010 First split

0.0000

PROGRAM LISTINGS

The following listings are included for your reference. A table of keycodes and keystrokes corresponding to the symbols used in the listings can be found in Appendix E of your Owner's Handbook.

Program	Page
1. Game of 21	L01-01
2. Dice	L02-01
3. Slot Machine	L03-01
4. Submarine Hunt	L04-01
5. Artillery Game	L05-01
6. Space War	L06-01
7. Super Bagels	L07-01
8. Nim _k	L08-01
9. Queen Board	L09-01
10. Hexapawn	L10-01
11. Tic-Tac-Toe	L11-01
12. Wari	L12-01
13. Racetrack	L13-01
14. Teaser	L14-01
15. Golf	L15-01
16. The Dealer	L16-01
17. Bowling Scorekeeper	L17-01
18. Biorhythms	L18-01
19. Timer	L19-01

Game of 21

001 *LBLM		057 0	
002 SFC		058 ST+7	11 for Ace
003 SFC		059 RCL7	
004 CFB		060 GSB3	
005 CFI		061 RTN	-----
006 ST0A		062 *LBL6	
007 1		063 CFB	
008 ST01		064 1	
009 GSB0		065 0	1 for Ace
010 ST0B		066 ST-7	
011 GSB5		067 RCL7	
012 GSB9		068 GSB3	-----
013 STC1	Bet,	069 RTN	
014 ST03		070 *LBL4	
015 1	New game	071 F0?	
016 ST01		072 GT0E	
017 GSB0		073 RCL4	
018 ST0C		074 CHS	player lose
019 GSB9		075 ST+9	
020 STC2		076 RCL7	
021 ST+3		077 EEX	
022 0		078 2	
023 GSB5		079 ÷	
024 SFC		080 -	-----
025 GSB6		081 GT06	
026 ST05		082 *LBL2	
027 ST07		083 RCL1	
028 GSB6		084 RCL2	
029 ST06		085 *	
030 ST+7		086 RCL3	HP BJ?
031 RCL5		087 +	
032 *		088 RCL0	
033 RCL7		089 X=Y?	
034 +		090 GT07	
035 2		091 FI?	
036 1		092 RTN	-----
037 ST00		093 *LBL8	
038 X=Y?		094 .	
039 GT02		095 2	
040 RCL7		096 1	player BJ!
041 GSB3	-----	097 RCL4	
042 R/S		098 1	
043 *LBL8		099 .	
044 GSB6		100 5	
045 ST+7		101 *	
046 RCL7		102 ST0A	
047 RCL0		103 ST+9	
048 X=Y		104 +	-----
049 X=Y?	Hit	105 *LBL6	
050 GT04		106 PSE	
051 X≠Y?		107 PSE	
052 GSB3		108 PSE	
053 R/S		109 PSE	Display with blinks.
054 *LBLD		110 RCL9	
055 SF0		111 DSP2	
056 1		112 FI?	
REGISTERS			
0 21	1 1 st D.	2 2 nd D.	3 ΣD.
4	5 1 st P.	6 2 nd P.	7 ΣP.
8	9 Σ Bet		
S0	S1	S2	S3
S4	S5	S6	S7
S8	S9		
A Bet	B 1 st D.	C 2 nd D.	D last K _i
E u _i	I Index		

113	R/S			169	GT00		
114	RTN			170	RTN		
115	*LBL7			171	*LBL0		
116	F1?			172	CF0		
117	GT04			173	SPC		
118	0	HP BJI		174	RCLB		
119	DSP9			175	GSB5		
120	GT06			176	PCLC		
121	*LBL9			177	GSB5		
122	1			178	SF1		
123	0			179	GSB2		
124	XZY	10 for J, Q, K		180	CF1		
125	XZY?			181	RCL3		
126	RTN			182	*LBL6		
127	XZY			183	1		
128	RTN			184	7		
129	*LBL3			185	XZY?		
130	EEH	Display		186	GT0d		
131	2			187	GSBe		
132	+	Bet.point		188	ST+3		
133	RCLA			189	RCL3		
134	+			190	2		
135	RTN			191	2		
136	*LBL5			192	XZY?		
137	DSP0			193	GT01		
138	PRTX	print		194	RCL3		
139	DSP2			195	GT06		
140	RTN			196	*LBLd		
141	*LBLc			197	RCL7		
142	0			198	RCL3		
143	ST01			199	-		
144	.			200	X=0?		
145	5			201	GT06		
146	2			202	X=0?		
147	8			203	GT04		
148	1			204	*LBL1		
149	1			205	RCLA		
150	6			206	ST+9		
151	3			207	RCL7		
152	ST0E			208	GSB3		
153	*LBL0	Shuffle		209	GT06		
154	PCLC			210	*LBLc		
155	9			211	1		
156	9			212	ST01		
157	7			213	GSB0		
158	X			214	GSB5		
159	FRC			215	GSB5		
160	ST0E			216	RTN		
161	1			217	*LBLc		
162	3			218	0		
163	x			219	ST09		
164	INT			220	RTN		
165	1						
166	+						
167	ST0D						
168	DSZ1						

LABELS						FLAGS	SET STATUS		
A Bet	B Hit	C Stand	D 11 for Ace	E 1 for Ace	0 11 for Ace	1 HP BJ	FLAGS	TRIG	DISP
a Shuffle	b Dealer	c New P.	d Used	e Next card	2		ON OFF		
0 R. No.	1 P. Win	2 HP BJ?	3 Display	4 P. lose	3		0 ☐ ☒	DEG ☒	FIX ☒
5 Print	6 blinks	7 HP BJI	8 P. BJI	9 10 for J,Q,K			1 ☐ ☒	GRAD ☐	SCI ☐
							2 ☐ ☒	RAD ☐	ENG ☐
							3 ☐ ☒		n <u>2</u>

Dice

001	*LBLA	Input seed.	057	6					
002	EEH		058	X					
003	CHS		059	1					
004	7		060	+					
005	X		061	INT					
006	.		062	RTN					
007	5		063	*LBL4	1 st Roll Routine				
008	2		064	3	Lose on 2, 3, 12.				
009	8		065	RCL5					
010	4		066	X=Y?					
011	1		067	GT07					
012	6		068	2					
013	+		069	X=Y?					
014	ST09	Store in R ₉ .	070	GT07					
015	*LBLC	Reset Winnings (WIN) to 0.	071	R4					
016	0		072	1					
017	ST08		073	2					
018	*LBL1	Display Winnings (WIN).	074	X=Y?					
019	DSP2		075	GT07					
020	FIX		076	R4	Win 7, 11.				
021	SF2	Set 1 st Roll Flag.	077	7					
022	RTN		078	X=Y?					
023	*LBLB	Store Bet in R ₇ .	079	GT06					
024	ST07		080	R4					
025	*LBL2	Get Dice Roll.	081	1					
026	GSD0		082	1					
027	FSE	1 st Roll?	083	X=Y?					
028	F2?		084	ST06					
029	GT04	Yes.	085	R1	Current Roll (C ROLL).				
030	GT05	No.	086	ST04					
031	*LBLD	Dice Roll Routine	087	GT02	Not 1 st Roll.				
032	GSD3	1 st die (1 DIE).	088	*LBL5					
033	ST06	Sum of 2 dice (2 DIE).	089	RCL5	Craps?				
034	ST05		090	7					
035	GSD3		091	X=Y?					
036	ST+5		092	GT07	YES				
037	1		093	R4					
038	0		094	RCL4					
039	÷		095	X≠Y?	Match				
040	RCL6		096	GT02	No.				
041	+		097	*LBL6	Win!				
042	RCL5		098	RCL7					
043	10*		099	RCL8	Increment Winnings (WIN).				
044	X		100	+					
045	DSP1		101	ST08					
046	SCI		102	GT01					
047	RTN		103	*LBL7	Lose!				
048	*LBL3	Random Number Generator.	104	RCL8					
049	3		105	RCL7					
050	RCL9		106	-	Decrement Winnings (WIN).				
051	9		107	ST08					
052	9		108	GT01					
053	7								
054	X								
055	FRC								
056	ST09								
REGISTERS									
0	1	2	3	4 C ROLL	5 2 DIE	6 1 DIE	7 BET	8 WIN	9 SEED
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A	B	C	D	E	F	G	H	I	J

Slot Machine

001 *LBLB	Recalls winnings	057 X#0?	
002 DSP2		058 GT02	
003 RCL0		059 1	
004 RTN		060 0	
005 *LBLB	-----	061 ST+0	
006 CLRG	Sets up new machine	062 RCL2	
007 ST0E	conditions.	063 X#0?	Adds \$10 for XX0.
008 CLX		064 GT09	
009 DSP0		065 9	
010 RTN	-----	066 0	
011 *LBLA	Play:	067 GT08	
012 DSP3	Removes dollar played	068 *LBL2	Adds additional \$90 for
013 1	from winnings.	069 RCL2	000.
014 ST-0		070 RCL3	
015 RCL1		071 X#Y?	
016 EEX	Sets up wheels.	072 GT09	
017 3		073 1	
018 X		074 0	
019 COS		075 GT08	Adds \$10 for XXX,
020 ABS		076 *LBL1	
021 ST0E		077 2	Adds \$2 for 1XY.
022 EEX		078 ST+0	
023 6		079 1	
024 +		080 0	
025 LSTX		081 RCL2	
026 -		082 X#Y?	
027 FRC		083 GT09	
028 ST04		084 3	Adds additional \$3 for
029 1		085 *LBL6	11X.
030 0		086 ST+0	
031 ST05		087 *LBL9	
032 X		088 RCL1	
033 INT		089 1	
034 ST01		090 0	
035 CHS		091 ÷	
036 LSTX		092 DSP1	
037 +		093 PSE	
038 RCL5		094 RCL2	
039 X		095 1	
040 INT		096 0	
041 ST02		097 0	
042 CHS		098 ÷	
043 LSTX		099 +	
044 +		100 DSP2	
045 RCL5		101 PSE	
046 X		102 RCL3	
047 INT		103 EEX	
048 ST03		104 3	
049 1	-----	105 ÷	
050 RCL1	Determine winnings.	106 +	
051 X=Y?		107 DSP3	
052 GT01		108 RTN	
053 RCL2			
054 X#Y?			
055 GT09			
056 RCL3			

REGISTERS									
0 Winnings	1 Wheel 1	2 Wheel 2	3 Wheel 3	4 Comb.	5 10	6	7	8	9
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A	B	C	D	E	F	G	H	I	J

--	--	--	--

LABELS					FLAGS	SET STATUS		
A Play	B Winnings	C	D	E Set up	0	FLAGS	TRIG	DISP
a	b	c	d	e	1	ON OFF 0 ☐ ☒ 1 ☐ ☒ 2 ☐ ☒ 3 ☒ ☐	DEG ☒ GRAD ☐ RAD ☐	FIX ☒ SCI ☐ ENG ☐ n <u>3</u>
0	1 Cherry loop	2 Used	3	4	2			
5	6	7	8 Adds	9 End of play	3			

Submarine Hunt

001 *LBLC	057 RTN	Sets movement flag. 1 unit after sonar.	058 *LBLB	Sonar Reading
002 1	059 1		060 ST+8	
003 ST0C	061 R4	Sensitivity toggle switch 1.9→2.9→1.9 etc.	062 DSP0	
004 SF0	063 CF1		064 GSB0	
005 RTN	065 RCLC	Start	066 ST05	
006 *LBLD	067 F0?		068 GSB0	
007 RCLD	069 RCL3	Determine initial position using input seed.	070 RTN	Submarine Move Routine
008 1	071 *LBL0		072 GSB0	
009 X=Y?	073 4		074 X=Y	
010 +	075 X>Y?		076 GT00	
011 ST0D	077 RCL5	Fire Depth Charge	078 CHS	
012 RTN	079 GT01		080 *LBL0	
013 *LBLE	081 RCL5		082 *LBL1	
014 CLRS	083 ST06		084 GSB0	
015 CF0	085 5	Move Submarine	086 X<0?	
016 .	087 GT00		088 SF2	
017 5	089 RCL1	Display Hit!	090 GT01	
018 2	091 *LBL0		092 RCL2	
019 8	093 *LBL1		094 RCL6	
020 4	095 +		096 X<0?	
021 1	097 GT00		098 9	
022 6	099 X=Y		100 X<Y?	
023 X	101 GT01		102 *LBL0	
024 ST00	103 RCL6		104 2	
025 GSB0	105 X		106 -	
026 ST01	107 *LBL1		108 F2?	
027 GSB0	109 GT00		110 ST02	
028 ST02	111 GT01		112 *LBL0	
029 DSP0				
030 1				
031 ST0D				
032 CLX				
033 RTN				
034 *LBLA				
035 1				
036 ST+7				
037 R4				
038 SF1				
039 GSB0				
040 X#0?				
041 GT00				
042 1				
043 ST05				
044 GSB0				
045 RTN				
046 *LBL0				
047 9				
048 1/X				
049 DSP3				
050 PSE				
051 DSP5				
052 PSE				
053 DSP7				
054 PSE				
055 DSP9				
056 PSE				

REGISTERS									
0 Seed	1 P ₁	2 P ₂	3 Response	4 d	5 Used	6 Used	7 Used	8 Used	9
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A	B	C	0, 1	D	1 or 2	E		I	

Artillery

001 *LBLc	INITIALIZATION	057 EEX							
002 FIX		058 4							
003 DSP2		059 ÷							
004 .		060 X≠Y							
005 5		061 X>0?							
006 2		062 ST02							
007 8		063 3							
008 4		064 6							
009 1		065 0							
010 6		066 +							
011 3		067 *LBL2							
012 ST08	R0 ← seed = .5284163	068 DSP0							
013 3		069 RND							
014 ST0C	RC ← spot rating = 3	070 +							
015 5		071 DSP4	Display brg. range.						
016 0		072 RTN	-----						
017 0		073 *LBLb							
018 ST0B	RB ← speed = 500	074 ST0B	Set new SPEED.						
019 EEX		075 RTN	-----						
020 2		076 *LBLc							
021 ST0D	RD ← kill range = 100	077 ST0C	Set new spotter rating.						
022 X?		078 RTN	-----						
023 ST09	Max. gun range = 10000	079 *LBLd							
024 RTN	-----	080 ST0D	Set new KILL range.						
025 *LBLd	START NEW BATTLE	081 RTN	-----						
026 SPC		082 *LBL1							
027 0		083 CF0	Routine to calc. move-						
028 ST01	Rounds = 0	084 X0?	ment of target in N-S and						
029 RCL9		085 SF0	E-W directions.						
030 2		086 RCLB							
031 ÷		087 GSB0							
032 ENT†		088 F0?							
033 GSB0		089 CHS							
034 +	Generate target range.	090 -							
035 3		091 RTN	-----						
036 6		092 *LBLe							
037 0		093 *LBLF	FIRE! Routine - - -						
038 GSB0		094 ISZ†							
039 ST0B	Generate target bearing.	095 X≠Y							
040 R+		096 DSP7							
041 +R		097 PRTX	Output bearing.						
042 ST01	Save target N-S.	098 X≠Y							
043 X≠Y		099 DSF1							
044 ST02	Save target E-W.	100 PRTX	Output elevation.						
045 4		101 ENT†							
046 5		102 +							
047 ST≠8		103 SIN							
048 RCL8		104 RCL9							
049 DSP0	Display target 45° sector.	105 x							
050 RND		106 +R							
051 x		107 ST03	Save round's N-S.						
052 RTN		108 X≠Y							
053 *LBLC	-----	109 ST04	Save round's E-W.						
054 RCL2	Routine to generate and	110 RCL1							
055 RCL1	display target's last position.	111 GSB1							
056 +P		112 ST01	Move target N-S.						
REGISTERS									
0 Seed	1 Targ N-S	2 Targ E-W	3 Shell N-S	4 Shell E-W	5	6	7 Used	8 Used	9 Gun Range
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A	B	C	D	E	F	G	H	I	J
SPEED		Spot Rating		KILL Range				Rounds Fired	

Space War

001 *LBLA	Initialize.	057 SF2	
002 CLRC	If seed = 0, use π .	058 CLX	
003 X=0?		059 RCLC	
004 P1		060 X=Y?	
005 ST09		061 SF2	
006 1		062 CLX	
007 0		063 RCLD	
008 ST04		064 X=Y?	
009 2	Point to R_A and prepare to	065 SF2	Set flag 2 if position
010 0	compute random starting	066 R;	occupied.
011 ST01	positions.	067 RTN	-----
012 *LBL9	-----	068 *LBLC	Long Scan.
013 GSB1	Compute a position. Check	069 4	
014 GSB0	not occupied.	070 FIX	
015 F2?		071 9	
016 GT09		072 9	
017 ST01	Store position.	073 9	Store constant 0.00400400
018 ISZ1		074 +	
019 RCL1		075 ST02	
020 2		076 DSP8	
021 5	Exit loop after filling R_6 .	077 SPC	
022 X=Y?		078 RCLC	
023 GT09		079 INT	
024 EEX		080 ST00	
025 3		081 RCL4	
026 ST06	Energy = 1000.	082 +	Scan line above KH.
027 3		083 GSB3	
028 ST07	Torpedoes = 3.	084 RCL0	
029 ST09		085 GSB3	Scan line with KH.
030 1		086 RCL0	
031 0		087 RCL4	
032 ST08	Days = 18	088 -	Scan line below KH.
033 RCLC	Display position of	089 GSB3	
034 RTN	Kittyhawk.	090 RTN	-----
035 *LBL1	Generate one starting posi-	091 *LBL3	Routine scans one line,
036 RCL9	tion of the form QQ.SS,	092 RCL2	i.e., 3 quadrants.
037 9	where QQ is quadrant, SS	093 ST01	
038 9	is sector.	094 R4	
039 7		095 ST03	
040 x		096 ST+1	$R_1 \leftarrow QQ.00400400$,
041 FRC		097 1	where QQ refers to middle
042 ST09		098 -	quadrant.
043 EEX		099 GSB0	
044 4		100 GSB5	
045 x		101 GSB5	First quadrant.
046 INT		102 ST+1	
047 GSB5		103 RCL3	
048 GSB5		104 GSB0	
049 RTN		105 EEX	
050 *LBL8	-----	106 5	Middle quadrant.
051 RCLA	Routine tests to see if	107 +	
052 X=Y?	position in X-register is	108 ST+1	
053 SF2	already occupied.	109 RCL3	
054 CLX		110 1	
055 RCL6		111 +	
056 X=Y?		112 GSB0	

REGISTERS									
0 Used	1 Used	2 Used	3 Used	4 10	5 Used	6 Energy	7 Torpedoes	8 Days	9 Alglogs
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A Alglog 1	B Alglog 2	C Alglog 3	D Base	E Kittyhawk	F Pointer				

001 *LBLA	Short Scan.	057 CHS	
002 FIX		058 10*	
003 DSP9		059 X	3, 4, or 7 times 10**(-S _x)
004 SPC	A "3" marks KH,	060 ST+;	Add to register S _y .
005 P+S	"4" marks Alglog,	061 RTN	-----
006 9	"7" marks Base.	062 *LBLC	Torpedo.
007 STOI		063 CF0	Save angle.
008 0		064 ST08	
009 *LBL9		065 RCL7	
010 STOI	Clear R ₅₀ - R ₅₉ to hold	066 1	If no torpedoes remain,
011 DSZI	scans of rows 0 thru 9,	067 X>Y?	display "Error".
012 GT09	respectively.	068 GT08	Otherwise subtract one
013 ST08		069 -	from no. torpedoes.
014 3		070 ST07	
015 RCL5	Locate KH in quadrant.	071 1	
016 GSB0		072 9	
017 4		073 STOI	
018 RCLA	Check Alglog 1.	074 GSB1	
019 GSB0		075 GSB1	Check if Alglogs 1, 2, and
020 4		076 GSB1	3 are in the path of the
021 RCLB	Check Alglog 2.	077 RCL9	torpedo.
022 GSB0		078 CF0	Display no. Alglogs left.
023 4		079 RTN	-----
024 RCLC	Check Alglog 3.	080 *LBL1	Routine tests if Alglog
025 GSB0		081 ISZI	will be hit by torpedo.
026 7		082 F0?	If F0 set, an Alglog has
027 RCLD	Check for Base.	083 RTN	been hit - return.
028 GSB0		084 RCLi	If Alglog not in same
029 9		085 INT	quadrant as KH, return.
030 STOI		086 RCL5	
031 *LBL8	Print R ₅₉ , R ₄₈ , ..., R ₅₀ as	087 INT	
032 RCLi	rows 9, 8, ..., 0.	088 X#Y?	
033 PRTX		089 RTN	Find angle to Alglog.
034 DSZI		090 GSB7	
035 GT08		091 CLX	Compare to angle of fire.
036 RCL0		092 RCL0	
037 PRTX		093	
038 P+S		094 ABS	
039 RTN		095 1	If Alglog 1° or more away,
040 *LBL0		096 XZ?	no hit.
041 ENT1	Routine tests whether an	097 RTN	
042 INT	object is in KH's quadrant.	098 1	If hit, store -1 as Alglog's
043 RCL5		099 CHS	position.
044 INT	If not, return.	100 STOI	
045 X#Y?		101 SF0	F0 set to indicate torpedo
046 RTN		102 1	is spent.
047 R4		103 ST-9	Decrement no. Alglogs.
048 R4	If so, locate 3, 4, or 7 in	104 RTN	
049 FRC	proper sector of proper	105 *LBL7	
050 GSB5	row, represented by R ₅₀ -R ₅₉	106 RCLi	Routine finds angle and
051 INT		107 FRC	distance from KH to
052 STOI		108 GSB5	Alglog.
053 CLX		109 STOI	
054 LSTX		110 INT	
055 FRC		111 RCL5	
056 GSB5		112 FRC	

REGISTERS									
0 Used	1 Used	2	3	4 10	5	6 Energy	7 Torpedos	8 Days	9 Alglogs
S0 Row 0	S1 Row 1	S2 Row 2	S3 Row 3	S4 Row 4	S5 Row 5	S6 Row 6	S7 Row 7	S8 Row 8	S9 Row 9
A Alglog 1	B Alglog 2	C Alglog 3	D Base	E Kittyhawk	I Pointer				

112	GSB5			169	RCL6				
114	INT			170	INT				
115	-			171	RCL6				
116	RCL1			172	INT				
117	FRC			173	X≠Y?				
118	RCL6			174	GT06				
119	GSB5			175	RCL0				
120	FRC			176	FRC				
121	-			177	GSB5				
122	GSB5			178	ST06				
123	→P			179	INT				
124	RTN			180	RCL6				
125	*LBL0			181	FRC				
126	ST00			182	GSB5				
127	ST-6			183	ST01				
128	1			184	INT				
129	9			185	-				
130	ST01			186	ABS				
131	GSB4			187	2				
132	GSB4			188	X≠Y?				
133	GSB4			189	GT06				
134	RCL9			190	RCL0				
135	RTN			191	FRC				
136	*LBL4			192	GSB5				
137	RCL6			193	RCL1				
138	X<0?			194	FRC				
139	GT06			195	GSB5				
140	ISZ1			196	-				
141	RCL1			197	ABS				
142	INT			198	2				
143	RCL6			199	X≠Y?				
144	INT			200	EE%				
145	X≠Y?			201	3				
146	RTN			202	ST06				
147	EEX			203	3				
148	2			204	3				
149	ST-6			205	ST07				
150	GSB7			206	GT00				
151	X?			207	*LBL6				
152	EEX			208	RCL6				
153	2			209	RCL7				
154	+			210	*LBL0				
155	RCL0			211	1				
156	X≠Y?			212	0				
157	RTN			213	÷				
158	1			214	X≠Y				
159	CHS			215	INT				
160	ST01			216	+				
161	1			217	FIX				
162	ST-9			218	DSP1				
163	RTN			219	RTN				
164	*LBL6			220	*LBL5				
165	CLX			221	1				
166	PSE			222	0				
167	GT06			223	x				
168	*LBL6			224	RTN				

Return r in X, θ in Y.

Phaser.

Save energy and subtract from total.

Check if Alglogs 1, 2, and 3 are in range.

Display no. Alglogs left.

Test if Alglog hit by phaser.

If energy < 0, exit.

If Alglog not in KH's quadrant, return.

Deduct 100 for shields.

Find distance r to Alglog.

Energy input must be greater than r² + 100.

If hit, store -1 as Alglog's position.

Decrement no. Alglogs.

Out of energy – flash zeros.

If Base and KH not in same quadrant, display same En.Torp.

KH must be within 1 sector of Base in Y-direction.

Also must be within 1 sector of Base in X-direction.

If successful, get 1000 units energy and 3 torpedoes.

If failed, display old energy and torpedoes.

Form display Energy.Torpedoes.

Multiply by 10.

LABELS					FLAGS	SET STATUS		
A Short scan	B	C Torpedo	D Phaser	E Dock	0 Torpedo	FLAGS	TRIG	DISP
a→En.torp	b	c	d	e	1	ON OFF		
0 Used	1 Torp hit?	2	3	4 Phaser hit?	2	0 ☐ ☒	DEG ☒	FIX ☒
5 10x	6 Flash zeros	7 KH to Alglog	8 Print reg.	9 Clear reg.	3	1 ☐ ☒	GRAD ☒	SCI ☐
						2 ☐ ☒	RAD ☐	ENG ☐
						3 ☐ ☒		n 2

Super Bagels

001	*LBL0	Number of digits.	057	LSTX	----- Compare for equality. -----						
002	INT		058	LN							
003	X=0?		059	EEN							
004	GT00		060	7							
005	8		061	X							
006	X<Y		062	ABS							
007	X>Y?		063	D+R							
008	GT00		064	1/X							
009	RCLB		065	STOE							
010	FRC		066	R4							
011	+	067	*LBL0	----- Check for digits out of position. -----							
012	STOE	068	INT								
013	*LBLC	069	STOD								
014	INT	070	RCLC								
015	9	071	1/X								
016	X<Y	072	R+D								
017	X>Y?	073	SF1								
018	GT00	074	GSB1								
019	RCL4	075	CLX								
020	+	076	STO9								
021	RCLB	077	STOE	----- Check for digits out of position. -----							
022	INT	078	P+S								
023	+	079	RCLD								
024	GT00	080	CF1								
025	*LBL0	081	GSB1								
026	RCLC	082	RCLB								
027	RTN	083	STO1								
028	*LBLA	084	*LBL3								
029	SPC	085	RCL i								
030	CF0	086	P+S								
031	CLRG	087	RCL i								
032	X=0?	088	X#Y?								
033	P i	089	GT00								
034	STOE	090	CHS								
035	1	091	X<Y								
036	0	092	CHS								
037	STOA	093	X<Y								
038	4	094	RCLA								
039	.	095	ST+9								
040	5	096	R4								
041	*LBL0	097	*LBL0								
042	STOE	098	STO i								
043	FIX	099	X<Y								
044	DSP1	100	P+S								
045	RTN	101	STO i								
046	*LBL0	102	DSZ1								
047	CF2	103	GT03								
048	F0?	104	RCLB								
049	GT00	105	STOE								
050	ENT+	106	*LBL9								
051	SF0	107	RCL0								
052	RCLC	108	STO1								
053	RCLB	109	RCL i								
054	X	110	P+S								
055	X	111	X<0?								
056	X=0?	112	GT05								
REGISTERS											
0	1 Used	2 Used	3 Used	4 Used	5 Used	6 Used	7 Used	8 Used	9		
S0	Used	S1 Used	S2 Used	S3 Used	S4 Used	S5 Used	S6 Used	S7 Used	S8 Used	S9 Used	
A	Used	B	Used	C	Used	D	Used	E	Used	I	Used

113	SF2				169	RCLC			
114	RCLB				170	1			
115	STOI				171	+			
116	*LBL6				172	STO0			
117	CLX				173	DSF1			
118	RCL1				174	XZY			
119	N=Y?				175	RTN			Win?
120	GT00				176	F2?			
121	DSZ1				177	RTN			
122	GT07				178	XZY			
123	GT05				179	DSF0			
124	*LBL7				180	RTN			Output guesses to win.
125	CLX				181	RTN			-----
126	RCL1				182	*LBL1			Decompose number.
127	N=Y?				183	RCLB			
128	GT06				184	FRC			
129	DSZ1				185	RCLA			
130	GT08				186	1/X			
131	GT05				187	+			
132	*LBL8				188	STO0			
133	CLX				189	CLX			
134	RCL1				190	RCLB			
135	N=Y?				191	STOI			
136	GT00				192	INT			
137	DSZ1				193	10X			
138	GT06				194	+			
139	GT05				195	AES			
140	*LBL0				196	STOI			
141	CHS				197	*LBL2			
142	STOI				198	RCLS			
143	RCLA				199	FRC			
144	ST+0				200	RCLA			
145	*LBL5				201	X			
146	P+S				202	STOI			
147	CLX				203	RCL0			
148	STOI				204	XZY			
149	DSZ1				205	FI?			
150	GT09				206	X			
151	P+S				207	1			
152	RCLD				208	+			
153	RCLB				209	INT			
154	INT				210	STOI			
155	10X				211	DSZ1			
156	+				212	GT02			
157	FRC				213	RTN			
158	RCL0				214	*LBL6			
159	RCLA				215	STOI			
160	÷				216	STO2			
161	+				217	STO3			
162	RCL9				218	STO4			
163	+				219	STO5			
164	RCLB				220	STO6			
165	STOI				221	STO7			
166	CLX				222	STO8			
167	GSB#				223	RTN			
168	CLX								Clear registers for next guess.

LABELS					FLAGS		SET STATUS		
A Start	B #digits	C Max digit	D #guesses	E Guess	0 Used	FLAGS		TRIG	DISP
a Used	b	c	d	e Used	1	ON OFF		DEG ☒	FIX ☒
0 Used	1	2 Used	3 Used	4	2	0 ☐	GRAD ☐	SCI ☐	
						1 ☐	RAD ☐	ENG ☐	
5 Used	6 Used	7 Used	8 Used	9 Used	3	2 ☐			
						3 ☐			n <u>1</u>

Nim_k

001 *LBLA	L mode.	057 RCL5	Set I to right-most pile.						
002 3		058 STOI							
003 2		059 *LBL6							
004 CHS		060 GSB2							
005 STOI		061 X<Y?	Test for binary unit in						
006 *LBLD		062 ST+0	designated pile/column.						
007 DSP1		063 GSI1	Accumulate Σc_{ij} .						
008 PRTX	Print pile.# taken.	064 STOI	Cycle till zero.						
009 STOI		065 0							
010 INT		066 ST+0	Double R0 for Σc_{ij} .						
011 CHS		067 RCL0							
012 10*		068 RCL0							
013 RCLi		069 RCL0							
014 RCLi		070 1							
015 FRC		071 +							
016 1		072 X>Y?							
017 0		073 STOI							
018 *		074 3	$k < \Sigma c_{ij}$						
019 INT		075 RCL0							
020 X>Y?		076 ABS							
021 STO0	Error, taking more than in	077 X>Y?							
022 X=0?	pile.	078 SF0	Set F0 for Σc_{ij} mod						
023 STO0	Error, taking zero delete #	079 +	$(k + 1) = 0$.						
024 ST-i	from pile.	080 R↓							
025 R↑		081 *LBL0							
026 *	Shift by 10.	082 ENT1							
027 1		083 R↓							
028 +		084 ÷							
029 CHS		085 INT							
030 RCLA		086 R↑							
031 +	Adjust A.	087 *							
032 STOA		088 -	Σc_{ij} mod $(k + 1)$.						
033 RCLB		089 STOI							
034 STOI		090 RCLC							
035 DSPi	Readjust display.	091 X<0?	neg. = L mode						
036 RCLA		092 F0?							
037 PRTX	Print new combination.	093 GT00	F0 not set.						
038 RTN		094 3							
039 *LBLC		095 RCLC							
040 3		096 ABS							
041 2		097 X>Y?							
042 *LBL0		098 GT00							
043 DSP0		099 RCL0							
044 PRTX		100 -							
045 STOC	Initialize C to -32, L mode	101 1	Subtract 1 from Σc_{ij} .						
046 CF0	32, W mode.	102 X<0?	If neg, replace with k.						
047 CF1		103 RCL0							
048 *LBL9		104 STOI							
049 RCLC		105 *LBL0							
050 2		106 RCL0							
051 ÷	Get 2^j	107 X=0?	If $\Sigma c_{ij} = 0$ go to next						
052 STOC		108 GT09	column.						
053 ABS		109 1							
054 1		110 0							
055 X=Y?		111 STOI	Initialize pointer to RS0.						
056 GTOI	Exit if C = 1.	112 *LBL7							
REGISTERS									
0 Σc_{ij}	1 p_1	2 p_2	3 p_3	4 p_4	5 p_5	6 p_6	7 p_7	8 p_8	9 p_9
S0 k_1	S1 k_2	S2 k_3	S3 k_4	S4 k_5	S5 k_6	S6 k_7	S7 k_8	S8 k_9	S9
A $k, p_1 p_2 p_3 \dots$	B no. of piles		C 2^j		D 10, pointer		E k	F Used	

113	STOI		169	RCLF		k
114	RCLB		170	PzS		Clear pointer reg's.
115	RCLi		171	CLRG		
116	X=0?	If pointer empty, use right	172	PzS		
117	X≠Y	most pile no.	173	STOE		Restore E, C, A, B.
118	STOI		174	R+		
119	*LBL6		175	STOC		
120	SSB2	Test for binary unit.	176	R4		
121	X>Y?		177	STOA		
122	GT00		178	F4		
123	SF1	Set F1, at least one pile	179	STOE		
124	SF2	decreased.	180	STCI		
125	DSZi	Set F2, this pile decreased.	181	DSPi		Adjust display.
126	GT06	Step to next pile.	182	0		
127	*LBL0		183	*LBL4		
128	F2?		184	RCLi		Build display.
129	GT00		185	+		
130	DSZ1		186	i		
131	GT06		187	0		
132	*LBL0		188	÷		Shift by 10.
133	RCLB	Recall D to I.	189	DSZ1		
134	X≠1		190	STOA		
135	STOI	Store pile # in pointer.	191	RCLF		Recall k.
136	ISZ1	Increment I (D).	192	+		Add to combination.
137	CLX	0 to I.	193	STOA		
138	X≠1		194	SPC		Space and print display.
139	STOD	I to D.	195	PRTX		
140	DSZ1	Decrement Σcol by 1.	196	RTN		
141	STO7	Go to new pile or new	197	*LBL5		Entry routine.
142	STO9	column.	198	CLRG		Clear all registers.
143	*LBL2	Subroutine.	199	PzS		
144	RCLi	Objects in pile.	200	CLRG		
145	RCLC		201	STOA		Store input in A.
146	ABS		202	ENT↑		
147	÷	Divided by 2 ⁱ .	203	INT		
148	FRC		204	STOE		
149	.		205	-		Input fractional part.
150	5	Compare fractional part	206	*LBL3		
151	RTN	to 0.5.	207	ISZ1		
152	*LBL1		208	1		Shift by 10 to get
153	F1?	Any pile decreased, exit	209	0		individual pile #'s.
154	GT00	with no change.	210	x		
155	RCLB		211	ENT↑		
156	STOI		212	INT		
157	*LBL5		213	STCi		
158	RCLi		214	-		
159	DSZ1	Otherwise decrement right-	215	X≠0?		Continue till fractional
160	X>0?	most pile by 1.	216	GT03		part = 0.
161	GT00		217	RCLi		
162	STCi		218	DSPi		Set display.
163	DSZ1		219	STOB		
164	GT05		220	RCLA		
165	*LBL0	Exit routine.	221	PRTX		Print input.
166	RCLB	Recall no. piles.	222	RTN		
167	RCLA	Display.				
168	RCLC	2 ⁱ				

LABELS					FLAGS	SET STATUS		
A L mode	B	C W mode	D Delete	E Enter	0 L/W	FLAGS	TRIG	DISP
a	b	c	d	e	1 ≥ 1 pile	ON OFF	DEG ☒	FIX ☒
0 Used	1 Used	2 Used	3 Used	4 Used	2 this pile	0 ☐	GRAD ☐	SCI ☐
5 Used	6 Used	7 Used	8 Used	9 Used	3	1 ☐	RAD ☐	ENG ☐
						2 ☐		n <u>4</u>
						3 ☐		

Queen Board

001 *LBLA	Current position.	057 1							
002 ST01	R ₁	058 -							
003 GSBE		059 X=Y?	127 = R ₂ ?						
004 1		060 ST05							
005 X=Y?		061 1							
006 ST02		062 -							
007 7		063 X=Y?	126 = R ₂ ?						
008 ST01	7→R ₁	064 GT09							
009 *LBL1		065 5							
010 RCL1		066 1							
011 RCL1		067 -							
012 EEX		068 X=Y?	75 = R ₂ ?						
013 1		069 GT09							
014 X		070 2							
015 +		071 -							
016 ST02	10K + R ₁ →R ₂	072 X=Y?	73 = R ₂ ?						
017 GSBE	Position good?	073 GT05							
018 1		074 2							
019 X=Y?		075 9							
020 ST00	Yes, recall R ₂ .	076 -							
021 RCL1		077 X=Y?	44 = R ₂ ?						
022 ST+2	K + R ₂ →R ₂	078 GT09							
023 RCL2	Position good?	079 3							
024 GSBE		080 -							
025 1		081 X=Y?	41 = R ₂ ?						
026 X=Y?		082 GT05							
027 ST00	Yes, recall R ₂ .	083 RTN							
028 RCL1		084 *LBL5							
029 EEX		085 1							
030 1		086 RTN							
031 X									
032 ST+2	10K + R ₂ →R ₂								
033 RCL2									
034 GSBE	Position good?								
035 1									
036 X=Y?									
037 ST00	Yes, recall R ₂ .								
038 DSZ1									
039 ST01									
040 RCL1									
041 *LBL2									
042 EEX									
043 1	Default move.								
044 ST+1	10 + R ₁ →R ₁								
045 RCL1									
046 RTN									
047 *LBL0									
048 RCL2									
049 RTN									
050 *LBL5	Test for good position.								
051 1									
052 5									
053 6									
054 X=Y?	158 = R ₂ ?								
055 ST09									
056 3									
REGISTERS									
0	1 Used	2 Used	3	4	5	6	7	8	9
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A	B	C	D	E	F	G	H	I	K

LABELS										FLAGS					SET STATUS										
A	Used		B			C			D			E	Used		0			FLAGS			TRIG			DISP	
a			b			c			d			e			1			ON OFF			DEG ☒			FIX ☒	
																		☐ ☒			GRAD ☐			SCI ☐	
0	Used		1	Used		2	Used		3			4			2			☐ ☒			RAD ☐			ENG ☐	
5			6			7			8			9	Used		3			☐ ☒						n <u>0</u>	

Hexapawn

001 *LBL0	Machine first.	057 *LBL9							
002 8		058 P;							
003 3		059 +							
004 8	Configurations for move 1.	060 ST06							
005 8		061 ST09	-----						
006 6		062 *LBL4	Move						
007 0		063 ST08							
008 7		064 RCL4							
009 ST01		065 ST09							
010 3		066 RCL5							
011 1	Configurations for move 2.	067 ST07							
012 3		068 2							
013 3		069 RCL8							
014 5		070 YP							
015 8		071 ST05							
016 3		072 3							
017 ST02		073 ST01	-----						
018 3	Configurations for move 3.	074 RCL6	Generate a random						
019 4		075 P;	number between 1 and 3.						
020 3		076 +							
021 1		077 X2							
022 4		078 FRC							
023 ST03		079 ST06							
024 ST00	-----	080 3							
025 *LBL0	Player first.	081 X							
026 1		082 1							
027 6		083 +							
028 7	Configurations for move 1.	084 INT	-----						
029 7		085 ST04	BEGIN loop 1						
030 7		086 *LBL1							
031 2		087 3							
032 1		088 RCL4							
033 5		089 1	INCREMENT trial move						
034 ST01		090 +							
035 1		091 X=YP							
036 6	Configurations for move 2.	092 1							
037 7		093 ST04							
038 5		094 X=1							
039 6		095 RCL1							
040 7		096 X=V							
041 3		097 X=1							
042 5		098 R4							
043 ST02	Configurations for move 3.	099 RCL5							
044 5		100 ÷							
045 2		101 FRC							
046 4		102 .							
047 4		103 5							
048 1		104 X=YP?	IF trial move OK						
049 3		105 GT04	THEN exit						
050 ST03		106 DSZ1	IF less than 3 tries						
051 *LBL0		107 GT01	THEN REPEAT loop 1						
052 0		108 RCL7	ELSE no move possible.						
053 ST04		109 ST05							
054 ST05		110 ST04							
055 ST06		111 CLX							
056 ST07		112 RTN							
REGISTERS									
0	1 Moves #1	2 Moves #2	3 Moves #3	4 Trial move	5 21(Config)	6 Random	7 21(Last conf)	8 This move	9 Last move
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A	B	C	D	E	I				

113	#LBL4																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
-----	-------	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Tic-Tac-Toe

001 *LBLA	057 STOB								
002 2	058 INT								
003 STOE	059 RTN								
004 1	060 *LBLB								
005 CHS	061 STOA								
006 STOE	062 STOI								
007 2	063 RTN								
008 GSBb	064 *LBLB								
009 GSBb	065 1								
010 GSBb	066 ST+0								
011 GSBb	067 RCL4								
012 RCL1	068 RTN								
013 RCL1	069 *LBLJ								
014 DSP9	070 RCL1								
015 R/S	071 CHS								
016 GSBb	072 10*								
017 GSBb	073 2								
018 GSBb	074 X								
019 RCL1	075 RCL1								
020 INT	076 +								
021 RCL1	077 STOE								
022 STOB	078 RCL1								
023 *LBL0	079 RTN								
024 +	080 *LBL1								
025 GSBb	081 EEX								
026 STOI	082 6								
027 GSBb	083 GSB9								
028 GSBb	084 EEX								
029 RCL1	085 3								
030 DSP9	086 GSB9								
031 RCL1	087 1								
032 +	088 GSB9								
033 R/S	089 SFC								
034 *LBL1	090 RTN								
035 GSBb	091 *LBL9								
036 GSBb	092 DSP3								
037 GSBb	093 RCL1								
038 RCL1	094 X								
039 GSBb	095 FRC								
040 X=Y?	096 RCL0								
041 GSBb	097 +								
042 STOI	098 PRTH								
043 GSBb	099 DSP0								
044 GSBb	100 RTN								
045 RCL1	101 *LBLD								
046 DSPS	102 RCL1								
047 RCL1	103 CHS								
048 +	104 10*								
049 R/S	105 RCL1								
050 STOE	106 +								
051 *LBLB	107 STOE								
052 RCLB	108 RCL1								
053 FRC	109 RTN								
054 1	110 *LBLA								
055 0	111 .								
056 X	112 5								
Start a new game. Check for the 1st move of player. Check for 2, 3, 4th move of player and decide HP's move. Check for next move. Store 2 for HP. Decode for printing. Print. Store 1 for player. Store constants for move sequences.									
REGISTERS									
0 play no.	1 .5873649	2	3 .5891467	4 .13598	5 .1374698	6 .31578	7 .13589	8 .3175964	9 .31587
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A Temp.	B Last X of LBL B		C	D	E Used	I 1, 2, ..., 9			

113	6			159	3									
114	7			170	1									
115	3			171	5									
116	6			172	8									
117	4			173	7									
118	5			174	ST05									
119	ST01			175	0									
120	.			176	PTN									
121	5													
122	8													
123	9													
124	1													
125	4													
126	6													
127	7													
128	ST03													
129	.													
130	1													
131	3													
132	5													
133	9													
134	8													
135	ST04													
136	.													
137	1													
138	3													
139	7													
140	4													
141	6													
142	9													
143	8													
144	ST05													
145	.													
146	3													
147	1													
148	5													
149	7													
150	8													
151	ST06													
152	.													
153	1													
154	3													
155	5													
156	8													
157	9													
158	ST07													
159	.													
160	3													
161	1													
162	7													
163	5													
164	9													
165	6													
166	4													
167	ST08													
168	.													
LABELS					FLAGS	SET STATUS								
A	New Game	B	next move	C		D		E	2 nd move	0		FLAGS	TRIG	DISP
a		b	St to A, I	c		d	2 for HP	e	Decode	1		ON OFF		
0	1 st move	1		2		3		4		2		0 ☐ ☒	DEG ☒	FIX ☒
5		6		7		8	Increase	9	Print	3		1 ☐ ☒	GRAD ☐	SCI ☐
												2 ☐ ☒	RAD ☐	ENG ☐
												3 ☐ ☒		n <u>0</u>

Wari

001 *LBLA	Move player 1	057 GTO	
002 CHS		058 GTD	
003 7		059 *LBLB	-----
004 +		060 SPC	Display board
005 SF1		061 6	
006 STOS		062 STOI	Initialize
007 *LBLC	Move player 2	063 *LBL3	I=6
008 6		064 RCLi	Begin loop 3
009 +		065 1	
010 OF1		066 3	
011 *LBL9		067 ROLI	
012 STOC	Initialize move loop	068 -	
013 STOI		069 STOI	
014 RCLi		070 CLX	
015 X=0?		071 +	
016 +		072 RCLi	
017 STOD		073 EEX	
018 0		074 2	
019 STOI		075 +	
020 *LBL1	Begin loop 1	076 +	$D = C(I) + \frac{C(13 - I)}{100}$
021 DSZ1	I=I - 1	077 PRTH	Print D
022 STOD	If I = 0	078 RCLi	
023 1	Then I=12	079 1	
024 2		080 -	
025 STOI		081 -	
026 *LBL8		082 CHS	
027 RCLi		083 STOI	
028 RCLC		084 DSZ1	I=I - 1
029 X=Y?	If I = original bin	085 GTD	If I > 0 then repeat loop 3
030 GTOI	then skip to next bin	086 SPC	
031 1	else	087 RTH	-----
032 ST+1	increment bin	088 *LBLD	Display score
033 RCLD	decrement # seeds	089 RCL4	
034 1		090 RCLB	
035 -		091 EEX	
036 STOD		092 2	
037 X=0?	If seeds used up	093 +	
038 GTOS	then try to capture	094 +	
039 GTOI	else repeat loop 1	095 RTN	-----
040 *LBLC	Initialize	096 *LBL5	CAPTURE
041 1		097 RCLi	
042 2		098 6	
043 STOI		099 -	
044 4		100 5	
045 *LBL6		101 -	
046 STOI		102 RCLC	
047 DSZ1		103 LSTX	
048 GTOD		104 -	
049 0		105 X	
050 STOA		106 XD0?	If start & finish on same
051 STOB		107 GTOD	side
052 *LBL6	-----	108 *LBL7	Then no capture
053 GSB8	Display board	109 2	Else begin loop 7
054 F0?		110 RCLi	
055 GTOD	Display score	111 X=Y?	
056 FI?		112 GTOD	If bin has 2 or 3 Then increment appro-
REGISTERS			
0	1 Bin 1	2 Bin 2	3 Bin 3
			4 Bin 4
			5 Bin 5
			6 Bin 6
			7 Bin 7
			8 Bin 8
			9 Bin 9
S0	S1 Bin 10	S2 Bin 11	S3 Bin 12
			S4
			S5
			S6
			S7
			S8
			S9
A Score 1	B Score 2	C Current bin	D # seeds
			E Random
			I Pointer & Counter

113	3				169	INT			
114	X=Y?				170	STO6			
115	STO6				171	6			
116	*LBL7				172	+			
117	F10				173	STO1			
118	GT09				174	RCL1			
119	RCL8				175	X=0?			
120	+				176	GT06			
121	STO8				177	RCL6			
122	STO7				178	PRTX			
123	*LBL9				179	GSBE			
124	RCL4				180	RTN			
125	+				181	*LBL4			
126	STO4				182	RCL1			
127	*LBL7				183	STO6			
128	0				184	!			
129	STO1				185	2			
130	ISZ1				186	STO1			
131	7				187	*LBL4			
132	RCL1				188	RCL1			
133	X=Y?				189	X=0?			
134	STO6				190	STO8			
135	1				191	!			
136	3				192	!			
137	X=Y?				193	÷			
138	STO6				194	FRC			
139	STO7				195	1			
140	*LBL6				196	!			
141	1				197	X			
142	STO1				198	RCL1			
143	*LBL2				199	RCL6			
144	RCL1				200	-			
145	X=0?				201	X=Y?			
146	GT06				202	GT09			
147	3				203	*LBL8			
148	X>Y?				204	DSZ1			
149	GT04				205	RCL1			
150	*LBL6				206	6			
151	ISZ1				207	X=Y?			
152	RCL1				208	GT08			
153	7				209	GT04			
154	X=Y?				210	*LBL8			
155	GT02				211	RCL6			
156	*LBL6				212	STO7			
157	RCL6				213	GT06			
158	P1				214	*LBL9			
159	+				215	RCL1			
160	8				216	6			
161	Y*				217	-			
162	FRC				218	PRTX			
163	STO6				219	GSBE			
164	6				220	RTN			
165	X								
166	1								
167	+								
168	→P								

LABELS					FLAGS	SET STATUS		
A Player 1	B Display	C Start	D Score	E Player 2	0 P1 vs. P2	FLAGS	TRIG	DISP
a	b Random move	c Random loop	d Loop	e Auto player 2	1 Player 1	ON OFF		
0 Loop	1 Loop	2 Loop	3 Loop	4 Used	2	0 ☐ ☐	DEG ☒	FIX ☒
5 Used	6 Used	7 Used	8 Used	9 Used	3	1 ☐ ☒	GRAD ☐	SCI ☐
						2 ☐ ☒	RAD ☐	ENG ☐
						3 ☐ ☒		n <u>2</u>

Racetrack

001 *LBL0	START	057 ISZ1							
002 CLR0		058 ISZ1							
003 P+S		059 RCL1							
004 CLR0		060 +							
005 1		061 ST01	Increment v _x						
006 7		062 SSB2							
007 ST01	Initialize loop 9 -----	063 DSZ1							
008 *LBL9		064 DSZ1							
009 RCL1		065 RCL1							
010 1		066 +							
011 -		067 ST01	Increment x						
012 4	Compute starting position	068 XZY							
013 ÷		069 ISZ1							
014 5		070 ISZ1							
015 ×		071 ISZ1							
016 5		072 RCL1							
017 5		073 +							
018 +		074 ST01	Increment v _y						
019 ST01		075 SSB2							
020 RCL1	Decrement pointer	076 DSZ1							
021 4		077 DSZ1							
022 -		078 RCL1							
023 ST01		079 +							
024 X>0?	While pointer is positive, repeat loop 9	080 ST01	Increment y -----						
025 GT09		081 RCL0							
026 RTN	Identify cars	082 ST01							
027 *LBLA		083 RCL1	Initialize loop 8						
028 1		084 ST00							
029 GT00		085 ISZ1							
030 *LBLB		086 RCL1							
031 2		087 ST00							
032 GT00		088 1							
033 *LBLC		089 7							
034 3		090 ST01							
035 GT00		091 *LBL0	Test for crash -----						
036 *LBLD		092 RCL0							
037 4		093 X=Y?	If same car						
038 GT00		094 GT00	Then skip test						
039 *LBLE		095 RCL1							
040 5		096 RCL0							
041 *LBL0		097 -							
042 PRTX	Print car ID	098 ISZ1							
043 1	Compute a pointer	099 RCL1							
044 -		100 RCL0							
045 4		101 -							
046 ×		102 +P							
047 1		103 2							
048 +		104 X≠Y?							
049 ST00		105 DSZ1							
050 ST01		106 X≠Y?	If distance ≥ 2						
051 R+		107 GT00	Then no crash						
052 9		108 RCL1							
053 X≠Y?	If lvi exceeds 9	109 2							
054 XZY	Then use 9 instead	110 -							
055 R+		111 4							
056 +R		112 ÷							
REGISTERS									
0 Pointer	1	2	3	4	5	6	7	8	9
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A	B	C	D	E	F	G	H	I	J
	Used	Used						Used	

113	1		169	GT06		THEN don't penalize
114	+		170	*LBL5		PENALIZE
115	PSE	Display ID of car in crash	171	RCL0		
116	PSE		172	2		
117	PSE		173	+		
118	0	Set v_y and v_x to zero for both cars	174	STOI		Set v_y and v_x to zero
119	ISZI		175	0		
120	STOI		176	STOI		
121	ISZI		177	ISZI		
122	STOI		178	STOI		
123	RCL0	Save pointer	179	*LBL6		Set up display
124	XZ1		180	RCL0		
125	XZY		181	3		
126	ISZI		182	+		
127	ISZI		183	STOI		
128	STOI		184	RCL1		
129	ISZI		185	DSZI		
130	STOI		186	RCL1		
131	XZY		187	DSZI		
132	3		188	+F		
133	-		189	RCL1		
134	XZ1	Restore pointer	190	DSZI		
135	*LBL0		191	RCL1		
136	RCL1	Decrement pointer	192	PRST		
137	4		193	RTN		
138	-		194	*LBL2		Subroutine to compute ΔS
139	STOI		195	LSTX		
140	XO00	WHILE pointer is positive repeat loop 8	196	+		
141	GT08		197	2		
142	RCL6	CHECK INSIDE FENCE	198	÷		
143	5		199	RTN		
144	0		200	*LBL7		Subroutine to compute $y^{2.5}$
145	÷		201	ABS		
146	GSB7		202	2		
147	RCL0		203	.		
148	7		204	5		
149	0		205	YX		
150	÷		206	RTN		
151	GSB7		207	*LBL6		Display position
152	+		208	PRTX		ID → y, x
153	1		209	1		
154	XOY0	If off track	210	-		
155	GSB5	Then penalize	211	4		
156	RCL6	CHECK OUTSIDE FENCE	212	x		
157	0		213	2		
158	0		214	+		
159	÷		215	STOI		
160	GSB7		216	RCL1		
161	RCL0		217	PRTX		
162	EE%		218	DSZI		
163	2		219	RCL1		
164	÷		220	PRTX		
165	GSB7		221	RTN		
166	+					
167	:					
168	XOY0	If on track				

LABELS					FLAGS	SET STATUS		
A 1	B 2	C 3	D 4	E 5	0	FLAGS	TRIG	DISP
a ID→y,x	b	c START	d	e	1	ON OFF	DEG ☒	FIX ☒
0 Used	1	2 Subroutine	3	4	2	1 ☐ ☒	GRAD ☐	SCI ☐
5 Off track	6 Display	7 Subroutine	8 Crash loop	9 Start loop	3	2 ☐ ☒	RAD ☐	ENG ☐
						3 ☐ ☒		n <u>2</u>

Teaser

001 *LBL0	Initialize	057 STX5	Set up display						
002 1		058 GTOD							
003 ST05		059 *LBL7							
004 CH5		060 STX4							
005 ST01		061 STX5							
006 ST02		062 STX7							
007 ST03		063 STX6							
008 ST04		064 GTOD							
009 ST06		065 *LBL8							
010 ST07		066 STX7							
011 ST08	Store move	067 STX8							
012 ST09		068 STX9							
013 ST0B		069 GTOD							
014 :		070 *LBL9							
015 0		071 STX5							
016 ST0A		072 STX6							
017 GTOD		073 STX8							
018 *LBLA		074 STX9							
019 ST01		075 *LBLD							
020 RCL1		076 F0?							
021 X<0?	Generate a negative 1 or display "Error"	077 SPC							
022 GT06		078 FIX							
023 CH5		079 DSF3							
024 GT01		080 RCL9							
025 *LBL1		081 X<0?							
026 STX1		082 0							
027 STX2		083 RCLA							
028 STX4		084 ÷							
029 STX5		085 RCL8							
030 GTOD		086 X<0?							
031 *LBL2	Go to appropriate routine to change board	087 CLX							
032 STX1		088 +							
033 STX2		089 RCLA							
034 STX3		090 ÷							
035 GTOD		091 RCL7							
036 *LBL3		092 X<0?							
037 STX2		093 CLX							
038 STX3		094 +							
039 STX5		095 RCLA							
040 STX6		096 ÷							
041 GTOD		097 F0?							
042 *LBL4		098 PRTX							
043 STX1		099 EEX							
044 STX4		100 6							
045 STX7		101 ÷							
046 GTOD		102 ST0B							
047 *LBL5		103 RCL6							
048 STX2		104 X<0?							
049 STX5		105 0							
050 STX8		106 RCLA							
051 STX4		107 ÷							
052 STX6		108 RCL5							
053 GTOD		109 X<0?							
054 *LBL6		110 CLX							
055 STX3		111 +							
056 STX6		112 RCLA							
REGISTERS									
0 Used	1 ±1	2 ±1	3 ±1	4 ±1	5 ±1	6 ±1	7 ±1	8 ±1	9 ±1
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
A 10	B # of moves		C		D		E		I

113	÷				
114	RCL4				
115	X<0?				
116	CLX				
117	+				
118	RCL4				
119	÷				
120	F0?				
121	PRTX				
122	EEX				
123	3				
124	÷				
125	ST+0				
126	RCL3				
127	X<0?				
128	0				
129	RCL4				
130	÷				
131	RCL2				
132	X<0?				
133	CLX				
134	+				
135	RCL4				
136	÷				
137	RCL1				
138	X<0?				
139	CLX				
140	+				
141	RCL4				
142	÷				
143	F0?				
144	PRTX				
145	RCL0				
146	+				
147	9				
148	÷				
149	RCLB				
150	1				
151	+				
152	STOB				
153	10*				
154	x				
155	SCI				
156	DSP9				
157	RTN				
158	*LBL E				
159	F0?				
160	GT0e				
161	SF0				
162	1				
163	RTN				
164	*LBL e				
165	CF0				
166	0				
167	RTN				

LABELS					FLAGS	SET STATUS			
A	B	C	D	E	0	True/False	FLAGS	TRIG	DISP
Shoot		Start	Print	Print?	0	True/False	ON OFF	DEG	FIX
a		c	d	e	1		0 ☒ ☐	GRAD ☐	SCI ☒
0	Used	2	Used	4	Used	2	1 ☐ ☒	RAD ☐	ENG ☐
5	Used	6	Used	8	Used	3	2 ☐ ☒		n <u>9</u>
		7	Used	9	Used	3	3 ☐ ☒		

113	3			169	XZY	x'
114	6			170	RCL3	x_{n-1}
115	+			171	+	
116	JX			172	ST03	x_n
117	x			173	CHS	
118	STOB	θ_c -----		174	XZY	
119	RCL3	Is ball holed?		175	CHS	
120	ABS			176	+P	$-y_n, -x_n \rightarrow D_{h_n}, \phi_n$
121	RCL0	W		177	ST04	D_{h_n}
122	XZY?	$W \leq x_{n-1} ?$		178	XZY	ϕ_{h_n}
123	GSB9	Yes—in woods.		179	ST05	Y_c
124	RCL4	No—escape woods.		180	RCL6	θ_c
125	4			181	RCLB	y_n
126	+			182	RCL2	x_n
127	RCL6	Y_c		183	RCL3	Print Y_c, θ_c, y_n, x -----
128	XZY?	$Y_c > D_h + 4?$		184	FRST	In woods?
129	GT04	Yes—missed hole.		185	ABS	W
130	RCL4	No—in hole?		186	RCL0	$W \leq x ?$
131	5			187	XZY?	Yes: In woods
132	1/X			188	ST06	No: No woods -----
133	-			189	GT07	Woods display -----
134	XZY?	$D_h - .2 > Y_c?$		190	*LBL8	Print D_h, ϕ_h
135	GT04	Yes—missed hole.		191	DSP6	D_h
136	LSTX	No—in hole?		192	*LBL7	Print D_h
137	RCL4	D_h		193	RCL4	ϕ_h
138	÷			194	PRTX	Print ϕ_h
139	TAN ⁻¹	γ : hole window		195	RCL5	
140	RCLB	θ_c		196	PRTX	
141	ABS			197	DSP8	
142	XZY?	$ \theta_c > \gamma?$		198	RTN	
143	GT04	Yes—missed hole.		199	*LBL9	
144	0	No—in hole!		200	RCL6	In woods.
145	DSP9			201	ABS	θ_c
146	PRTX	Print 0.000000000.		202	RCL5	
147	RCL1	$S' = \text{total strokes.}$ -----		203	ABS	ϕ_h
148	RCL7	H		204	STN	$\sin \theta $
149	1			205	3	
150	8			206	x	$3 \sin \phi $
151	÷	H/18		207	RCL3	x
152	RCL0	h		208	ABS	
153	x	h (H/18)		209	RCL0	W
154	-	$S' - h$ (H/18)		210	-	$ x - W$
155	RCL9	Σ Par		211	÷	$3 \sin \phi / (W - x)$
156	-	$[S' - h(H/18)] - \Sigma$ Par		212	TAN ⁻¹	$\beta = \text{woods window}$
157	DSP8			213	ABS	
158	PRTX	Print S.		214	XZY?	$ \beta > \theta ?$
159	RTN			215	RTN	Yes—escape woods
160	*LBL4	Missed hole.		216	RCL6	No—hit trees
161	RCLB	θ_{cn}		217	9	
162	RCL5	$\phi_{h\ n-1}$		218	÷	
163	+			219	ST0E	$Y_c = Y_c' / 9$
164	RCL6	Y_{cn}		220	RCLB	θ'
165	+R	$Y_c, \theta + \phi \rightarrow y', x'$		221	9	
166	RCL2	y_{n-1}		222	x	
167	+	y_n		223	ST0B	$\theta = 9(\theta')$
168	ST02			224	RTN	

LABELS					FLAGS	SET STATUS		
A HOLE	B WOOD	C IRON	D PUTTER	E	0	FLAGS		DISP
a	1	2	3	4	5	ON OFF		
0	1	2	3	4	5	0	☐ ☐ ☐	DEG ☐ FIX ☐
5	6	7	8	9	0	1	☐ ☐ ☐	GRAD ☐ SCI ☐
						2	☐ ☐ ☐	RAD ☐ ENG ☐
						3	☐ ☐ ☐	n <u>0</u>

The Dealer

001 *LBLE	Deal n cards.	057 2	Place selected card at bottom of deck.									
002 SPC		058 *										
003 *LBLE		059 INT										
004 STOA		060 RCLC										
005 GSBW		061 -										
006 PRTX		062 EEX										
007 RCLA		063 2										
008 1		064 ÷										
009 -		065 *										
010 X>0?		066 ST-1										
011 GTOE		067 RT										
012 RTN		068 STOI										
013 *LBLE		069 LSTX										
014 RCLB		070 RCLD										
015 9		071 ÷										
016 9		072 ST+1										
017 7		073 RCLC										
018 *		074 1										
019 FRC		075 -										
020 STOB		076 STOC										
021 RCLC	077 X(0?	If all cards have been dealt, shuffle.										
022 *	078 GSEC											
023 INT	079 RCLC		If bingo flag is set, go to zero.									
024 5	080 *LBLE5											
025 ÷	081 F1?											
026 STOI	082 STOB			Calculate suit.								
027 FRC	083 1											
028 1	084 3											
029 0	085 ÷											
030 *	086 1											
031 10*	087 +											
032 STOD	088 INT											
033 RCL1	089 LSTX				Calculate card.							
034 *	090 FRC											
035 FRC	091 1											
036 EEX	092 3											
037 2	093 *											
038 *	094 1											
039 INT	095 *											
040 STOE	096 5											
041 RCL1	097 +											
042 RCLC	098 INT											
043 5	099 10*											
044 ÷	100 *	Combine for output.										
045 STOI	101 SCI											
046 FRC	102 RTN											
047 1	103 *LBLE		Bingo output.									
048 0	104 1											
049 *	105 +											
050 10*	106 FIN											
051 1/X	107 RTN											
052 LSTX	108 *LBLE											
053 RCL1	109 2			Shuffle: Store 51 as number in deck for cards or 74 for bingo.								
054 *	110 3											
055 FRC	111 ENT1											
056 EEX	112 5											
REGISTERS												
0 .000102...	1 .050607...				2 .101112...	3 .151617...	4 .202122...	5 .252627...	6 .303132...	7 .353637...	8 .404142...	9 .454647...
S0 .505152...	S1 .555657...				S2 .606162...	S3 .656667...	S4 .707172...	S5 .757677...	S6 .808182...	S7 .858687...	S8 .909192...	S9 .959697...
A #Cards	B Seed				C Deck Point	D Used	E Card	F	G	H Hole		

113	I	169	6	
114	F1?	170	3	
115	+	171	STOB	-----
116	STOC	172	GSBc	Store number of cards.
117	CLX	173	RTN	
118	RTN	174	STOB	Store user seed.
119	*LBLc	175	RTN	-----
120	SF1	176	*LBLc	Set registers for review
121	GTOb	177	SPC	of cards dealt.
122	*LBLc	178	RCLC	
123	CF1	179	GSBc	
124	*LBLc	180	R1	
125	I	181	RCLC	
126	8	182	STOB	
127	STOI	183	X*Y	-----
128	.	184	STOC	Pull values out of deck
129	0	185	*LBL4	in order previously dealt
130	5	186	RCLD	for review.
131	0	187	5	
132	5	188	+	
133	0	189	STOI	
134	5	190	FRC	
135	0	191	1	
136	5	192	0	
137	0	193	X	
138	5	194	10*	
139	CHS	195	RCLi	
140	ENT↑	196	X	
141	ENT↑	197	FRC	
142	ENT↑	198	EEX	
143	.	199	2	
144	8	200	X	
145	5	201	INT	
146	9	202	GSB5	
147	6	203	PRT%	
148	9	204	RCLC	-----
149	7	205	RCLD	Is review complete?
150	9	206	1	
151	8	207	-	
152	9	208	STOB	
153	9	209	X*Y?	
154	STOI	210	GT04	-----
155	PZS	211	R4	Display last value from
156	*LBL7	212	R4	review.
157	+	213	RTN	-----
158	STOI			
159	DSZ1			
160	GTCT			
161	+			
162	STOB			
163	.			
164	5			
165	2			
166	8			
167	4			
168	1			

LABELS					FLAGS	SET STATUS		
A → Hit	B	C n → Deal	D	E Review	0	FLAGS	TRIG	DISP
a Cards; seed	b	c Shuffle	d	e Bingo; seed	1 Bingo	ON OFF		
0 Used	1	2	3	4 Review	2	0 ☐ ☒	DEG ☒	FIX ☒
5 Suit/Card	6 Deal	7 Start	8	9	3	1 ☐ ☒	GRAD ☐	SCI ☐
						2 ☐ ☒	RAD ☐	ENG ☐
						3 ☐ ☒		n 0

Bowling Scorekeeper

001 *LBL0	Start	057 *LBL6	Spare
002 CLRG	Clear primary registers.	058 *LBLE	Strike
003 P=S		059 INT	Remove extra digits.
004 S		060 +	
005 STOI		061 1	
006 EEX		062 +	
007 4		063 STOI	Store and decrement for indirect address.
008 CHS		064 DSZ1	NOP
009 *LBL9		065 CF0	
010 STOI	Store .0001 in secondary registers 0 to 9.	066 P=S	Recall secondary register.
011 DSZ1		067 RCL1	
012 GT09		068 P=S	
013 STOI		069 ENT1	
014 CLX		070 INT	
015 RTN		071 1	10 pins
016 *LBLA	Entry for other than spare or strike.	072 0	
017 INT		073 ENT1	
018 LSTX		074 *LBL0	
019 FRC		075 R1	Pins down
020 ENT1		076 ST0A	New CS to A
021 ENT1		077 R1	
022 1		078 GSB1	Shift, get S1
023 0		079 ST0B	S1 to B
024 X		080 GSB1	Shift, get S2
025 FRC		081 ST0C	S2 to C
026 X#0?	Error if extra digits in entry e.g. 1.08.	082 GSB1	Shift, get PS
027 ST0B		083 ST0D	PS to D
028 R1		084 RCL1	Recall primary register.
029 1		085 CF1	Clear 2 nd ball flag.
030 0		086 X#0?	If reg. neg. set 2 nd ball flag.
031 +		087 SF1	Remove sign.
032 +	Original entry	088 ABS	
033 STOI	Store and decrement for indirect address.	089 EEX	
034 DSZ1	NOP	090 2	
035 CF0		091 X	
036 FRC		092 INT	
037 EEX		093 EEX	
038 2		094 2	
039 X	Pin count	095 +	
040 P=S		096 ST0E	SSS.FS to E
041 RCL1	Recall secondary register (CS.S1S2 PS).	097 RCLC	
042 P=S		098 X#0?	If strike 2 balls back GSB2.
043 ENT1		099 GSB2	Strike 1 ball back now 2 balls back.
044 INT	CS = current sum	100 RCLB	No strike 1 ball back.
045 ENT1		101 ST0C	
046 R1		102 0	
047 +	New CS	103 ST0B	
048 1		104 F1?	
049 1		105 GT03	If 2 nd ball GTO 3
050 X>Y?	Less than 11?	106 RCLD	Add count
051 GT0B	Continue	107 X#0?	
052 *LBL8	else error	108 GSB2	
053 ISZ1		109 0	
054 RCL1		110 ST0D	0 to PS
055 GSB0		111 RCLA	
056 GT0L		112 1	
REGISTERS			
0 Used	1 Used	2 Used	3 Used
4 Used	5 Used	6 Used	7 Used
8 Used	9 Used		
S0 Used	S1 Used	S2 Used	S3 Used
S4 Used	S5 Used	S6 Used	S7 Used
S8 Used	S9 Used		
A CS	B S1	C S2	D PS
E SSS.FS	F	G	H ID.P

113	0			169	2				
114	X#Y?	Strike?		170	X				
115	GT04	No strike		171	INT				NF.
116	ST06	Store strike		172	RCL I				
117	GT05			173	FRC				Last ball
118	*LBL4	2 nd ball		174	+				NF.LB
119	SF1			175	EEX				
120	GT06			176	4				
121	*LBL3	Spare?		177	÷				.00NFLB
122	CF1			178	RCL E				
123	RCL A			179	+				SSS.FSNFLB
124	1			180	F1?				If flag set.
125	0	CS # 10?		181	CHS				Make negative.
126	X#Y?			182	ST0 i				Update print register.
127	GT07			183	PRTX				Print display.
128	ST0D			184	RTN				
129	GT05	Add count		185	*LBL1				Subroutine to shift and
130	*LBL7			186	-				take integer part.
131	GSB2			187	EEX				
132	*LBL5			188	2				
133	0			189	X				
134	ST0A	0 to CS		190	ENT1				
135	RCL i	Increment frame count.		191	INT				
136	ABS			192	RTN				
137	EEX			193	*LBL2				Scoring subroutine
138	4			194	RCLB				S1
139	CHS			195	RCLC				S2
140	+			196	+				
141	ST0 i			197	RCLD				PS
142	*LBL6	Ready score and frame		198	+				
143	RCLD	display, PS.		199	RCL A				CS
144	EEX			200	+				
145	2			201	RCL E				SSS.FS
146	÷			202	ABS				
147	RCLC	S2		203	+				
148	+			204	EEX				
149	EEX			205	2				
150	2			206	CHS				.01
151	÷			207	+				Increment FS
152	RCLB	S1		208	ST0E				Update E
153	+			209	FRC				Compare FS to .1
154	EEX			210	.				
155	2			211	1				
156	÷			212	X>Y?				If less than .1
157	RCL A	CS		213	RTN				Continue
158	+	CS.S1S2PS		214	RCL E				Otherwise
159	P*S			215	ABS				
160	ST0 i	Update sec. reg.		216	ST0 i				Update print register.
161	P*S			217	PRTX				Print and stop.
162	RCL i			218	R/S				
163	ABS	Remove sign.		219	*LBLD				
164	EEX			220	1				Recall score display.
165	2			221	-				
166	X			222	ST0 i				
167	FRC	.NF		223	RCL i				
168	EEX			224	RTN				

LABELS					FLAGS	SET STATUS		
A ID.P entry	B Spare	C Strike	D Recall score	E	0	FLAGS		DISP
a Start	b Error	c	d	e	1 2 nd ball	ON OFF		
0 Used	1 Shift sub.	2 Score sub.	3 Spare	4 2 nd	2	0 ☐ ☒	DEG ☒	FIX ☒
5 Add frame	6 Display	7 Add count	8 Error	9 Initialize	3	1 ☐ ☒	GRAD ☐	SCI ☐
						2 ☐ ☒	RAD ☐	ENG ☐
						3 ☐ ☒		n 4

Biorhythms

[illegible]

113 RCL9		169 *LBLD		DAYS UNTIL
114 RCL1		170 4		NEXT MIN
115 ÷		171 ENT1		
116 FRC		172 3		
117 FI?		173 ÷		
118 GT08		174 ST07		
119 2		175 1		
120 x		175 RCLC		
121 Pi		177 GT08		
122 x		178 *LBL5		
123 SIN		179 2		
124 RND		180 ST07		
125 ST01		181 RCLC		
126 *LBL8		182 *LBL0		
127 RCL1		183 ST09		
128 x		184 X2Y		
129 GSB2		185 ST00		
130 ST01		186 SF1		
131 X21		187 GSB6		
132 3		188 CF1		
133 -		189 3		
134 X21		190 ST01		
135 DSZ1		191 *LBL3		
136 GT06		192 GSB2		
137 RTN		193 RCL1		
138 *LBL1		194 X21		
139 GSB9		195 3		
140 *LBL8		196 -		
141 DSZ1		197 X21		
142 GT06		198 RCL1		
143 SPC		199 RCL7		
144 RTN		200 ÷		
145 *LBL9	PRINT/PAUSE	201 -		
146 F0?		202 CHS		
147 PRTX		203 X2?		
148 F0?		204 GT03		
149 RTN		205 *LBL4		
150 R/S		206 PRTX		
151 RTN		207 *LBL3		
152 *LBL5	PRINT TOGLE	208 RCL1		
153 DSP0		209 RCL0		
154 F0?		210 ÷		
155 GT00		211 +		
156 1		212 RCL1		
157 SF0		213 X2Y		
158 PRTX		214 X2Y?		
159 *LBL0		215 GT05		
160 0		216 GT04		
161 CF0		217 *LBL5		
162 RTN		218 SPC		
163 *LBL4	DAYS UNTIL NEXT MAX	219 DSZ1		
164 4		220 GT03		
165 ST07		221 RTN		
166 1				
167 RCLC				
168 GT00				

LABELS					FLAGS	SET STATUS		
A Birth date	B Bio date	C +1	D MIN	E Critical	0	FLAGS		
a	b Print cycle	c N(M, D, Y)	d MAX	e Print?	1	TRIG		
0 Used	1 Used	2	3	4	2	DISP		
5	6	7	8	9 Print/Pause	3			

ON OFF					
0	☒	☐	DEG	☐	FIX ☒
1	☐	☒	GRAD	☐	SCI ☐
2	☐	☒	RAD	☒	ENG ☐
3	☐	☒	n 2		

Timer

001	*LSLA	INITIALIZATION	057	*LBL6	CALIBRATE 5				
002	CLRG		058	R↓	SECOND TIMER				
003	P+S		059	GSB4					
004	CLRG	-----	060	R↑					
005	5	C _b (MINIMUM	061	HMS+					
006	1	INTERVAL	062	XZY					
007	3	CALIBRATION	063	HMS+					
008	4	CONSTANT)	064	-					
009	1/X	-----	065	LSTX					
010	STOE		066	RCLC	K _a				
011	5	C _a (5 SECOND	067	x					
012	.	PRIMARY)	068	÷	C _a				
013	7		069	RCLB					
014	8		070	+					
015	3		071	PRTX					
016	8	-----	072	RTN					
017	STOB		073	*LBLC	COUNT-UP TIMER				
018	.	K _a (5 SECOND	074	GSBA					
019	1	SECONDARY)	075	R/S					
020	8		076	*LBL7	FAST LOOP				
021	7		077	ISZI					
022	5	-----	078	STO7					
023	STOC	C _c (COUNT-UP)	079	*LBLc	CONVERT TO TIME				
024	6		080	RCLI					
025	8		081	RCLD					
026	6		082	x					
027	8		083	+HMS					
028	1/X	-----	084	RTN					
029	STOD		085	P+S	SPLITS				
030	6	C _d (COUNT-DOWN)	086	RCL1					
031	7		087	STOI					
032	9		088	RCL2					
033	8		089	STOI					
034	1/X	-----	090	RCL3					
035	STO5		091	STO2					
036	8		092	RCL4					
037	RTN		093	STO3					
038	*LBL5	5 SECOND TIMER	094	RCL5					
039	PSE	LOOP	095	STO4					
040	HMS+		096	RCL6					
041	7		097	STO5					
042	2		098	RCL7					
043	8		099	STO6					
044	1/X	0.00:05 HRS	100	RCL8					
045	+		101	STO7					
046	STOA	TIME	102	RCL9					
047	RCLB	C _a	103	STO8					
048	RCLI		104	RCL8					
049	+		105	STO9					
050	STOI	-----	106	P+S					
051	*LBL1	FAST LOOP	107	RCL1					
052	DSZI		108	P+S					
053	STOI	-----	109	STO8					
054	RCLA	TIME	110	P+S					
055	+HMS		111	RCL2					
056	GTOS		112	STOI					
REGISTERS									
0 SPLIT	1 SPLIT	2 SPLIT	3 SPLIT	4 t _s SPLIT	5 C _d SPLIT	6 SPLIT	7 SPLIT	8 SPLIT	9 SPLIT
S0 SPLIT	S1 SPLIT	S2 SPLIT	S3 SPLIT	S4 SPLIT	S5 SPLIT	S6 SPLIT	S7	S8	S9
A		B		C		D		E	

113	RCL3		168	*LBL6	FAST LOOP
114	ST02		170	PSE	
115	PCL4		171	RCL5	
116	ST03		172	HMS+	
117	RCL5		173	GT06	
118	ST04		174	*LBL6	CALIBRATE MIN. INTERVAL TIMER
119	RCL6		175	R4	
120	ST05		176	GSB4	
121	GT06		177	X=Y	
122	*LBL6		178	R4	
123	RCL5		179	RCL5	
124	ST06		180	GT02	
125	RCL4		181	PRTX	
126	ST05		182	RTN	
127	RCL3		183	*LBL2	CALIBRATE SUBROUTINE
128	ST04		184	X=Y	
129	RCL2		185	HMS+	
130	ST03		186	X	
131	RCL1		187	X=Y	
132	ST02		188	HMS+	
133	P=S		189	+	
134	RCL0		190	1/X	
135	P=S		191	PRTX	
136	ST01		192	RTN	
137	P=S		193	*LBL4	CALIBRATE COUNT-UP TIMER
138	RCL9		194	GSB4	
139	ST00		195	GSB6	
140	RCL8		196	R/S	
141	ST09		197	X=Y	
142	RCL7		198	RCL0	
143	ST08		199	GT02	
144	PCL6		200	*LBL0	COUNT-DOWN TIMER
145	ST07		201	GSB4	
146	RCL5		202	R/S	
147	ST06		203	ST04	START TIME t_s
148	RCL4		204	HMS+	
149	ST05		205	RCL5	
150	RCL3		206	=	
151	ST04		207	ST01	
152	RCL2		208	RCL4	
153	ST03		209	R/S	
154	RCL1		210	*LBL3	FAST LOOP
155	ST02		211	DSZ1	
156	RCL1		212	GT02	
157	ST01		213	0	
158	4		214	RTN	
159	4	C ₀ SPLIT CONSTANT	215	*LBL6	CALIBRATE COUNT- DOWN TIMER
160	4		216	GSB4	START TIME t_s
161	2		217	RCL4	
162	+		218	X=1	
163	ST01		219	RCL5	C _d
164	P=S		220	GT02	
165	GT07		221	*LBL4	CALCULATE SWEEP SECOND HAND TIME INTERVAL t_c
166	*LBL6	MINIMUM INTERVAL VISIBLE TIMER	222	CHS	
167	GSB4		223	HMS+	
168	R/S		224	RTN	

LABELS					FLAGS	SET STATUS		
^A 5 SEC	^B MIN INT	^C COUNT-UP	^D COUNT-DN	^E SPLITS	0	FLAGS	TRIG	DISP
^A 5 SEC CAL	^B MIN CAL	^C CU to TIME	^D CAL C-U	^E CAL C-D	1	ON OFF		
0	1 5 SEC LP	2 CAL SUB	3 C-U LOOP	4 T _c	2	0 ☐ ☒	DEG ☒	FIX ☒
5 5 SEC LP	6 MIN LOOP	7 C-U LOOP	8	9	3	1 ☐ ☒	GRAD ☐	SCI ☐
						2 ☐ ☒	RAD ☐	ENG ☐
						3 ☐ ☒		n 4

Appendix A
MAGNETIC CARD
SYMBOLS AND CONVENTIONS

MAGNETIC CARD SYMBOLS AND CONVENTIONS	
SYMBOL OR CONVENTION	INDICATED MEANING
White mnemonic: x A	White mnemonics are associated with the user definable key they are above when the card is inserted in the calculator's window slot. In this case the value of x could be input by keying it in and pressing A.
Gold mnemonic: y x f E	Gold mnemonics are similar to white mnemonics except that the gold f key must be pressed before the user definable key. In this case y could be input by pressing f E.
x y A	is the symbol for ENTER. In this case ENTER is used to separate the input variables x and y. To input both x and y you would key in x, press ENTER, key in y and press A.
X A	The box around the variable x indicates input by pressing STO A.
(x) A	Parentheses indicate an option. In this case, x is not a required input but could be input in special cases.
→ x A	→ is the symbol for calculate. This indicates that you may calculate x by pressing key A.
→ x, y, z A	This indicates that x, y, and z are calculated by pressing A once. The values would be printed in x, y, z order.
→ x; y; z A	The semi-colons indicate that after x has been calculated using A, y and z may be calculated by pressing R/S.
→ "x," y A	The quote marks indicate that the x value will be "paused" or held in the display for one second. The pause will be followed by the display of y.
↔ x A	The two-way arrow ↔ indicates that x may be either output or input when the associated user definable key is pressed. If numeric keys have been pressed between user-definable keys, x is stored. If numeric keys have not been pressed, the program will calculate x.

SYMBOLS AND CONVENTIONS (Continued)

SYMBOL OR CONVENTION	INDICATED MEANING
P? A	The question mark indicates that this is a mode setting, while the mnemonic indicates the type of mode being set. In this case a print mode is controlled. Mode settings typically have a 1.00 or 0.00 indicator displayed after they are executed. If 1.00 is displayed, the mode is on. If 0.00 is displayed, it is off.
START A	The word START is an example of a command. The start function should be performed to begin or start a program. It is included when initialization is necessary.
DEL A	This special command indicates that the last value or set of values input may be deleted by pressing A.



Sales and service from 172 offices in 65 countries.
1000 N.E. Circle Blvd., Corvallis, OR 97330

00097-90134

● B C D E