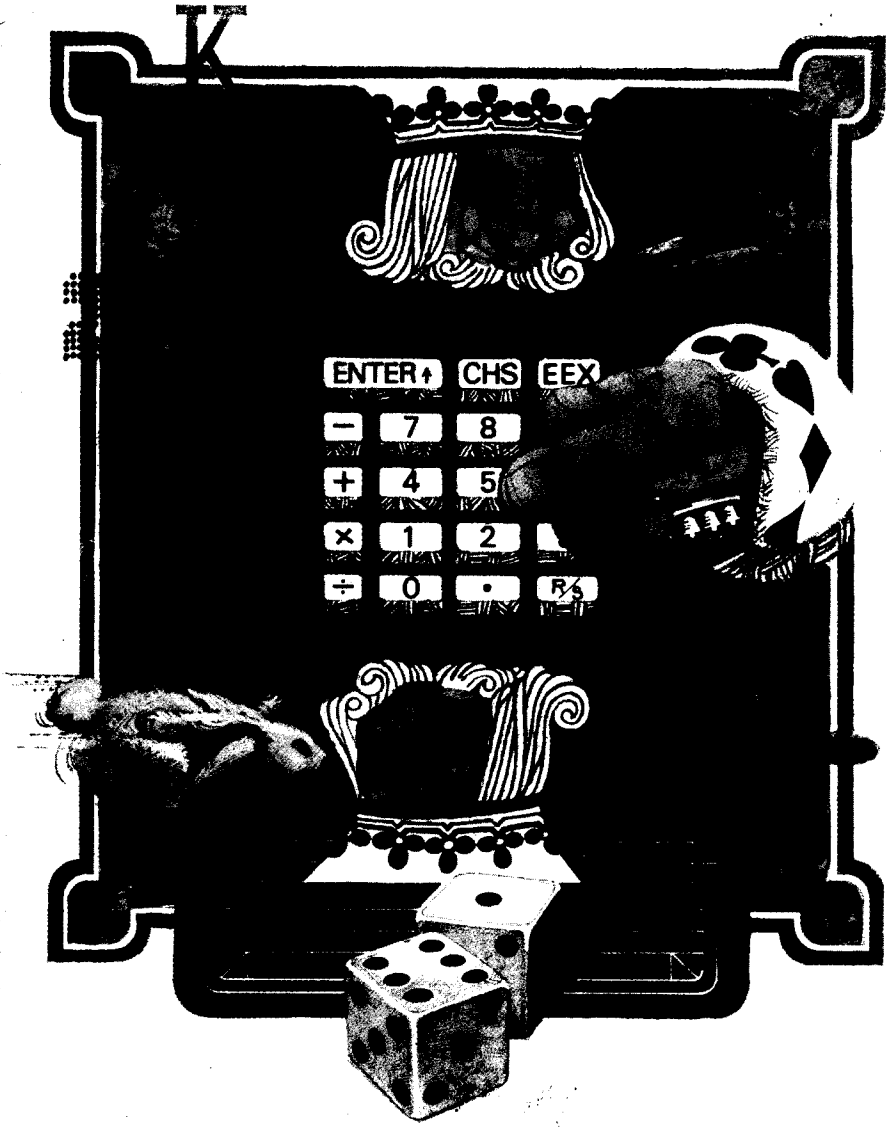
HEWLETT-PACKARD

HP-67/HP-97

Games Pac I



PRINTED IN U.S.A.

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
___01 Chemical
___02 Civil/Structural
___03 Electrical/Electronic
___04 Industrial
___05 Mechanical
___06 Surveying
___10 Other (Specify)_____

Science
___31 Biology
___32 Chemistry
___33 Earth Sciences
___34 Mathematics
___35 Medical Sciences
___36 Physics
___37 Statistics
___39 Other (Specify) _____

Business
___51 Accounting
___52 Banking
___53 Insurance
___54 Investment Analysis
___55 Real Estate
___56 Securities
___57 Sales
___58 Marketing
___59 Other (Specify)_____

Other
___71 Architecture
___72 Aviation
___73 Computer Science
___74 Education
___75 Navigation
___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

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;

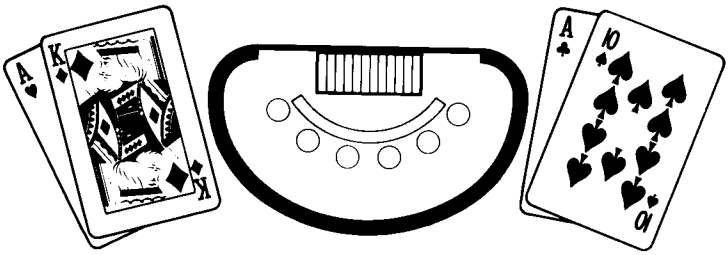
then the next register in the stack will be displayed, and so on. After display of all four registers, the program will halt execution if a key was pressed at any time during the display of the stack contents. In both cases execution of the halted program may be re-initiated by pressing [R/S].

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.		f A	
3	Stop shuffle and cut deck.		R/S	Ignore output
4	Input your bet (whole *even*			
	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?		f 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: **Outputs:**

f A → Display will not stabilize until R/S is pressed.

Wait 20 or 30 seconds.

R/S → Ignore output.

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

100 A → 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.

B → 4. *** Your 3rd card
100.21 Bet. Score

C → 9. *** Dealer's 1st card
7. *** Dealer's 2nd card
12. *** Dealer's 3rd card

	100.21	Your win. score Display blinks to indicate end of game. Positive number means you win.
	100.00	Your account
100 A →	6. ***	Dealer's 1st card
	0. ***	Dealer's 2nd card
	1. ***	Your 1st card
	7. ***	Your 2nd card
	100.08	Your bet. score
D →	100.18	Your bet. score

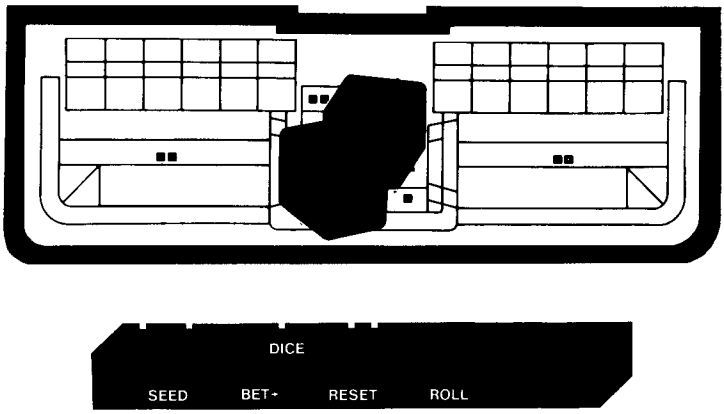
Your Ace is now counted as 11.
You decide to stand.

C →	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.			

Example 1:

Load sides 1 and 2.

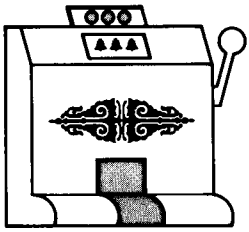
Keystrokes:	**Outputs:**	
Set seed.		
9 A →	0.00	
Place bet.		
10 B →	5.1 06	■ ◆ = 6
	6.3 09	◆ ■ = 9
	3.6 09	
	1.4 05	
	6.4 10	
	6.6 12	
	2.6 08	
	2.4 06	
	10.00	WIN!
Place bet.		
10 B →	3.2 05	
	3.2 05	
	20.00	WIN!
Place bet.		
10 B →	6.6 12	
	10.00	LOSE!
Place bet.		
10 B →	5.6 11	
	20.00	WIN!
Place bet.		
10 B →	1.4 05	
	3.3 06	
	4.5 09	
	6.6 12	
	3.1 04	
	5.2 07	
	10.00	LOSE!
Another shooter		
C →	0.00	

Example 2:

Keystrokes:	**Outputs:**
Set seed.	
1 A →	0.00
Roll dice.	
D →	5.3 08
D →	1.2 03
D →	1.1 02
D →	3.1 04
D →	2.6 08

etc.

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 $0.D_1D_2D_3$. 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.

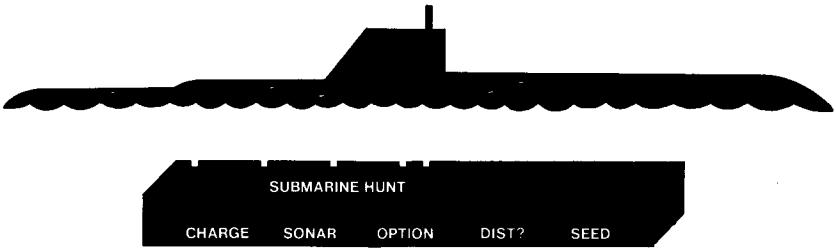
STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1.			
2	Enter seed (a many digit			
	number between 0 and 1).	Seed	E	0.
3	Play.		A	Combination
	Winning combinations:			
	0.1XY $ 2.00			
	0.11X $ 5.00			
	0.ZZZ $10.00			
	(where Z is 2 or more)			
	0.ZZ0 $10.00			
	(where Z is 2 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:		**Outputs:**	
.963258741 E	→	0.	
A	→	0.450	
A	→	0.001	
A	→	0.000	JACKPOT!
B	→	97.00	
A	→	0.173	A $2 WINNER
B	→	98.00	
A	→	0.991	
A	→	0.026	
A	→	0.902	
A	→	0.999	A $10 WINNER
B	→	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 [f][E].

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:		D	2. or 1.
	"2" means sensitivity			
	distance is 2 squares.			
	"1" means sensitivity			
	distance is 1 square.			
6	**SONAR**	Row	ENTER↑	
	"0" means no echo.	Col.	B	0. or 1.
	"1" means echo received.			
	or			
	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:		f E	XX.YY
	XX = number of depth			
	charges fired.			
	YY = number of sonar			
	readings.			
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	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 ⟶ 0.111
0.11111
0.1111111
0.111111111
0.111111111 A hit!

Example 2:

Keystrokes:	**Outputs:**
60 E ⟶	0.
C ⟶	1.

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

First move:

7 ENTER↑ 4 B ⟶ 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.

	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9											9				x	x	x					9
8				x	x	x					8			x	x	x	x	x				8
7				x	x	x					7			x	x	x	x	x				7
6				x	x	x					6			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 1st move

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	x	⊗			7				x	x	x	x	x			7
6		⊗	⊗	⊗	x	x	x	⊗			6				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

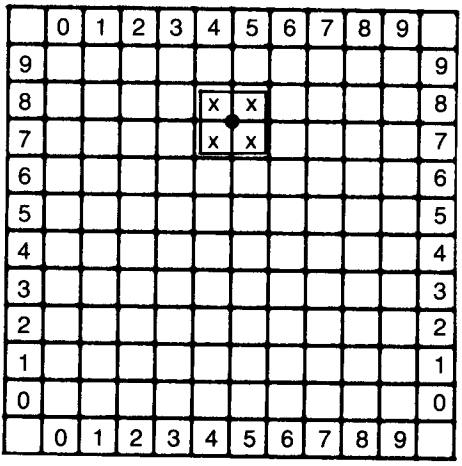
Fourth move: You try a depth charge.

7.5 ENTER↑ 4.5 A ⟶

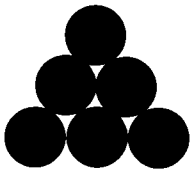
0.111	
0.11111	
0.1111111	
0.111111111	
0.111111111	A hit!

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.

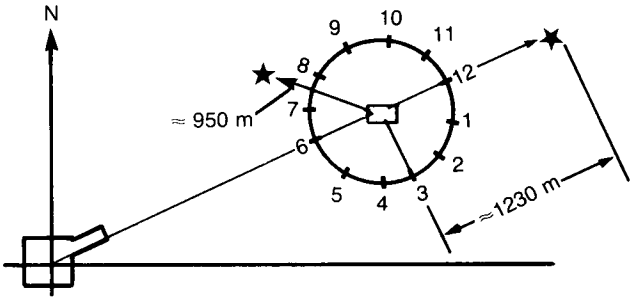
(1) Initial display for each new battle is one of the 8 main compass directions: 0 (North), 45, 90 (East), 135, 180 (South), 225, 270 (West), or 315. The target lies in that *general* direction from the hidden gun and is 5 to 10 kilometers away.

(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 ([f][A]) sets the following parameters:

a) Maximum gun range = 10,000 meters. (Thus, the formula for any shell's range = 10,000 × Sin (2 × elevation).)

b) Target KILL range = 100 meters.

c) Target SPEED = 500 meters/minute (along each axis).

d) Spotter rating = 3.0 (1 = poor, 4 = perfect).

e) 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 ±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 ——→ 10000.00

(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 ***

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

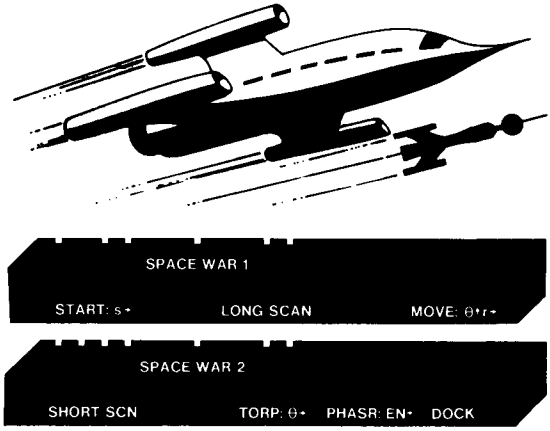
Example 3:

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 B	65 A
52 A	53 KH	54	55
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:

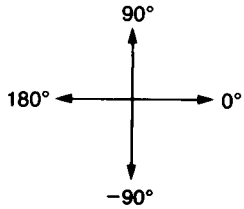
63.00400401
53.10400400
43.00400400

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 $\theta = 180°$ 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			
	Space War 1.			
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.XXXXXXXXX
	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:	**Outputs:**	
.63154897 A →	50.53	(KH position)
C →	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 → 14.62 (New position)

C → 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:

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

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:

	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							7	6
5	●	●									5											5
4	●	●									4											4
3											3											3
2				●	●	●		KH			2											2
1				●	●			B			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**

Move to S68.

Load side 1 and side 2 of SW1.

0 ENTER↑ .6 E ⟶ 15.68

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	7	6
5	●	●									5											5
4	●	●									4											4
3											3											3
2				●	●	●		KH			2											2
1				●	●			B			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**

Dock.

Load side 1 and side 2 of SW2.

E → 1000.3 (Energy, torpedos replenished)

Load side 1 and side 2 of SW1.

90 ENTER↑ 5 E → 65.68 (New position)

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	●	●				KH					6									3		6
5	●	●									5											5
4	●	●									4											4
3											3											3
2				●	●	●					2											2
1				●	●	B					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**

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:

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

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.

Super Bagels

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 ≤ D ≤ 8).	D	B	D.M*
4	To change the maximum digit			
	found in the number to be			
	guessed, key in the number			
	(0 ≤ M ≤ 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 ≤ 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:	**Outputs:**
1.23456987 EEX CHS 12 A →	4.5
3214 E →	11.3214 *** (1 right, 1 in right place)
3015 E →	2.3015 *** (2 right)
5234 E →	21.5234 *** (1 right, 2 in right place)
5203 E →	30.5203 *** (3 in right place)
5223 E →	30.5223 ***
5253 E →	40.5253 *** (Correct)!
	6. *** (6 guesses)

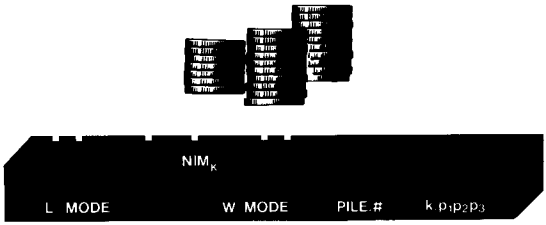
Example 2:

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:	Outputs:
1.23456987 A →	4.5
3 B →	3.5
9 C →	3.9
512 E →	1.512 ***
702 E →	1.702 ***
217 E →	10.217 ***
236 E →	20.236 ***
235 E →	20.235 ***
239 E →	20.239 ***
238 E →	30.238 ***
	7. *** (Number of guesses)

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

Nim_k



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

$$c_1, c_2, \ldots, c_n \text{ counters,}$$

represent these numbers

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

in the binary scale of notation with

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

where $c_i = c_{i0} + c_{i1}\, 2^1 + c_{i2}\, 2^2 + \ldots + c_{ij}\, 2^j$

$$(i = 1, 2, \ldots, n).$$

The combination is safe when

$$\sum_{i=1}^{n} c_{ij} = 0 \,(\text{mod}\, k + 1) \qquad (j = 0, 1, 2, \ldots).$$

That is, for every place j the sum of the n digits c_{ij} (i = 1, ..., 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

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

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 gen-			
	erates calculator's move.			$k.p_1p_2p_3...$
4	Input choice of pile and number			
	of objects to be removed.	pile.#	D	pile.#
	m = the number of piles yet			$m.p_1p_2p_3...$
	available to choose objects			
	from this turn.			

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: **Outputs:**

7.478379895 E ⟶ 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.

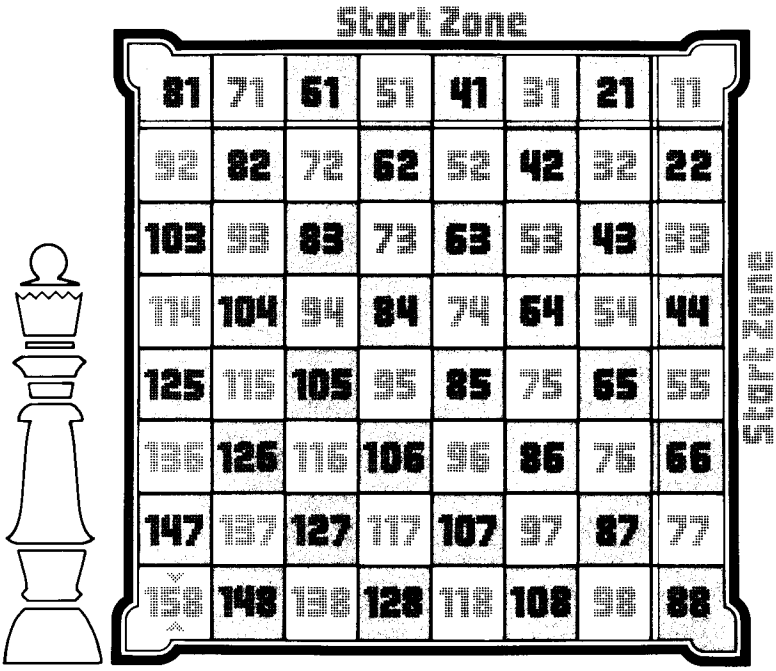
7.455155555 ***

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

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 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:

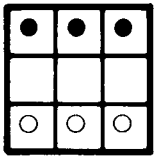


Figure 1. Starting position of pawns

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

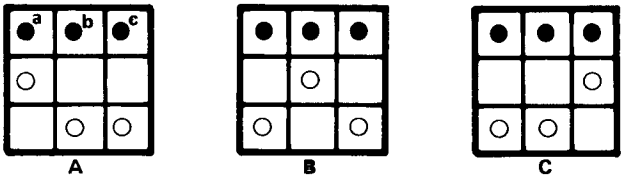


Figure 2. Opening moves

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.

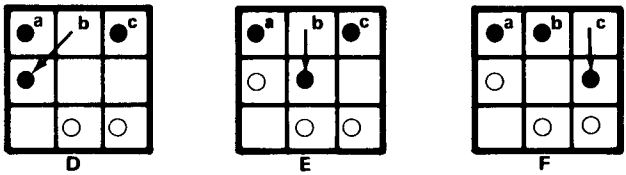


Figure 3. Black's responses to white's A move

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.

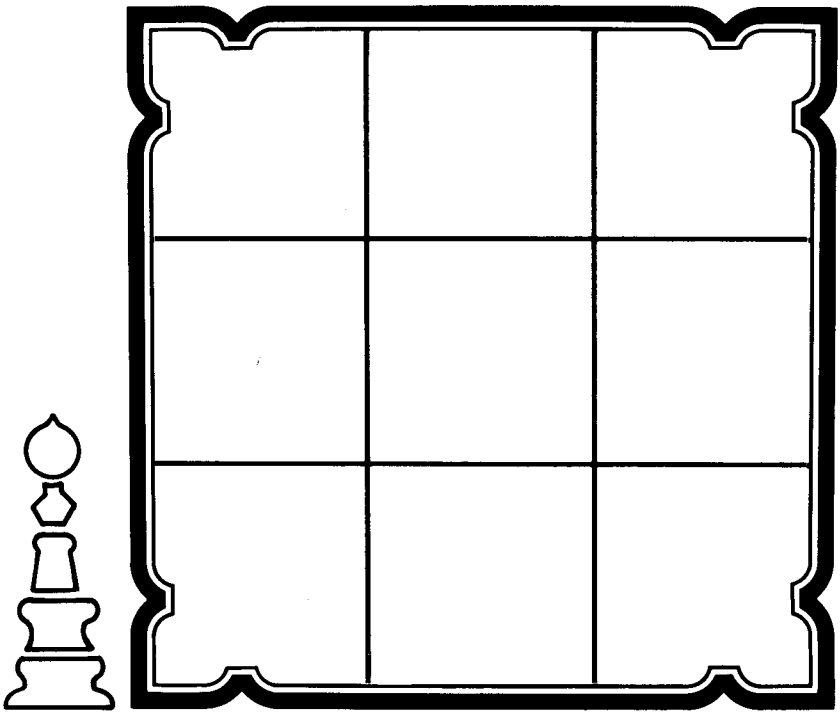
*Similarly, if you punish the machine for making good moves, it will eventually lose consistently—the calculator is your slave.

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	

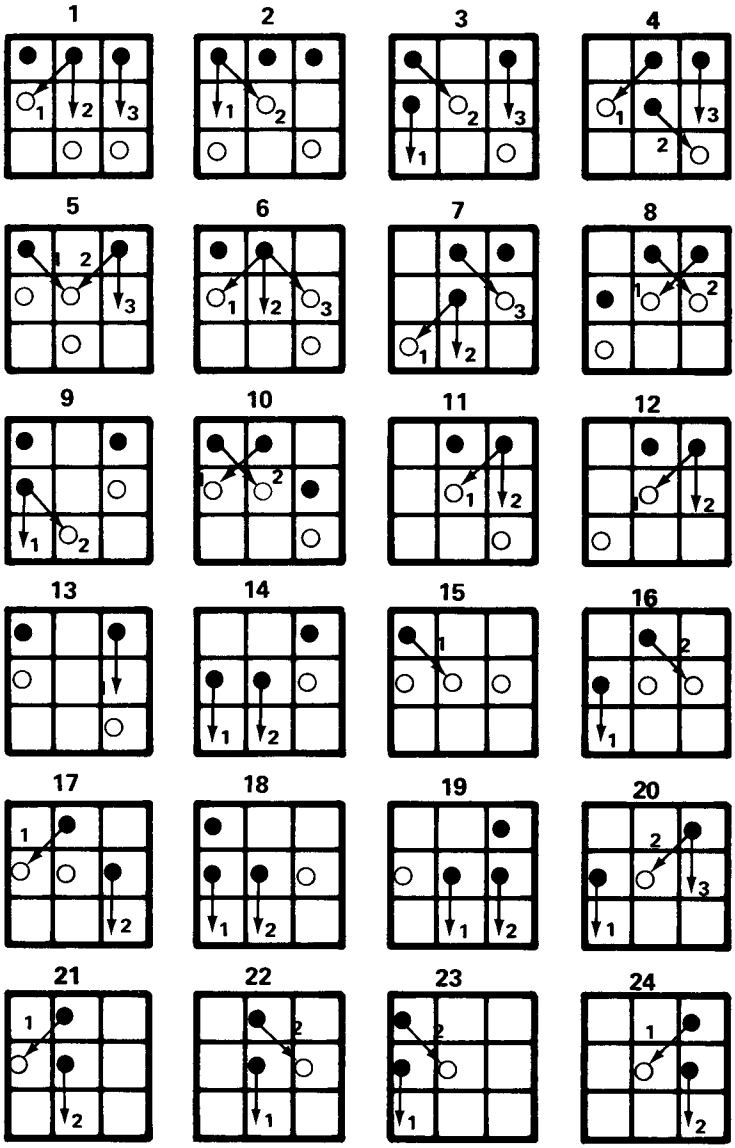
STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
6	To play another game:			
	a) If the same player plays			
	first, go to step 3.			
	b) If the other player plays			
	first, go to step 2.			



Playing Board for Hexapawn

Board Positions and Responses

You move *first:* You are white (○)



Board Positions and Responses

You move *second:* You are white (○)

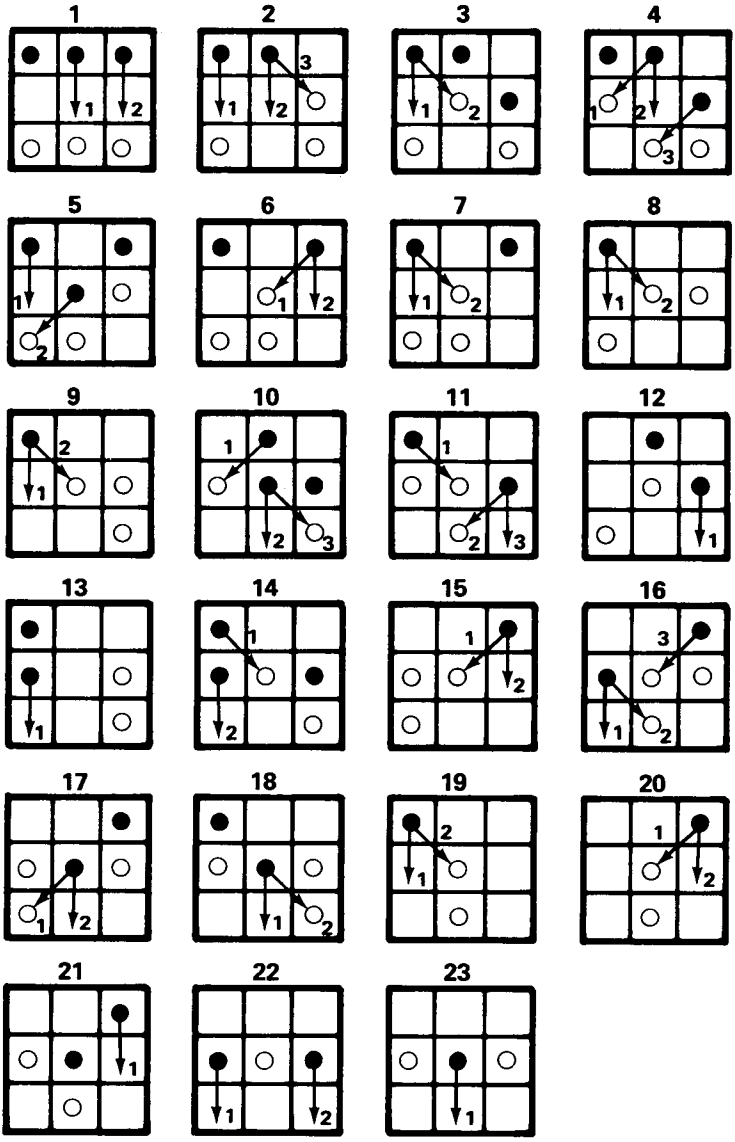


Figure 5

Example 1:

Load sides 1 and 2.

Keystrokes: **Outputs:**

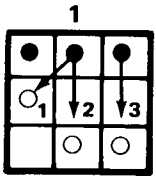
You move first.

f C (Wait a few seconds.)

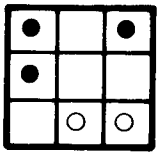
R/S → Ignore display.

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 → 1.00



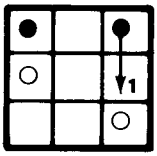
Board position 1



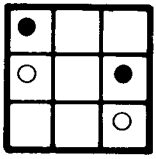
Board after calculator's response

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

13 A → 1.00



Board position 13



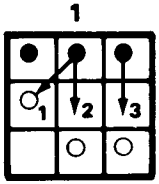
Board after calculator's response

You can't move, so the calculator wins.

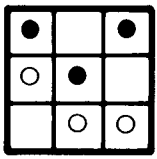
Example 2:

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

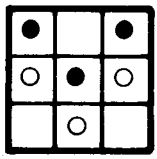
1 A ——————→ 2.00



Board position 1



Board after calculator's response



Your winning board position

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

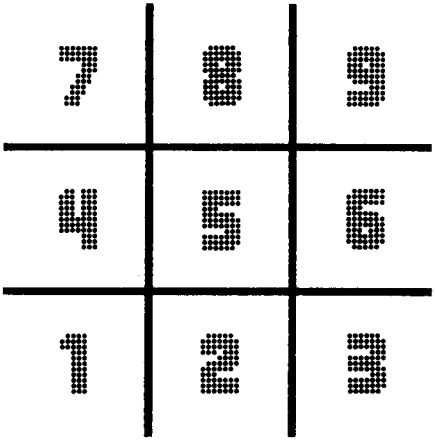
B ——————→ Ignore output.

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.

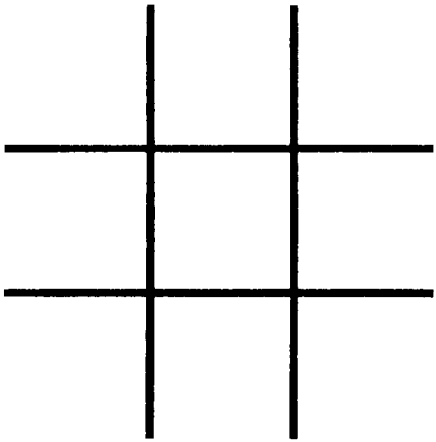
Tic·Tac·Toe

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.		f A	0.000000000
3	To start a game		A	Board*
4	Repeat step 5 until the end of			
	the game.			
5	Your turn. Enter position			
	number (see diagram above).	$1 \leq x \leq 9$	R/S	
	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 cor-			
	responds to this board			
	position:			
	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 4th move.			
	The 7 means the calculator has			
	just moved into position 7			
	(see diagram above).			

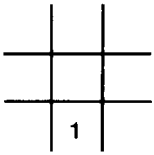
Example:

Load sides 1 and 2.

Keystrokes:	**Outputs:**
f A →	0.000000000

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

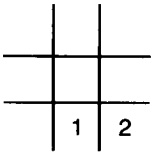
A →	0.000 ***
	0.000 ***
	0.010 ***
	0.010000000



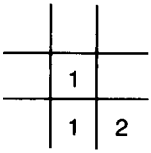
Board position 1A
Calculator's opening move

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

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

Flushed with success, you decide to play another game:

A	→	0.000 ***
		0.000 ***
		0.010 ***
		0.010000000
5 R/S	→	1.000 ***
		1.020 ***
		1.110 ***
		1.110020000

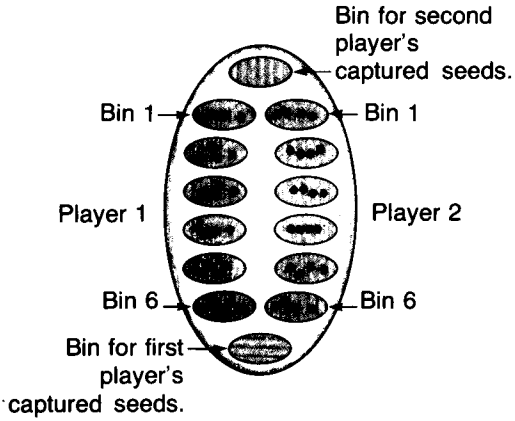
***Shown by PRINT on HP-97 and by PAUSE on HP-67.

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

Wari

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.



Wari Board at start of game

Bin 1	4.04	Bin 1
	4.04	
Player 1	4.04	Player 2
	4.04	
	4.04	
Bin 6	4.04	Bin 6

HP-97 Wari Printout 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 cal-			
	culator, go to step 9.			
	**See text and example.			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
5	Player 2: Select move.	Move	E	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:		h CF	
			0 f	
			E	Move,
				Board,
				Score
	For HP-97:		f CLF	
			0 f	
			E	Move,
				Board,
				Score
10	Select move (choose bin			
	number from 1 to 6).	Move	A	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.		f 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	
			0	
	For HP-97		f STF	
			0	
	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: **Outputs:**

[C] ⟶

4.04 ***		Starting board
4.04 ***		
4.04 ***		
4.04 ***		
4.04 ***		
4.04 ***		
0.00	Score	

Move 1:

You choose to move your counters out of bin 2:

2 [A] ⟶

4.04 ***		Board after your 1st move
0.04 ***		
5.04 ***		
5.04 ***		
5.04 ***		
5.04 ***		
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 ***		Board after Cal's 1st move
1.05 ***		
5.00 ***		
5.04 ***		
5.04 ***		
5.04 ***		
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 ***		Board after your 2nd move
1.05 ***		
5.01 ***		
5.05 ***		
0.05 ***		
6.05 ***		
4.00 ***	Cal's 2nd move	

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

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 ***
1.09 ***
6.05 ***
1.00 ***
2.00 ***
2.01 ***
} Board after your 5th move

6.00 *** Cal's 5th move

0.09 ***	Board after Cal's 5th 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 ***	Board
2.00 ***	
1.00 ***	
5.00 ***	
4.00 ***	
0.00 ***	
31.04	Score

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:

f E ⟶ 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

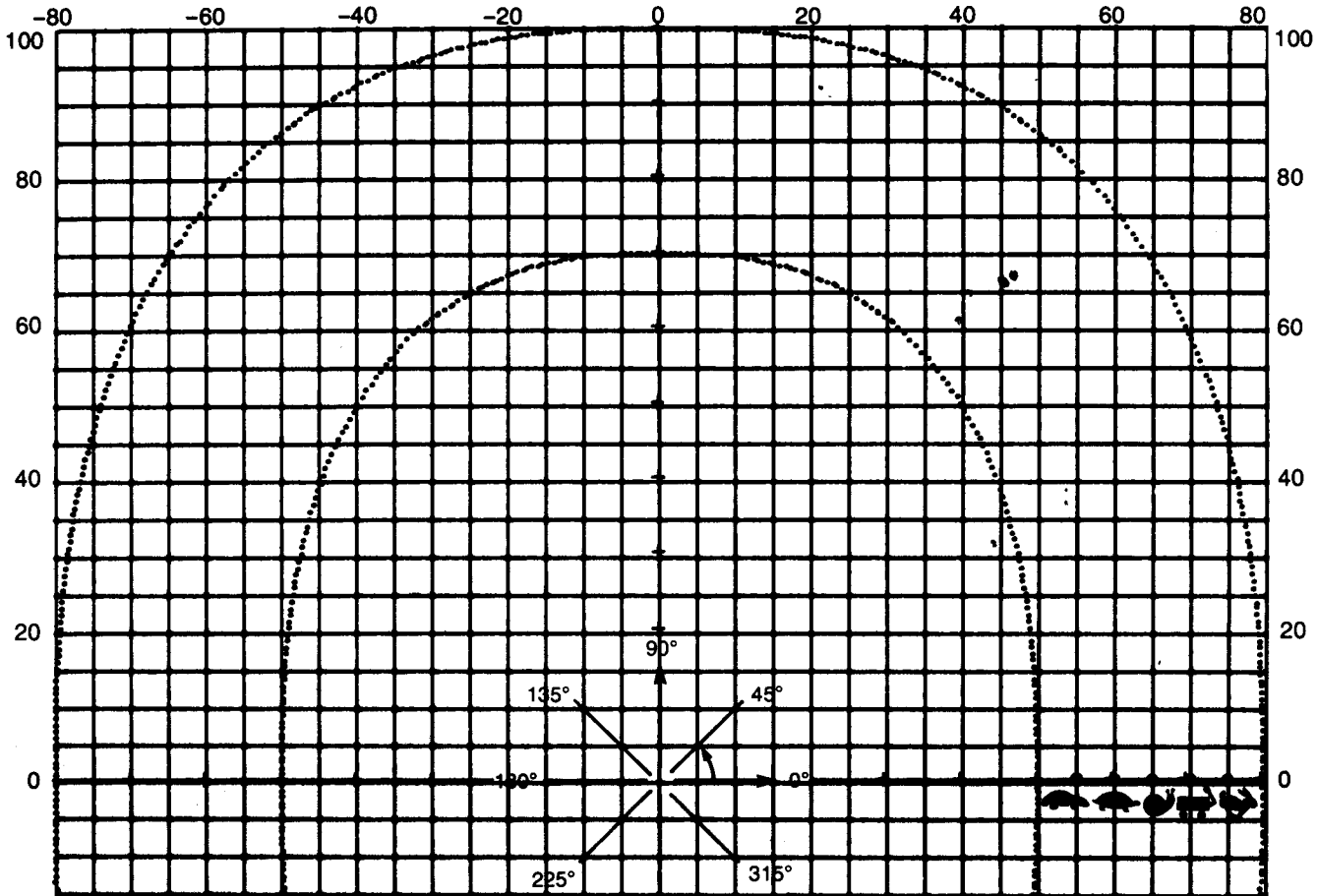
$$\left(\frac{x}{80}\right)^{5/2} + \left(\frac{y}{100}\right)^{5/2} = 1$$

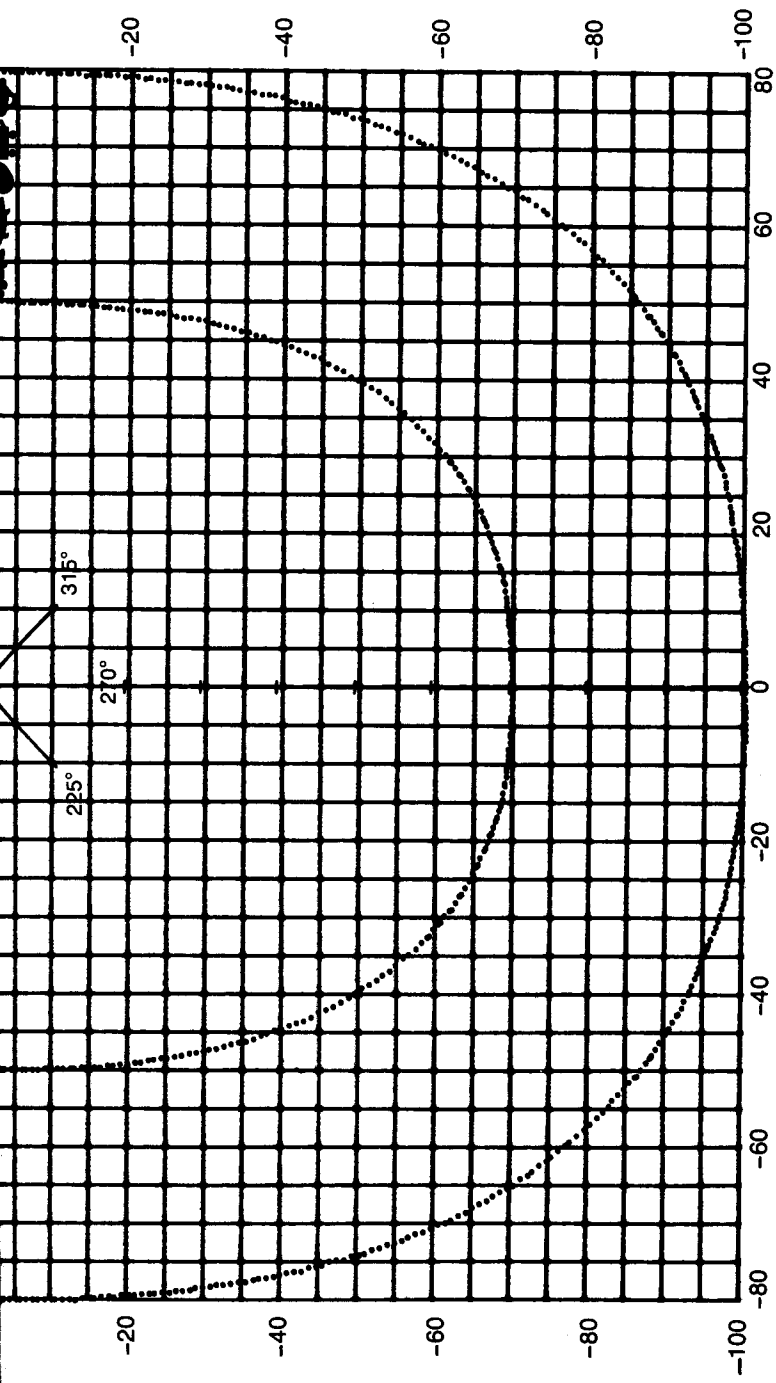
inner boundary

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

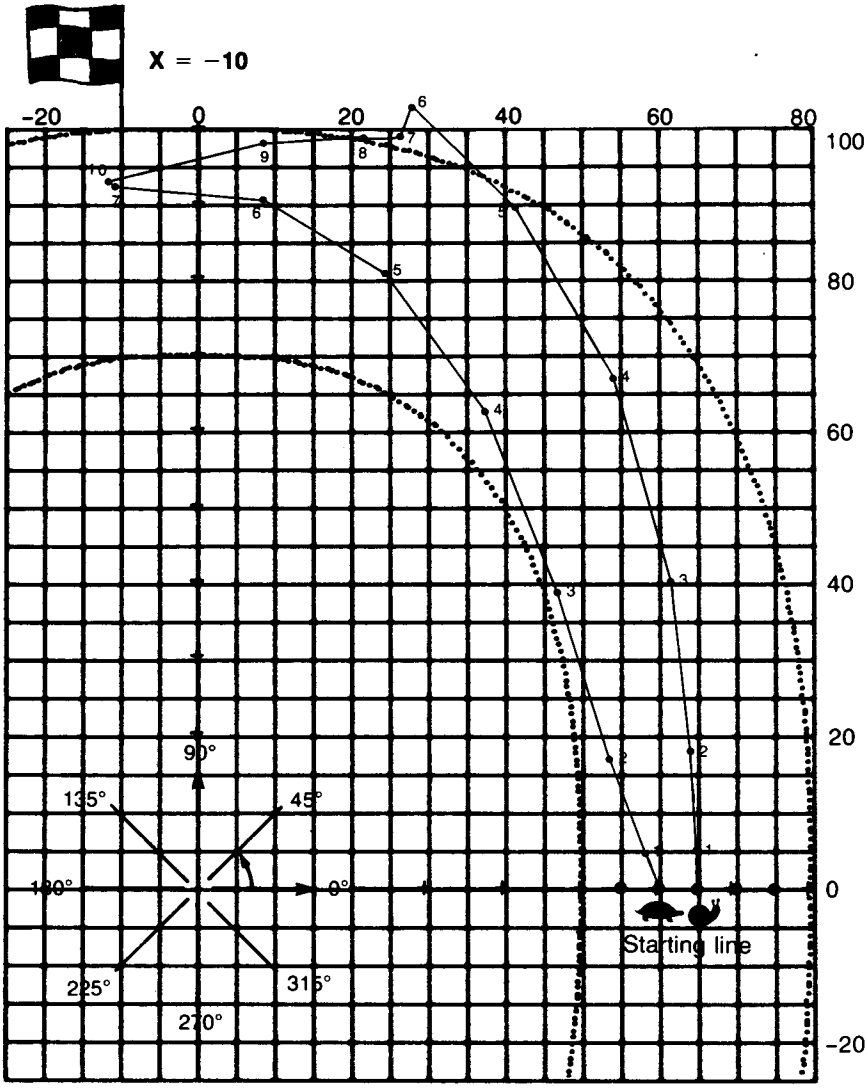
At the start, the five racers are located as shown on the racetrack at the points (55, 0), (60, 0), (65, 0), (70, 0), and (75, 0). Moves are made by placing direction in the y-register, acceleration in the x-register, and pressing A, B, C, D, or E as appropriate for the desired racer. The direction convention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


-80
-60
-40
-20
0
20
40
60
80
100
80
60
40
20
0
90°
135°
45°
180°
0°
225°
315°



Racetrack
Shows grid and starting positions



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.		f 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.	$1 \leq ID \leq 5$	f 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:	**Outputs:**
Initialize.	
f C →	-3.00 (Ignore this display.)
The racers start by accelerating in the directions 110 and 90 at the maximum acceleration.	
110 ENTER↑ 9 B →	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
90 ENTER↑ 9 C →	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 →	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
100 ENTER↑ 9 C →	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
100 ENTER↑ 9 C →	3.00 ***	Snail
	96.67 ***	Direction
	26.91 ***	Speed
	40.23 ***	y
	61.87 ***	x

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

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

```
TEASER

MOVE-          START          PRINT?
```

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.

```
1 0 1                          1 0 1
1 1 0                          1 0 1
0 0 1 ←——— a move here yields: 0 1 0
```

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

```
1 0 0                          1 0 1
1 0 1 ←——— a move here yields: 1 0 0
1 0 0                          1 0 1
```

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

```
1 0 0                          1 1 0
0 1 1 ——— a move here yields:   1 0 0
1 0 0                          1 1 0
```

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 ≤ moves ≤ 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.			

Example:

Load sides 1 and 2.

Keystrokes:	**Outputs:**
C →	0.000 ***
	0.010 ***
	0.000 ***
	9.000010000 00
5 A →	0.010 ***
This is the only possible move.	0.101 ***
	0.010 ***
	9.010101010 01
6 A →	0.011 ***
	0.100 ***
	0.011 ***
	9.011100011 02

The game continues. We pick up the action several moves later.

4 A →	0.100 ***
	0.011 ***
	0.100 ***
	9.100011100 09
6 A →	0.101 ***
	0.010 ***
	0.101 ***
	9.101010101 10
5 A →	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.

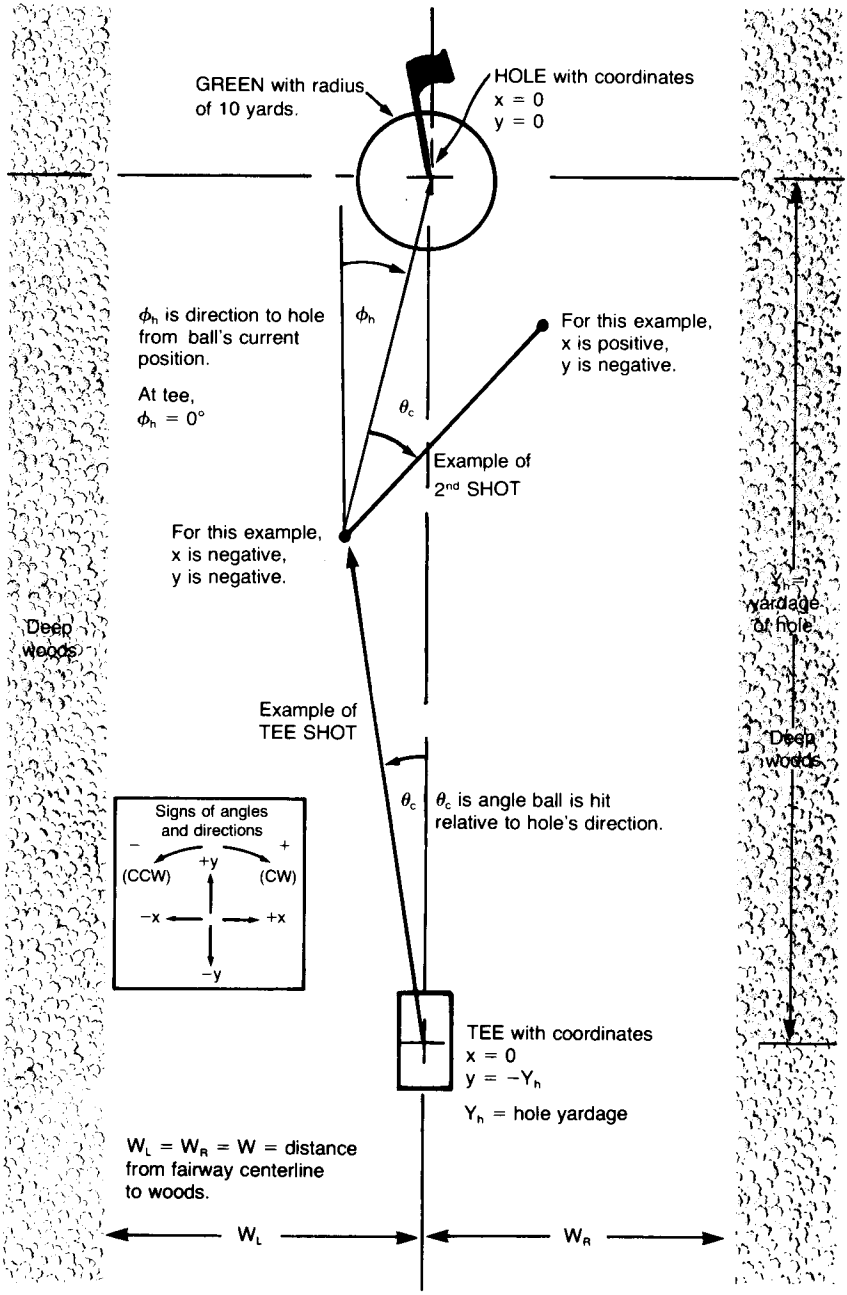
Golf

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 $\phi = 0°$, 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.		f CL REG	
			f P≷S	
			f CL REG	
3	Choose and store a six			
	digit random number seed			
	u_0, between 0 and 1 such as			
	.345762. To change the		STO A	
	character of each game, store			
	a different random number.			
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_h, Par, W*
7	Choose club and enter club			
	number:			
	Either:			
	Wood (Club #1, 2, 3 or 4),	C#	B	0
	or:			
	Iron (Club #1, 2, 3, 4, ..., 10			
	or 11),	C#	C	0
	or:			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	Putter (use only if hole dis-			
	tance (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	
			f P≷S	H_1
	Player 2:	H_2	STO 7	
			f P≷S	H_2
14	Design hole.		A	h, Y_h Par, W*
15	Player 1, make your mark.	1	SPACE	
			PRINT X	1†*
16	Player 1, choose club and			
	enter club number:			
	Either:			
	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.		f 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:			
	Either:			

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.		f P↔S	
25	Player 1, go to step 15.			
26	Player 2, allow player 1 to hit			
	his tee shot.		f 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:			
	Either:			
	If ball is not sunk in hole:			Y_c, θ_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.			
	Or:			
	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.

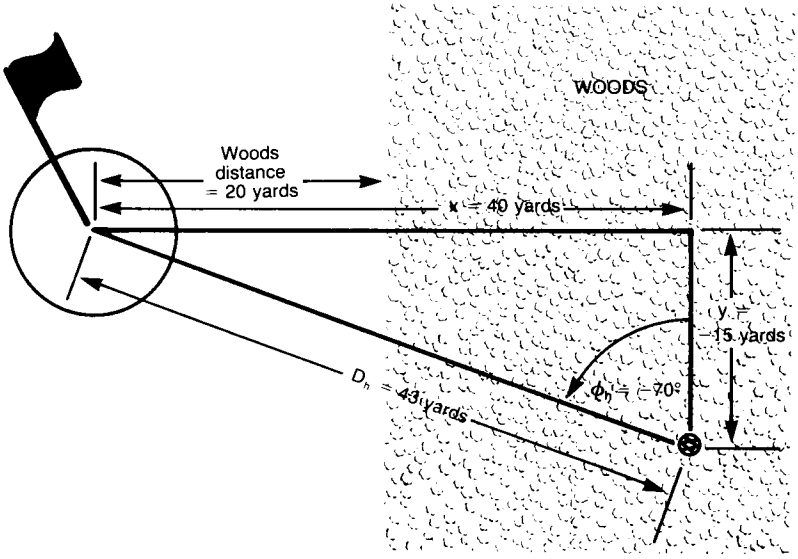
Keystrokes:	**Outputs:**	
f CL REG f P≈S f CL REG		
.637914 STO A →	1.	u_0 Seed

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

18 STO 7 →	18.	
A →	1. ***	h Hole no.
	241. ***	Y_h Hole yd.
	4. ***	Par
	20. ***	W Woods dist.
2 B →	0.	
1 R/S →	229. ***	Y_c Club yd.
	10. ***	θ_c Club angle
	−15. ***	y } Ball location
	40. ***	x }
	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 →	0.	
1 R/S →	211. ***	Y_c Club yd.
	−2. ***	θ_c Club angle
	−19. ***	y } Ball location
	−6. ***	x } Ball location
	19. ***	D_h Hole dist.
	18. ***	ϕ_h Hole angle
11 C →	0.	
.2 R/S →	4. ***	Y_c Club yd.
	−11. ***	θ_c Club angle
	−15. ***	y } Ball location
	−5. ***	x } Ball location
	16. ***	D_h Hole dist.
	20. ***	ϕ_h Hole angle

11 C →	0.	
.2 R/S →	20. ***	Y_c Club yd.
	1. ***	θ_c Club angle
	4. ***	y } Ball location
	2. ***	x } Ball location
	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 →	0.	
.6 R/S →	5. ***	Y_c Club yd.
	−3. ***	θ_c Club angle
	−1.613743000−02 ***	y } Ball location
	2.725378030−01 ***	x } Ball location

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.

	2.730151474−01 ***	D_h Hole dist.
	−87. ***	ϕ_h Hole angle
D →	0.	
.1 R/S →	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:	Outputs:		
f CL REG f P≷S f CL REG			
.385246 STO A →	3.852460000–01		u_0 Seed
0 STO 7 f P≷S →	0.		Janet's handicap
34 STO 7 f P≷S →	34.		D.C.'s handicap
A →	1.	***	h Hole no.
	571.	***	Y_h Hole yd.
	5.	***	Par
	24.	***	W Woods dist.
1 f SPACE PRINT X →	1.		Janet's I.D. no.

This step is optional.

1 B →	0.		
1 R/S →	246.	***	Y_c Club yd.
	2.	***	θ_c Club angle
	–325.	***	y } Ball location
	11.	***	x } Ball location
	325.	***	D_h Hole dist.
	–2.	***	ϕ_h Hole angle
f P≷S			
2 f SPACE PRINT X →	2.		D.C.'s I.D. no.
1 B →	0.		
1 R/S →	204.	***	Y_c Club yd.
	–8.	***	θ_c Club angle
	–369.	***	y } Ball location
	–29.	***	x } Ball location
	370.509734	***	D_h Hole dist.
	4.455967	***	ϕ_h Hole angle

f P≷S

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.		Janet
1 B →	0.		
1 R/S →	276.	***	Y_c Club yd.
	1.	***	θ_c Club angle
	–49.	***	y } Ball location
	9.	***	x } Ball location
	50.	***	D_h Hole dist.
	–10.	***	ϕ_h Hole angle

f [P≷S]

2 f [SPACE] [PRINT x] →	2.	D.C.
1 [B] →	0.	
1 [R/S] →	25. ***	Y_c Club yd.
	-65. ***	θ_c Club angle
	-357. ***	y } Ball location
	-51. ***	x } Ball location
	360.811319 ***	D_h Hole dist.
	8.046678 ***	ϕ_h Hole angle

f [P≷S]

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] →	1.	Janet
10 [C] →	0.	
.5 [R/S] →	48. ***	Y_c Club yd.
	1. ***	θ_c Club angle
	-1. ***	y } Ball location
	1. ***	x } Ball location
	2. ***	D_h Hole dist.
	-52. ***	ϕ_h Hole angle

f [P≷S]

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

2 f [SPACE] [PRINT x] →	2.	D.C.
1 [B] →	0.	
1 [R/S] →	30. ***	Y_c } Result of D.C.'s swing
	-53. ***	θ_c } Result of D.C.'s swing
	-337. ***	y } Result of D.C.'s swing
	-72. ***	x } Result of D.C.'s swing
	344.037744 ***	D_h } Hole location from ball
	12.002730 ***	ϕ_h } Hole location from ball

f [P≷S]

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

1 f SPACE PRINT x →	1.	Janet
D →	0.	
.3 R/S →	3. *** Y_c	Result of Janet's swing
	−7. *** θ_c	
	1. *** y	
	−1. *** x	
	1. *** D_h	Hole location from ball
	114. *** ϕ_h	
f P≷S		

Too bad, Janet missed a good birdie chance.

2 f SPACE PRINT x →	2.	D.C.
1 B →	0.	
1 R/S →	32. *** Y_c	
	100. *** θ_c	
	−348. *** y	
	−42. *** x	
	350.755487 *** D_h	
	6.898699 *** ϕ_h	
f P≷S		
1 f SPACE PRINT x →	1.	Janet
D →	0.	
.2 R/S →	0.000000000 ***	
	0.	S
f P≷S		

Janet made her par.

2 f SPACE PRINT x →	2.	D.C.
1 B →	0.	
1 R/S →	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 →	0.	
1 R/S →	98. ***	Y_c
	1. ***	θ_c
	-7. ***	y
	1. ***	x
	7. ***	D_h
	-10. ***	ϕ_h

D.C. is on the green.

D →	0.	
.8 R/S →	8. ***	Y_c
	-12. ***	θ_c
	2.430006000-03 ***	y
	-2. ***	x
	2. ***	D_h
	90. ***	ϕ_h
D →	0.	
.3 R/S →	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.

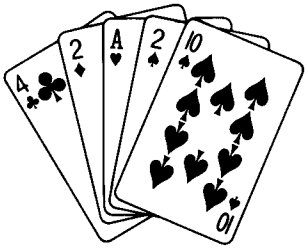
A →	2. ***	h
	399. ***	Y_h
	4. ***	Par
	22. ***	W

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.

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.		f A	0.
4	Optional: input seed			
	(0 < seed < 1).	seed	R/S	seed
5	Deal one card,		A	"suit. card"
	or deal a specified number			
	of cards.	n	C	"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.		f C	0. 00
9	Go to step 5 and deal cards.			
10	Select bingo dealer.		f E	0.
11	Optional: input seed			
	(0 < 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:	**Outputs:**
f A →	0.
5 C →	4. 04 *** (4C)
	3. 02 *** (2D)
	2. 01 *** (AH)
	1. 02 *** (2S)
	1. 10 *** (10S)
	0. 00
5 C →	3. 09 *** (9D)
	3. 12 *** (QD)
	4. 03 *** (3C)
	3. 04 *** (4D)
	3. 01 *** (AD)
	0. 00

f A →	0. 00
.896348 R/S →	9. -01
5 C →	3. 08 ***
	4. 10 ***
	1. 10 ***
	1. 01 ***
	1. 06 ***
	0. 00
5 C →	1. 03 ***
	2. 11 ***
	2. 02 ***
	1. 11 ***
	2. 13 ***
	0. 00

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:	**Outputs:**
f E →	0. 00
A →	62.
A →	41.
A →	20.
A →	3.
A →	14.
A →	52.
A →	57.
A →	64.
A →	45.
A →	74.
E →	62. ***
	41. ***
	20. ***
	3. ***
	14. ***
	52. ***

	57. ***
	64. ***
	45. ***
	74. ***
f C →	0.
10 C →	17. ***
	54. ***
	36. ***
	55. ***
	6. ***
	21. ***
	33. ***
	3. ***
	61. ***
	64. ***
	0.

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

Bowling Scorekeeper

BOWLING SCOREKEEPER

ID.P - SCR ID - ID - ID - SCORE

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.

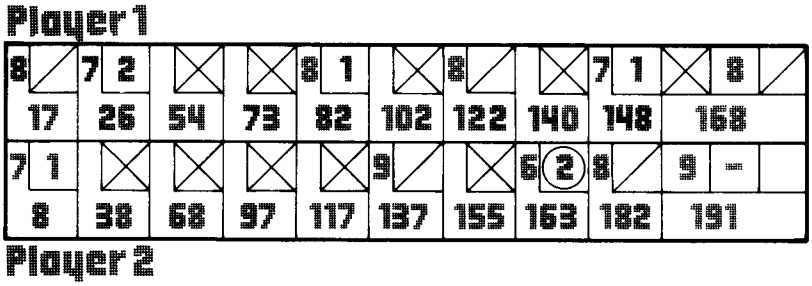
PS = Spare previous frame.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Initialize.		[f] [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.



Load sides 1 and 2.

Keystrokes: **Outputs:**

f A → 0.0000

1.8 A → -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

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

1 [C] ——————→ 148.091110

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

1.8 [A] ——————→ −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] ——————→ −182.091009 ***
2.0 [A] ——————→ 191.100000 ***

Player 2 wins easily.

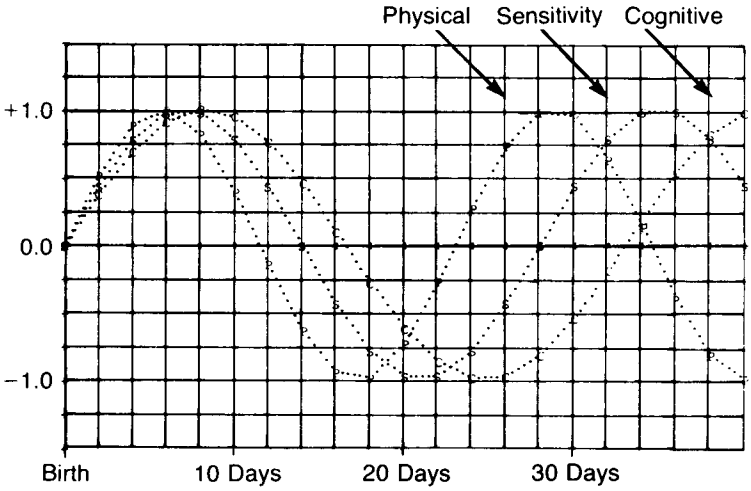
***Shown by PRINT on HP-97 and by PAUSE on HP-67.

Biorhythms

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 alterness and judgement.

For each cycle a day is considered either high, low, or critical. The high ($0 < x \leqslant 1$) times are regarded as energetic times, you are your most dynamic in the cycle. The low ($-1 \leqslant x < 0$) 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		f E	0*
	Later for automatic output of			
	results set PRINT/PAUSE			
	mode.		f E	1*
3	Key in the following:			
	Birthdate	MM.DDYYYY	A	
	Biodate	MM.DDYYYY	B	P**
				S**
				C**
4	To calculate the cycles for			
	Biodate + 1, 2, ...		C	P (day+1)
				S (day+1)
				C (day+1)
				P (day+2)
				S (day+2)
				C (day+2)
	To stop cycle		R/S	

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		f 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 f 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: **Outputs:**

6.071948 A ⟶ 6.07

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 ⟶ 0.82 *** P
0.00 *** S
-0.95 *** C

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

C →

0.63 *** P		
0.22 *** S	June 1	
-0.99 *** C		
0.40 *** P		
0.43 *** S	June 2	
-1.00 *** C		
0.14 *** P		
0.62 *** S	June 3	
-0.97 *** C		
-0.14 *** P		
0.78 *** S	June 4	
-0.91 *** C		
-0.40 *** P		
0.90 *** S	June 5	
-0.81 *** C		
-0.63 *** P		
0.97 *** S	June 6	
-0.69 *** C		

R/S → Ignore output

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 ***	
15.00 ***	P
26.50 ***	

14.00 ***
28.00 *** } S

10.00 ***
26.50 *** } C

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 ***
32.25 *** } P

21.00 *** S

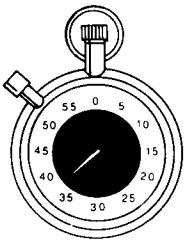
1.75 *** C

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.

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 cali-			
	brated, 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.		f 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.		f 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
	⋮	
	0.0155	
Finish time t_f = 9:38:34. →	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:	**Outputs:**	
C →	0.0000	

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 →		Ignore display.
f C →	0.0110	Elapsed time, H.MS
R/S →	0.0100	Last split
R/S →	0.0050	Previous split
⋮	⋮	
R/S →	0.0010	First split
R/S →	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.

Game of 21

001	*LBLA	
002	SPC	
003	SPC	
004	CF0	
005	CF1	
006	STOA	
007	1	
008	STOI	
009	GSB0	
010	STOB	
011	GSB5	
012	GSB9	
013	STO1	Bet,
014	STO3	
015	1	New game
016	STOI	
017	GSB0	
018	STOC	
019	GSB9	
020	STC2	
021	ST+3	
022	0	
023	GSB5	
024	SPC	
025	GSBe	
026	STO5	
027	STO7	
028	GSBe	
029	STO6	
030	ST+7	
031	RCL5	
032	x	
033	RCL7	
034	+	
035	2	
036	1	
037	STO0	
038	X=Y?	
039	GTO2	
040	RCL7	
041	GSB3	
042	R/S	----------
043	*LBLB	
044	GSBe	
045	ST+7	
046	RCL7	
047	RCL0	
048	X≠Y	
049	X>Y?	Hit
050	GTO4	
051	X≤Y?	
052	GSB3	
053	R/S	----------
054	*LBLD	
055	SF0	
056	1	
057	0	
058	ST+7	11 for Ace
059	RCL7	
060	GSB3	
061	RTN	
062	*LBLE	----------
063	CF0	
064	1	
065	0	
066	ST-7	1 for Ace
067	RCL7	
068	GSB3	
069	RTN	
070	*LBL4	----------
071	F0?	
072	GTOE	
073	RCLA	
074	CHS	
075	ST+9	player lose
076	RCL7	
077	EEX	
078	2	
079	÷	
080	-	
081	GTO6	----------
082	*LBL2	
083	RCL1	
084	RCL2	
085	x	
086	RCL3	HP BJ?
087	+	
088	RCL0	
089	X=Y?	
090	GTO7	
091	F1?	
092	RTN	
093	*LBL8	----------
094	.	
095	2	
096	1	
097	RCLA	player BJ!
098	1	
099	.	
100	5	
101	x	
102	STOA	
103	ST+9	
104	+	
105	*LBL6	----------
106	PSE	
107	PSE	
108	PSE	
109	PSE	Display with blinks.
110	RCL9	
111	DSP2	
112	F1?	

REGISTERS

0 21	1 1st D.	2 2nd D.	3 ΣD.	4	5 1stP.	6 2nd P.	7 ΣP.	8	9 Σ Bet
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A Bet	B 1st D.	C 2nd D.	D last K_i	E u_i	I Index

113	R/S	
114	RTN	
115	*LBL7	
116	FI?	
117	GTO4	
118	0	HP BJI
119	DSP9	
120	GTO6	
121	*LBL9	
122	1	
123	0	
124	X⇄Y	10 for J, Q, K
125	X≤Y?	
126	RTN	
127	X⇄Y	
128	RTN	
129	*LBL3	
130	EEX	Display
131	2	
132	÷	Bet.point
133	RCLA	
134	+	
135	RTN	
136	*LBL5	
137	DSP0	
138	PRTX	print
139	DSP2	
140	RTN	
141	*LBLa	
142	0	
143	STOI	
144	.	
145	5	
146	2	
147	8	
148	1	
149	1	
150	6	
151	3	
152	STOE	
153	*LBL0	Shuffle
154	RCLE	
155	9	
156	9	
157	7	
158	×	
159	FRC	
160	STOE	
161	1	
162	3	
163	×	
164	INT	
165	1	
166	+	
167	STOD	
168	DSZI	
169	GTO0	
170	RTN	
171	*LBLC	
172	CF0	
173	SPC	
174	RCLB	
175	GSB5	
176	RCLC	
177	GSB5	
178	SF1	
179	GSB2	
180	CF1	
181	RCL3	Stand and
182	*LBLb	
183	1	HP takes cards.
184	7	
185	X≤Y?	
186	GTOd	
187	GSBe	
188	ST+3	
189	RCL3	
190	2	
191	2	
192	X≤Y?	
193	GTO1	
194	RCL3	
195	GTOb	
196	*LBLd	
197	RCL7	
198	RCL3	
199	-	
200	X=0?	
201	GTO6	
202	X<0?	
203	GTO4	
204	*LBL1	
205	RCLA	
206	ST+9	
207	RCL7	
208	GSB3	
209	GTO6	
210	*LBLe	
211	1	
212	STOI	
213	GSB0	Next card
214	GSB5	
215	GSB9	
216	RTN	
217	*LBLc	
218	0	New player
219	STO9	
220	RTN	

LABELS

A Bet	B Hit	C Stand	D 11 for Ace	E 1 for Ace
a Shuffle	b Dealer	c New P.	d Used	e Next card
0 R. No.	1 P. Win	2 HP BJ?	3 Display	4 P. lose
5 Print	6 blinks	7 HP BJ!	8 P. BJ!	9 10 for J,Q,K

FLAGS

0	11 for Ace
1	HP BJ
2	
3	

SET STATUS

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

Dice

001	*LBLA	Input seed.
002	EEX	
003	CHS	
004	7	
005	×	
006	.	
007	5	
008	2	
009	8	
010	4	
011	1	
012	6	
013	+	
014	STO9	Store in R_9.
015	*LBLC	Reset Winnings (WIN) to 0.
016	0	
017	STO8	
018	*LBL1	Display Winnings (WIN).
019	DSP2	
020	FIX	
021	SF2	Set 1st Roll Flag.
022	RTN	
023	*LBLB	Store Bet in R_7.
024	STO7	
025	*LBL2	Get Dice Roll.
026	GSBD	
027	FSE	1st Roll?
028	F29	
029	GTO4	Yes.
030	GTO5	No.
031	*LBLD	Dice Roll Routine
032	GSB3	1st die (1 DIE).
033	STO6	Sum of 2 dice (2 DIE).
034	STO5	
035	GSB3	
036	ST*5	
037	1	
038	0	
039	÷	
040	RCL6	
041	+	
042	RCL5	
043	10ˣ	
044	×	
045	DSP1	
046	SCI	
047	RTN	
048	*LBL3	Random Number Generator.
049	3	
050	RCL9	
051	9	
052	9	
053	7	
054	×	
055	FRC	
056	STO9	

057	6	
058	×	
059	1	
060	+	
061	INT	
062	RTN	
063	*LBL4	1st Roll Routine
064	3	Lose on 2, 3, 12.
065	RCL5	
066	X=Y?	
067	GTO7	
068	2	
069	X=Y?	
070	GTO7	
071	R↓	
072	1	
073	2	
074	X=Y?	
075	GTO7	
076	R↓	Win 7, 11.
077	7	
078	X=Y?	
079	GTO6	
080	R↓	
081	1	
082	1	
083	X=Y?	
084	GTO6	
085	R↓	Current Roll (C ROLL).
086	STO4	
087	GTO2	Not 1st Roll.
088	*LBL5	
089	RCL5	Craps?
090	7	
091	X=Y?	
092	GTO7	YES
093	R↓	
094	RCL4	
095	X≠Y?	Match
096	GTO2	No
097	*LBL6	Win!
098	RCL7	
099	RCL8	Increment Winnings (WIN).
100	+	
101	STO8	
102	GTO1	
103	*LBL7	Lose!
104	RCL8	
105	RCL7	
106	-	Decrement Winnings (WIN).
107	STO8	
108	GTO1	

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	I

LABELS					FLAGS
A SEED	B BET	C RESET	D ROLL	E	0
a	b	c	d	e	1
0	1 DISPLAY	2 GET DIE	3 RANDOM	4 1 ROLL	2 1ST ROLL
5 ≠ 1 ROLL	6 WIN	7 LOSE	8	9	3

SET STATUS

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

Slot Machine

001	*LBLB	Recalls winnings
002	DSP2	
003	RCL0	
004	RTN	
005	*LBLE	Sets up new machine conditions.
006	CLRG	
007	STOE	
008	CLX	
009	DSP0	
010	RTN	
011	*LBLA	Play:
012	DSP3	Removes dollar played from winnings.
013	1	
014	ST-0	
015	RCLE	
016	EEX	Sets up wheels.
017	3	
018	×	
019	COS	
020	ABS	
021	STOE	
022	EEX	
023	6	
024	+	
025	LSTX	
026	-	
027	FRC	
028	STO4	
029	1	
030	0	
031	STO5	
032	×	
033	INT	
034	STO1	
035	CHS	
036	LSTX	
037	+	
038	RCL5	
039	×	
040	INT	
041	STO2	
042	CHS	
043	LSTX	
044	+	
045	RCL5	
046	×	
047	INT	
048	STO3	
049	1	Determine winnings.
050	RCL1	
051	X=Y?	
052	GTO1	
053	RCL2	
054	X≠Y?	
055	GTO9	
056	RCL3	
057	X≠0?	
058	GTO2	
059	1	
060	0	
061	ST+0	
062	RCL2	
063	X≠0?	Adds $10 for XX0.
064	GTO9	
065	9	
066	0	
067	GTO8	
068	*LBL2	Adds additional $90 for 000.
069	RCL2	
070	RCL3	
071	X≠Y?	
072	GTO9	
073	1	
074	0	
075	GTO8	Adds $10 for XXX,
076	*LBL1	
077	2	Adds $2 for 1XY.
078	ST+0	
079	1	
080	0	
081	RCL2	
082	X≠Y?	
083	GTO9	
084	3	Adds additional $3 for 11X.
085	*LBL8	
086	ST+0	
087	*LBL9	
088	RCL1	
089	1	
090	0	
091	÷	
092	DSP1	
093	PSE	
094	RCL2	
095	1	
096	0	
097	0	
098	÷	
099	+	
100	DSP2	
101	PSE	
102	RCL3	
103	EEX	
104	3	
105	÷	
106	+	
107	DSP3	
108	RTN	

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	I

LABELS					FLAGS
A Play	B Winnings	C	D	E Set up	0
a	b	c	d	e	1
0	1 Cherry loop	2 Used	3	4	2
5	6	7	8 Adds	9 End of play	3

SET STATUS

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

Submarine Hunt

001	*LBLC	Sets movement flag. 1 unit after sonar.
002	1	
003	STOC	
004	SF0	
005	RTN	
006	*LBLD	Sensitivity toggle switch 1.9→2.9→1.9 etc.
007	RCLD	
008	1	
009	X=Y?	
010	+	
011	STOD	
012	RTN	
013	*LBLE	Start
014	CLRG	
015	CF0	Determine initial position using input seed.
016	.	
017	5	
018	2	
019	8	
020	4	
021	1	
022	6	
023	×	
024	STO0	
025	GSBd	
026	STO1	
027	GSBd	
028	STO2	
029	DSP0	
030	1	
031	STOD	
032	CLX	
033	RTN	
034	*LBLA	Fire Depth Charge
035	1	
036	ST+7	
037	R↓	
038	SF1	
039	GSBc	
040	X≠0?	
041	GTO0	
042	1	
043	STO5	
044	GSBa	Move Submarine
045	RTN	
046	*LBL0	Display Hit!
047	9	
048	1/X	
049	DSP3	
050	PSE	
051	DSP5	
052	PSE	
053	DSP7	
054	PSE	
055	DSP9	
056	PSE	
057	RTN	
058	*LBLB	Sonar Reading
059	1	
060	ST+8	
061	R↓	
062	DSP0	
063	CF1	
064	GSBc	
065	RCLC	
066	STO5	
067	F0?	
068	GSBa	
069	RCL3	
070	RTN	
071	*LBLa	Submarine Move Routine
072	GSBd	
073	4	
074	X⇄Y	
075	X>Y?	
076	GTO0	
077	RCL5	
078	CHS	
079	GTO1	
080	*LBL0	
081	RCL5	
082	*LBL1	
083	STO6	
084	GSBd	
085	5	
086	X>Y?	
087	GTO0	
088	SF2	
089	RCL1	
090	GTO1	
091	*LBL0	
092	RCL2	
093	*LBL1	
094	RCL6	
095	+	
096	X<0?	
097	GTO0	
098	9	
099	X⇄Y	
100	X≤Y?	
101	GTO1	
102	*LBL0	
103	RCL6	
104	2	
105	×	
106	−	
107	*LBL1	
108	F2?	
109	GTO0	
110	STO2	
111	GTO1	
112	*LBL0	

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

113	STO1	
114	*LBL1	
115	RCL3	
116	DSP0	
117	RTN	
118	*LBLc	Determine Response and Calculate Distance.
119	RCL2	
120	-	
121	X⇄Y	
122	RCL1	
123	-	
124	→P	
125	STO4	
126	F1?	
127	GTO0	
128	RCLD	
129	-	
130	*LBL0	
131	.	
132	9	
133	-	
134	X<0?	
135	GTO0	
136	0	
137	GTO1	
138	*LBL0	
139	1	
140	*LBL1	
141	STO3	
142	RTN	
143	*LBLd	Random Number Generator
144	RCL0	
145	9	
146	9	
147	7	
148	×	
149	FRC	
150	STO0	
151	1	
152	0	
153	×	
154	INT	
155	RTN	
156	*LBLe	Check Status
157	DSP2	
158	RCL7	
159	RCL8	
160	1	
161	0	
162	0	
163	÷	
164	+	
165	RTN	

LABELS

A Depth charge	B Sonar	C Option	D Dist	E Start
a Move sub	b	c d	d random ≠	e
0	1	2	3	4
5	6	7	8	9

FLAGS

0	Motion?
1	Depth charge
2	P_1?
3	

SET STATUS

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

Artillery

001	*LBLa	INITIALIZATION
002	FIX	
003	DSP2	
004	.	
005	5	
006	2	
007	8	
008	4	
009	1	
010	6	
011	3	
012	STO0	R_0←seed = .5284163
013	3	
014	STOC	R_C←spot rating = 3
015	5	
016	0	
017	0	
018	STOB	R_B←speed = 500
019	EEX	
020	2	
021	STOD	R_D←kill range = 100
022	X²	
023	STO9	Max. gun range = 10000
024	RTN	----------
025	*LBLA	START NEW BATTLE
026	SPC	
027	0	
028	STOI	Rounds = 0
029	RCL9	
030	2	
031	÷	
032	ENT↑	
033	GSB0	
034	+	Generate target range.
035	3	
036	6	
037	0	
038	GSB0	
039	STO8	Generate target bearing.
040	R↑	
041	→R	
042	STO1	Save target N-S.
043	X⇄Y	
044	STO2	Save target E-W.
045	4	
046	5	
047	ST÷8	
048	RCL8	
049	DSP0	Display target 45° sector.
050	RND	
051	×	
052	RTN	----------
053	*LBLC	Routine to generate and
054	RCL2	display target's last position.
055	RCL1	
056	→P	

057	EEX	
058	4	
059	÷	
060	X⇄Y	
061	X>0?	
062	GTO2	
063	3	
064	6	
065	0	
066	+	
067	*LBL2	
068	DSP0	
069	RND	
070	+	
071	DSP4	Display brg. range.
072	RTN	----------
073	*LBLb	
074	STOB	Set new SPEED.
075	RTN	----------
076	*LBLc	
077	STOC	Set new spotter rating.
078	RTN	----------
079	*LBLd	
080	STOD	Set new KILL range.
081	RTN	----------
082	*LBL1	Routine to calc. move-
083	CF0	ment of target in N-S and
084	X<0?	E-W directions.
085	SF0	
086	RCLB	
087	GSB0	
088	F0?	
089	CHS	
090	-	
091	RTN	----------
092	*LBLe	FIRE! Routine - - -
093	*LBLE	
094	ISZI	
095	X⇄Y	
096	DSP7	
097	PRTX	Output bearing.
098	X⇄Y	
099	DSP1	
100	PRTX	Output elevation.
101	ENT↑	
102	+	
103	SIN	
104	RCL9	
105	×	
106	→R	
107	STO3	Save round's N-S.
108	X⇄Y	
109	STO4	Save round's E-W.
110	RCL1	
111	GSB1	
112	STO1	Move target N-S.

REGISTERS

0	1	2	3	4	5	6	7	8	9
Seed	Targ N-S	Targ E-W	Shell N-S	Shell E-W			Used	Used	Gun Range
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
	SPEED	Spot Rating	KILL Range		Rounds Fired

```
113  RCL2
114  GSB1
115  STO2
116   CHS
117  RCL4
118     +
119  RCL3
120  RCL1
121     -
122    →P
123  STO7
124  RCLD
125   X⇄Y
126  X>Y?
127  GTO7
128  RCLC
129  RCLB
130    R↓
131    R↓
132  PRST
133   R/S
134 *LBL7
135    R↓
136    R↓
137  RCL2
138  RCL1
139    →P
140     5
141     0
142     0
143 *LBL8
144   PSE
145  X>Y?
146  GTO8
147    R↓
148    R↓
149     -
150     3
151     0
152     ÷
153     4
154  RCLC
155     -
156  GSB9
157 *LBL5
158  X<0?
159  GTO6
160  DSP0
161   RND
162  STO8
163  RCL7
164     .
165     2
166  ENT↑
167     4
168  RCLC
```

Move target E–W.

Generate "miss" data.

R_7 ←miss

Test if round hit within KILL range.

WIN! Print stack.

Otherwise, test if target has closed within 500 meters.

LOSE! So flash "500.0" repeatedly.

If no decision, generate spotter's estimate of hour angle to shell hit.

R_8 ←hour angle.

Generate up to 20% range error for each level of spotter incompetence.

```
169     -
170     ×
171     ×
172  X=0?
173  GTO3
174  GSB9
175  RCLD
176 *LBL3
177  X≤Y?
178   X⇄Y
179  RCL9
180     ÷
181  DSP3
182   RND
183  RCL8
184     +
185  DSP4
186  PRTX
187   RTN
188 *LBL6
189     1
190     2
191     +
192  GTO5
193 *LBL9
194     -
195  LSTX
196  ENT↑
197     +
198  GSB0
199     +
200   RTN
201 *LBL0
202  RCL0
203     9
204     9
205     7
206     ×
207   FRC
208  STO0
209     ×
210   RTN
```

Display and print hh.DDDD feedback to gunner.

Patch to bring hour angle within clock numbers.

Routine to generate spotter's estimate of "miss" data.

Random number generator.

LABELS					FLAGS
A START	B	C →TARGET	D	E FIRE!	0 Switch
a INIT.	b New speed	c New spot	d New kill	e FIRE!	1
0 Ran. Gen.	1 ΔTarget	2 Tgt brg.rng	3 Used	4	2
5 Used	6 Patch	7 Tests	8 LOSS!	9 Err. Gen.	3

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n 4	

Space War

001	*LBLA	Initialize.
002	CLRG	If seed = 0, use π.
003	X=0?	
004	Pi	
005	STO9	
006	1	
007	0	
008	STO4	
009	2	Point to R_A and prepare to
010	0	compute random starting
011	STOI	positions.
012	*LBL9	Compute a position. Check
013	GSB1	not occupied.
014	GSB0	
015	F2?	
016	GTO9	
017	STOi	Store position.
018	ISZI	
019	RCLI	
020	2	
021	5	Exit loop after filling R_E.
022	X≠Y?	
023	GTO9	
024	EEX	
025	3	
026	STO6	Energy = 1000.
027	3	
028	STO7	Torpedoes = 3.
029	STO9	
030	1	
031	8	
032	STO8	Days = 18
033	RCLE	Display position of
034	RTN	Kittyhawk.
035	*LBL1	Generate one starting posi-
036	RCL9	tion of the form QQ.SS,
037	9	where QQ is quadrant, SS
038	9	is sector.
039	7	
040	×	
041	FRC	
042	STO9	
043	EEX	
044	4	
045	×	
046	INT	
047	GSB5	
048	GSB5	
049	RTN	
050	*LBL8	Routine tests to see if
051	RCLA	position in X-register is
052	X=Y?	already occupied.
053	SF2	
054	CLX	
055	RCLG	
056	X=Y?	
057	SF2	
058	CLX	
059	RCLC	
060	X=Y?	
061	SF2	
062	CLX	
063	RCLD	
064	X=Y?	
065	SF2	Set flag 2 if position
066	R↓	occupied.
067	RTN	
068	*LBLC	Long Scan.
069	4	
070	FIX	
071	9	
072	9	
073	9	Store constant 0.00400400
074	÷	
075	STO2	
076	DSP8	
077	SPC	
078	RCLE	
079	INT	
080	STO0	
081	RCL4	
082	+	Scan line above KH.
083	GSB3	
084	RCL0	
085	GSB3	Scan line with KH.
086	RCL0	
087	RCL4	
088	-	Scan line below KH.
089	GSB3	
090	RTN	
091	*LBL3	Routine scans one line,
092	RCL2	i.e., 3 quadrants.
093	STO1	
094	R↓	
095	STO3	
096	ST+1	R_1 ←QQ.00400400,
097	1	where QQ refers to middle
098	-	quadrant.
099	GSB0	
100	GSB5	
101	GSB5	First quadrant.
102	ST+1	
103	RCL3	
104	GSB0	
105	EEX	
106	5	Middle quadrant.
107	÷	
108	ST+1	
109	RCL3	
110	1	
111	+	
112	GSB0	

REGISTERS

0	1	2	3	4	5	6	7	8	9
Used	Used	Used	Used	10	Used	Energy	Torpedoes	Days	Alglogs
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
Alglog 1	Alglog 2	Alglog 3	Base	Kittyhawk	Pointer

113	EEX	
114	8	Last quadrant.
115	÷	
116	ST+1	
117	RCL1	
118	PRTX	Print line.
119	RTN	
120	*LBL0	
121	0	Routine checks one quadrant for presence of Alglogs or Base.
122	STO5	
123	R↓	
124	RCLA	
125	INT	
126	X=Y?	Forms a two-digit number AB, where A is no. of Alglogs, B no. of Bases in quadrant.
127	GSB1	
128	CLX	
129	RCLB	
130	INT	
131	X=Y?	
132	GSB1	
133	CLX	
134	RCLC	
135	INT	
136	X=Y?	
137	GSB1	
138	CLX	
139	RCLD	
140	INT	
141	X=Y?	
142	GSB2	
143	RCL5	
144	RTN	
145	*LBL1	
146	RCL4	Alglog detected—add 10 to R_5.
147	ST+5	
148	RTN	
149	*LBL2	
150	1	Base found—add 1 to R_5.
151	ST+5	
152	RTN	
153	*LBLE	Move.
154	→R	Convert (r, θ) to $(\Delta x, \Delta y)$.
155	FIX	
156	DSP1	
157	RND	
158	STO0	$\Delta Q_x.\Delta S_x$
159	X⇄Y	
160	RND	$\Delta Q_y.\Delta S_y$
161	DSP2	
162	RCLE	
163	GSB5	
164	STO3	
165	INT	
166	RCLE	
167	GSB4	
168	STO2	
169	INT	
170	GSB5	
171	FRC	
172	+	$Q_y.S_y$
173	+	$(Q_y + \Delta Q_y).(S_y + \Delta S_y)$
174	FRC	
175	LSTX	
176	INT	
177	GSB4	
178	+	$Q_y\ O.S_y$ (new Y-position)
179	RCL2	
180	FRC	
181	RCL3	
182	FRC	
183	GSB4	
184	INT	
185	+	$Q_x.S_x$
186	RCL0	
187	+	$(Q_x + \Delta Q_x).(S_x + \Delta S_x)$
188	INT	
189	LSTX	
190	FRC	
191	GSB5	
192	+	$Q_x.OS_x$
193	*LBL7	
194	+	$Q_yQ_x.S_yS_x$ (QQ.SS)
195	GSB8	Test not occupied.
196	F2?	
197	GTO8	
198	STOE	
199	1	
200	ST-8	Decrement no. days remaining.
201	RCL8	
202	X<0?	If negative, game over.
203	GTO6	Otherwise display new position.
204	RCLE	
205	RTN	
206	*LBL6	
207	CLX	Ran out of days—flash zeros.
208	PSE	
209	GTO6	
210	*LBL4	
211	RCL4	Multiply by 10.
212	×	
213	RTN	
214	*LBL5	
215	RCL4	
216	÷	Divide by 10.
217	RTN	
218	*LBL8	
219	.	Position occupied—add 0.1 and try again.
220	1	
221	GTO7	
222	*LBLa	
223	RCL8	Display remaining days.
224	RTN	

LABELS

A Start	B	C Long Scan	D	E Move	0
a →Days	b	c	d	e	1
0 Used	1 Used	2 Used	3 Scan line	4 10x	2 Occupied
5 10 ÷	6 Flash zeros	7 Exit move	8 Occupied?	9 Start loop	3

SET STATUS

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

001	*LBLA	Short Scan.
002	FIX	
003	DSP9	
004	SPC	A "3" marks KH,
005	P⇄S	"4" marks Alglog,
006	9	"7" marks Base.
007	STOI	
008	0	
009	*LBL9	
010	STOi	Clear R_{s0} – R_{s9} to hold
011	DSZI	scans of rows 0 thru 9,
012	GTO9	respectively.
013	STO0	
014	3	
015	RCLE	Locate KH in quadrant.
016	GSB0	
017	4	
018	RCLA	Check Alglog 1.
019	GSB0	
020	4	
021	RCLB	Check Alglog 2.
022	GSB0	
023	4	
024	RCLC	Check Alglog 3.
025	GSB0	
026	7	
027	RCLD	Check for Base.
028	GSB0	
029	9	
030	STOI	
031	*LBL8	Print R_{s9}, R_{s8}, ..., R_{s0} as
032	RCLi	rows 9, 8, ..., 0.
033	PRTX	
034	DSZI	
035	GTO8	
036	RCL0	
037	PRTX	
038	P⇄S	
039	RTN	
040	*LBL0	Routine tests whether an
041	ENT↑	object is in KH's quadrant.
042	INT	
043	RCLE	
044	INT	If not, return.
045	X≠Y?	
046	RTN	
047	R↓	
048	R↓	If so, locate 3, 4, or 7 in
049	FRC	proper sector of proper
050	GSB5	row, represented by R_{s0}–R_{s9}
051	INT	
052	STOI	
053	CLX	
054	LSTX	
055	FRC	
056	GSB5	
057	CHS	
058	10^x	
059	×	3, 4, or 7 times 10**(–S_x)
060	ST+i	Add to register S_y.
061	RTN	
062	*LBLC	Torpedo.
063	CF0	
064	STO0	Save angle.
065	RCL7	
066	1	If no torpedoes remain,
067	X>Y?	display "Error".
068	GTOB	Otherwise subtract one
069	-	from no. torpedoes.
070	STO7	
071	1	
072	9	
073	STOI	
074	GSB1	
075	GSB1	Check if Alglogs 1, 2, and
076	GSB1	3 are in the path of the
077	RCL9	torpedo.
078	CF0	Display no. Alglogs left.
079	RTN	
080	*LBL1	Routine tests if Alglog
081	ISZI	will be hit by torpedo.
082	F0?	If F0 set, an Alglog has
083	RTN	been hit—return.
084	RCLi	If Alglog not in same
085	INT	quadrant as KH, return.
086	RCLE	
087	INT	
088	X≠Y?	
089	RTN	
090	GSB7	Find angle to Alglog.
091	CLX	
092	RCL0	Compare to angle of fire.
093	-	
094	ABS	
095	1	If Alglog 1° or more away,
096	X≤Y?	no hit.
097	RTN	
098	1	If hit, store –1 as Alglog's
099	CHS	position.
100	STOi	
101	SF0	F0 set to indicate torpedo
102	1	is spent.
103	ST-9	Decrement no. Alglogs.
104	RTN	
105	*LBL7	Routine finds angle and
106	RCLi	distance from KH to
107	FRC	Alglog.
108	GSB5	
109	STOI	
110	INT	
111	RCLE	
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

113	GSB5	
114	INT	
115	-	
116	RCL1	
117	FRC	
118	RCLE	
119	GSB5	
120	FRC	
121	-	
122	GSB5	
123	→P	Return r in X, θ in Y.
124	RTN	
125	*LBLD	
126	STO0	Phaser. Save energy and subtract from total.
127	ST-6	
128	1	
129	9	
130	STOI	
131	GSB4	Check if Alglogs 1, 2, and 3 are in range.
132	GSB4	
133	GSB4	
134	RCL9	Display no. Alglogs left.
135	RTN	
136	*LBL4	
137	RCL6	Test if Alglog hit by phaser.
138	X<0?	
139	GTO6	If energy < 0, exit.
140	ISZI	
141	RCLI	
142	INT	If Alglog not in KH's quadrant, return.
143	RCLE	
144	INT	
145	X≠Y?	
146	RTN	
147	EEX	Deduct 100 for shields.
148	2	
149	ST-6	
150	GSB7	Find distance r to Alglog.
151	X²	
152	EEX	Energy input must be greater than r^2 + 100.
153	2	
154	+	
155	RCL0	
156	X≤Y?	
157	RTN	
158	1	If hit, store −1 as Alglog's position.
159	CHS	
160	STOI	
161	1	
162	ST-9	Decrement no. Alglogs.
163	RTN	
164	*LBL6	
165	CLX	Out of energy – flash zeros.
166	PSE	
167	GTO6	
168	*LBLE	Dock.
169	RCLC	
170	INT	If Base and KH not in same quadrant, display same En.Torp.
171	RCLE	
172	INT	
173	X≠Y?	
174	GTOa	
175	RCLD	
176	FRC	
177	GSB5	
178	STO0	
179	INT	
180	RCLE	
181	FRC	
182	GSB5	
183	STO1	
184	INT	
185	-	
186	ABS	KH must be within 1 sector of Base in Y-direction.
187	2	
188	X≤Y?	
189	GTOa	
190	RCL0	
191	FRC	
192	GSB5	
193	RCL1	
194	FRC	
195	GSB5	
196	-	
197	ABS	
198	2	Also must be writhin 1 sector of Base in X-direction.
199	X≤Y?	
200	GTOa	
201	EEX	If successful, get 1000 units energy and 3 torpedoes.
202	3	
203	STO6	
204	3	
205	STO7	
206	GTO0	
207	*LBLa	
208	RCL6	If failed, display old energy and torpedoes.
209	RCL7	
210	*LBL0	
211	1	Form display Energy.Torpedoes.
212	0	
213	÷	
214	X⇄Y	
215	INT	
216	+	
217	FIX	
218	DSP1	
219	RTN	
220	*LBL5	
221	1	Multiply by 10.
222	0	
223	x	
224	RTN	

LABELS

A Short scan	B	C Torpedo	D Phaser	E Dock
a →En.torp	b	c	d	e
0 Used	1 Torp hit?	2	3	4 Phaser hit?
5 10x	6 Flash zeros	7 KH to Alglog	8 Print reg.	9 Clear reg.

FLAGS

0	Torpedo
1	
2	
3	

SET STATUS

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

Super Bagels

001	*LBLB	Number of digits.
002	INT	
003	X=0?	
004	GTO6	
005	8	
006	X≷Y	
007	X>Y?	
008	GTO6	
009	RCLB	
010	FRC	
011	+	
012	GTO0	
013	*LBLC	Maximum digit.
014	INT	
015	9	
016	X≷Y	
017	X>Y?	
018	GTO6	
019	RCLA	
020	÷	
021	RCLB	
022	INT	
023	+	
024	GTO0	
025	*LBLD	Number of guesses.
026	RCLC	
027	RTN	
028	*LBLA	Input seed.
029	SPC	
030	CF0	
031	CLRG	
032	X=0?	
033	Pi	
034	STOE	
035	1	
036	0	
037	STOA	
038	4	
039	.	
040	5	
041	*LBL0	
042	STOB	
043	FIX	
044	DSP1	
045	RTN	
046	*LBLE	Input guess.
047	CF2	
048	F0?	
049	GTOa	
050	ENT↑	
051	SF0	
052	RCLE	
053	RCLB	
054	×	
055	×	
056	X=0?	
057	LSTX	
058	LN	
059	EEX	
060	7	
061	×	
062	ABS	
063	D→R	
064	1/X	
065	STOE	
066	R↓	
067	*LBLa	
068	INT	
069	STOD	
070	RCLE	
071	1/X	
072	R→D	
073	SF1	
074	GSB1	
075	CLX	
076	STO9	
077	STO0	
078	P≷S	
079	RCLD	
080	CF1	
081	GSB1	
082	RCLB	
083	STOI	
084	*LBL3	Compare for equality.
085	RCLi	
086	P≷S	
087	RCLi	
088	X≠Y?	
089	GTO0	
090	CHS	
091	X≷Y	
092	CHS	
093	X≷Y	
094	RCLA	
095	ST+9	
096	R↓	
097	*LBL0	
098	STOi	
099	X≷Y	
100	P≷S	
101	STOi	
102	DSZI	
103	GTO3	
104	RCLB	
105	STO0	
106	*LBL9	Check for digits out of position.
107	RCL0	
108	STOI	
109	RCLi	
110	P≷S	
111	X<0?	
112	GTO5	

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	
114	RCLB	
115	STOI	
116	*LBL6	
117	CLX	
118	RCLI	
119	X=Y?	
120	GTO8	
121	DSZI	
122	GTO7	
123	GTO5	
124	*LBL7	
125	CLX	
126	RCLI	
127	X=Y?	
128	GTO6	
129	DSZI	
130	GTO8	
131	GTO5	
132	*LBL8	
133	CLX	
134	RCLI	
135	X=Y?	
136	GTO0	
137	DSZI	
138	GTO6	
139	GTO5	
140	*LBL0	
141	CHS	Mark digits used.
142	STOI	
143	RCLA	
144	ST+0	
145	*LBL5	
146	PZS	
147	CLX	
148	STOI	
149	DSZI	
150	GTO9	
151	PZS	
152	RCLD	Prepare output for user.
153	RCLB	
154	INT	
155	10^x	
156	÷	
157	FRC	
158	RCL0	
159	RCLA	
160	÷	
161	+	
162	RCL9	
163	+	
164	RCLB	
165	STOI	
166	CLX	
167	GSBe	
168	CLX	
169	RCLC	
170	1	
171	+	
172	STOC	
173	DSPI	
174	X⇄Y	
175	PRTX	
176	F2?	Win?
177	RTN	
178	X⇄Y	
179	DSP0	
180	PRTX	Output guesses to win.
181	RTN	
182	*LBL1	Decompose number.
183	RCLB	
184	FRC	
185	RCLA	
186	1/X	
187	+	
188	STO0	
189	CLX	
190	RCLB	
191	STOI	
192	INT	
193	10^x	
194	÷	
195	ABS	
196	STO9	
197	*LBL2	
198	RCL5	
199	FRC	
200	RCLA	
201	×	
202	STO9	
203	RCL0	
204	X⇄Y	
205	F1?	
206	×	
207	1	
208	+	
209	INT	
210	STOI	
211	DSZI	
212	GTO2	
213	RTN	
214	*LBLe	Clear registers for next guess.
215	STO1	
216	STO2	
217	STO3	
218	STO4	
219	STO5	
220	STO6	
221	STO7	
222	STO8	
223	RTN	

LABELS

A Start	B #digits	C Max digit	D #guesses	E Guess
a Used	b	c	d	e Used
0 Used	1	2 Used	3 Used	4
5 Used	6 Used	7 Used	8 Used	9 Used

FLAGS

0	Used
1	
2	
3	

SET STATUS

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

Nim_k

001	*LBLA	L mode.
002	3	
003	2	
004	CHS	
005	GTO0	
006	*LBLD	
007	DSP1	
008	PRTX	Print pile.# taken.
009	STOI	
010	INT	
011	CHS	
012	10^x	
013	RCLI	
014	RCLI	
015	FRC	
016	1	
017	0	
018	×	
019	INT	
020	X>Y?	
021	GTOa	Error, taking more than in
022	X=0?	pile.
023	GTOa	Error, taking zero delete #
024	ST-i	from pile.
025	R↑	
026	×	Shift by 10.
027	1	
028	+	
029	CHS	
030	RCLA	
031	+	Adjust A.
032	STOA	
033	RCLB	
034	STOI	
035	DSPi	Readjust display.
036	RCLA	
037	PRTX	Print new combination.
038	RTN	
039	*LBLC	
040	3	
041	2	
042	*LBL0	
043	DSP0	
044	PRTX	
045	STOC	Initialize C to –32, L mode
046	CF0	32, W mode.
047	CF1	
048	*LBL9	
049	RCLC	
050	2	
051	÷	Get 2^i
052	STOC	
053	ABS	
054	1	
055	X=Y?	
056	GTO1	Exit if C = 1.

057	RCLB	
058	STOI	Set I to right-most pile.
059	*LBL8	
060	GSB2	
061	X≤Y?	Test for binary unit in
062	ST+0	designated pile/column.
063	ISZI	Accumulate Σc_{ij}.
064	GTO8	Cycle till zero.
065	3	
066	STX0	Double R0 for Σc_{ij}.
067	RCL0	
068	RCL0	
069	RCLE	
070	1	
071	+	
072	X>Y?	
073	GTO0	
074	3	$k < \Sigma c_{ij}$
075	RCLC	
076	ABS	
077	X>Y?	
078	SF0	Set F0 for Σc_{ij} mod
079	+	(k + 1) = 0.
080	R↓	
081	*LBL0	
082	ENT↑	
083	R↓	
084	÷	
085	INT	
086	R↑	
087	×	
088	–	Σc_{ij} mod (k + 1).
089	STO0	
090	RCLC	
091	X<0?	neg. = L mode
092	F0?	
093	GTO0	
094	3	F0 not set.
095	RCLC	
096	ABS	
097	X>Y?	
098	GTO0	
099	RCL0	
100	1	
101	–	Subtract 1 from Σc_{ij}.
102	X<0?	If neg, replace with k.
103	RCLE	
104	STO0	
105	*LBL0	
106	RCL0	
107	X=0?	If Σc_{ij} = 0 go to next
108	GTO9	column.
109	1	
110	0	
111	STOC	Initialize pointer to RS0.
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 $k.p_1 p_2 p_3 \ldots$	B no. of piles	C 2^i	D 10, pointer	E k	I Used

113	STOI	
114	RCLB	
115	RCLi	
116	X=0?	If pointer empty, use right
117	X<>Y	most pile no.
118	STOI	
119	*LBL6	
120	GSB2	Test for binary unit.
121	X>Y?	
122	GTO8	
123	SF1	Set F1, at least one pile
124	SF2	decreased.
125	DSZi	Set F2, this pile decreased.
126	GTO6	Step to next pile.
127	*LBL0	
128	F2?	
129	GTO8	
130	DSZI	
131	GTO6	
132	*LBL8	
133	RCLD	Recall D to I.
134	XEI	
135	STOi	Store pile # in pointer.
136	ISZI	Increment I (D).
137	CLX	0 to I.
138	XEI	
139	STOD	I to D.
140	DSZi	Decrement Σcol by 1.
141	GTO7	Go to new pile or new
142	GTO9	column.
143	*LBL2	Subroutine.
144	RCLi	Objects in pile.
145	RCLC	
146	ABS	
147	÷	Divided by 2^i.
148	FRC	
149	.	
150	5	Compare fractional part
151	RTN	to 0.5.
152	*LBL1	
153	F1?	Any pile decreased, exit
154	GTO8	with no change.
155	RCLB	
156	STOI	
157	*LBL5	
158	RCLi	
159	DSZi	
160	X>0?	Otherwise decrement right-
161	GTO8	most pile by 1.
162	STOi	
163	DSZI	
164	GTO5	
165	*LBL8	Exit routine.
166	RCLB	Recall no. piles.
167	RCLA	Display.
168	RCLC	2^i

169	RCLE	k
170	PZS	
171	CLRG	Clear pointer reg's.
172	PZS	
173	STOE	Restore E, C, A, B.
174	R↓	
175	STOC	
176	R↓	
177	STOA	
178	R↓	
179	STOB	
180	STOI	
181	DSPi	Adjust display.
182	0	
183	*LBL4	
184	RCLi	Build display.
185	+	
186	1	
187	0	
188	÷	Shift by 10.
189	DSZI	
190	GTO4	
191	RCLE	Recall k.
192	+	Add to combination.
193	STOA	
194	SPC	Space and print display.
195	PRTX	
196	RTN	
197	*LBLE	Entry routine.
198	CLRG	Clear all registers.
199	PZS	
200	CLRG	
201	STOA	Store input in A.
202	ENT↑	
203	INT	
204	STOE	
205	−	Input fractional part.
206	*LBL3	
207	ISZI	
208	1	Shift by 10 to get
209	0	individual pile #'s.
210	x	
211	ENT↑	
212	INT	
213	STOi	
214	−	
215	X≠0?	Continue till fractional
216	GTO3	part = 0.
217	RCLI	
218	DSPi	Set display.
219	STOB	
220	RCLA	
221	PRTX	Print input.
222	RTN	

LABELS

A L mode	B	C W mode	D Delete	E Enter
a	b	c	d	e
0 Used	1 Used	2 Used	3 Used	4 Used
5 Used	6 Used	7 Used	8 Used	9 Used

FLAGS

0	L/W
1	≥ 1 pile
2	this pile
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☑	DEG	☑	FIX	☑
1	☐	☑	GRAD	☐	SCI	☐
2	☐	☑	RAD	☐	ENG	☐
3	☐	☑			n 4	

Queen Board

001	*LBLA	
002	STO1	Current position.
003	GSBE	R_1
004	1	
005	X=Y?	
006	GTO2	
007	7	
008	STOI	$7 \rightarrow R_I$
009	*LBL1	
010	RCL1	
011	RCLI	
012	EEX	
013	1	
014	x	
015	+	
016	STO2	$10K + R_1 \rightarrow R_2$
017	GSBE	Position good?
018	1	
019	X=Y?	
020	GTO0	Yes, recall R_2.
021	RCLI	
022	ST+2	$K + R_2 \rightarrow R_2$
023	RCL2	Position good?
024	GSBE	
025	1	
026	X=Y?	
027	GTO6	Yes, recall R_2.
028	RCLI	
029	EEX	
030	1	
031	x	
032	ST+2	$10K + R_2 \rightarrow R_2$
033	RCL2	
034	GSBE	Position good?
035	1	
036	X=Y?	
037	GTO0	Yes, recall R_2.
038	DSZI	
039	GTO1	
040	RCL1	
041	*LBL2	
042	EEX	
043	1	Default move.
044	ST+1	$10 + R_1 \rightarrow R_1$
045	RCL1	
046	RTN	
047	*LBL0	
048	RCL2	
049	RTN	
050	*LBLE	Test for good position.
051	1	
052	5	
053	8	
054	X=Y?	$158 = R_2$?
055	GTO9	
056	3	
057	1	
058	-	
059	X=Y?	$127 = R_2$?
060	GTO9	
061	1	
062	-	
063	X=Y?	$126 = R_2$?
064	GTO9	
065	5	
066	1	
067	-	
068	X=Y?	$75 = R_2$?
069	GTO9	
070	2	
071	-	
072	X=Y?	$73 = R_2$?
073	GTO9	
074	2	
075	9	
076	-	
077	X=Y?	$44 = R_2$?
078	GTO9	
079	3	
080	-	
081	X=Y?	$41 = R_2$?
082	GTO9	
083	RTN	
084	*LBL9	
085	1	
086	RTN	

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	I K

LABELS					FLAGS
A Used	B	C	D	E Used	0
a	b	c	d	e	1
0 Used	1 Used	2 Used	3	4	2
5	6	7	8	9 Used	3

SET STATUS

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

Hexapawn

001	*LBLC	Machine first.
002	8	
003	3	
004	8	
005	8	Configurations for move 1.
006	6	
007	0	
008	7	
009	STO1	
010	3	
011	1	
012	3	Configurations for move 2.
013	9	
014	5	
015	8	
016	3	
017	STO2	
018	3	
019	4	Configurations for move 3.
020	3	
021	1	
022	4	
023	STO3	
024	GTO0	
025	*LBL6	Player first.
026	1	
027	6	
028	7	
029	7	Configurations for move 1.
030	7	
031	2	
032	1	
033	5	
034	STO1	
035	1	
036	6	
037	7	Configurations for move 2.
038	5	
039	6	
040	7	
041	3	
042	5	
043	STO2	
044	5	
045	2	Configurations for move 3.
046	4	
047	4	
048	1	
049	3	
050	STO3	
051	*LBL0	
052	0	
053	STO4	
054	STO5	
055	STO6	
056	STO7	
057	*LBL9	
058	Pi	
059	+	
060	STO6	
061	GTO9	
062	*LBLA	Move
063	STO8	
064	RCL4	
065	STO9	
066	RCL5	
067	STO7	
068	2	
069	RCL8	
070	Y*	
071	STO5	
072	3	
073	STO1	
074	RCL6	Generate a random number between 1 and 3.
075	Pi	
076	+	
077	X²	
078	FRC	
079	STO6	
080	3	
081	×	
082	1	
083	+	
084	INT	
085	STO4	
086	*LBL1	BEGIN loop 1
087	3	
088	RCL4	
089	1	INCREMENT trial move
090	+	
091	X>Y?	
092	1	
093	STO4	
094	X⇄I	
095	RCLi	
096	X⇄Y	
097	X⇄I	
098	R↓	
099	RCL5	
100	÷	
101	FRC	
102	.	
103	5	
104	X≤Y?	IF trial move OK
105	GTO4	THEN exit
106	DSZI	IF less than 3 tries
107	GTO1	THEN REPEAT loop 1
108	RCL7	ELSE no move possible.
109	STO5	
110	STO4	
111	CLX	
112	RTN	

REGISTERS

0	1 Moves #1	2 Moves #2	3 Moves #3	4 Trial move	5 2↑(Config)	6 Random	7 2↑(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
114   RCL4
115    RTN
116  #LBLB
117   RCL4
118    X⇄I
119   RCL5
120      2
121      ÷
122   ST-i
123    R↓
124    X⇄I
125    RTN
```

DISPLAY move

- - - - - - - - - - - -

PUNISH
Remove last configuration from move list

LABELS					FLAGS
A Move	B Punish	C Mach. 1st	D	E	0
a	b	c Player 1st	d	e	1
0 Used	1 Loop 1	2	3	4 Display	2
5	6	7	8	9 Used	3

SET STATUS

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

Tic-Tac-Toe

001	*LBLA	Start a new game.
002	0	
003	STOE	
004	1	
005	CHS	
006	STO0	
007	2	
008	GSB6	
009	GSB8	
010	GSBD	
011	GSBe	
012	RCLI	
013	RCLE	
014	DSP9	
015	R/S	
016	GSB6	Check for the 1st move of player.
017	GSB8	
018	GSBd	
019	RCLI	
020	INT	
021	RCLI	
022	STOB	
023	*LBL0	
024	+	
025	GSBB	
026	STOI	
027	GSBD	
028	GSBe	
029	RCLI	
030	DSP9	
031	RCLE	
032	+	
033	R/S	
034	*LBLE	Check for 2, 3, 4th move of player and decide HP's move.
035	GSB6	
036	GSB8	
037	GSBd	
038	RCLI	
039	GSBB	
040	X=Y?	
041	GSBB	
042	STOI	
043	GSBD	
044	GSBe	
045	RCLI	
046	DSP9	
047	RCLE	
048	+	
049	R/S	
050	GTOE	
051	*LBLB	Check for next move.
052	RCLB	
053	FRC	
054	1	
055	0	
056	x	
057	STOB	
058	INT	
059	RTN	
060	*LBL6	Store to R_A & R_I.
061	STOA	
062	STOI	
063	RTN	
064	*LBL8	Increase move number.
065	1	
066	ST+0	
067	RCLA	
068	RTN	
069	*LBLd	Store 2 for HP.
070	RCLI	
071	CHS	
072	10^x	
073	2	
074	x	
075	RCLE	
076	+	
077	STOE	
078	RCLI	
079	RTN	
080	*LBLe	Decode for printing.
081	EEX	
082	6	
083	GSB9	
084	EEX	
085	3	
086	GSB9	
087	1	
088	GSB9	
089	SPC	
090	RTN	
091	*LBL9	Print.
092	DSP3	
093	RCLE	
094	x	
095	FRC	
096	RCL0	
097	+	
098	PRTX	
099	DSP0	
100	RTN	
101	*LBLD	Store 1 for player.
102	RCLI	
103	CHS	
104	10^x	
105	RCLE	
106	+	
107	STOE	
108	RCLI	
109	RTN	
110	*LBLa	Store constants for move sequences.
111	.	
112	5	

REGISTERS

0	1	2	3	4	5	6	7	8	9
play no.	.5873649		.5891467	.13598	.1374698	.31578	.13589	.3175964	.31587
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
Temp.	Last X of LBL B			Used	1, 2, ..., 9

113	8		169	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	RTN	
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

A New Game	B next move	C	D	E 2nd move
a	b St to A, I	c	d 2 for HP	e Decode
0 1st move	1	2	3	4
5	6	7	8 Increase	9 Print

FLAGS

0	
1	
2	
3	

SET STATUS

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

Wari

001	*LBLA	Move player 1
002	CHS	
003	7	
004	+	
005	SF1	
006	GTO9	
007	*LBLE	Move player 2
008	6	
009	+	
010	CF1	
011	*LBL9	Initialize move loop
012	STOC	
013	STOI	
014	RCLi	
015	X=0?	
016	÷	
017	STOD	
018	0	
019	STOi	
020	*LBL1	Begin loop 1
021	DSZI	I←I – 1
022	GTO8	If I = 0
023	1	Then I←12
024	2	
025	STOI	
026	*LBL8	
027	RCLI	
028	RCLC	
029	X=Y?	If I = original bin
030	GTO1	then skip to next bin
031	1	else
032	ST+i	increment bin
033	RCLD	decrement # seeds
034	1	
035	-	
036	STOD	
037	X=0?	If seeds used up
038	GTO5	then try to capture
039	GTO1	else repeat loop 1
040	*LBLC	Initialize
041	1	
042	2	
043	STOI	
044	4	
045	*LBL0	
046	STOi	
047	DSZI	
048	GTO8	
049	0	
050	STOA	
051	STOB	
052	*LBL6	Display board
053	GSBB	
054	F0?	
055	GTOD	Display score
056	F1?	
057	GTOe	
058	GTOD	
059	*LBLB	Display board
060	SPC	
061	6	Initialize
062	STOI	I←6
063	*LBL3	Begin loop 3
064	RCLi	
065	1	
066	3	
067	RCLI	
068	-	
069	STOI	
070	CLX	
071	+	
072	RCLi	
073	EEX	
074	2	
075	÷	$D = C(I) + \frac{C(13 - I)}{100}$
076	+	
077	PRTX	Print D
078	RCLI	
079	1	
080	3	
081	-	
082	CHS	
083	STOI	
084	DSZI	I←I – 1
085	GTO3	If I > 0 then repeat loop 3
086	SPC	
087	RTN	
088	*LBLD	Display score
089	RCLA	
090	RCLB	
091	EEX	
092	2	
093	÷	
094	+	
095	RTN	
096	*LBL5	CAPTURE
097	RCLI	
098	6	
099	÷	
100	5	
101	-	
102	RCLC	
103	LSTX	
104	-	
105	x	
106	X>0?	If start & finish on same side
107	GTO6	Then no capture
108	*LBL7	Else begin loop 7
109	2	
110	RCLi	
111	X=Y?	If bin has 2 or 3
112	GTO7	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 Bin 10	S1 Bin 11	S2 Bin 12	S3	S4	S5	S6	S7	S8	S9

A Score 1	B Score 2	C Current bin	D # seeds	E Random	I Pointer & Counter

113	3	
114	X≠Y?	priate score,
115	GTO6	
116	*LBL7	Else no more capture.
117	F1?	— — — — — — — —
118	GTO9	Begin loop 7
119	RCLB	If player 1
120	+	Then go to 9
121	STOB	Else
122	GTO7	increment score 2
123	*LBL9	— — — — — — — —
124	RCLA	Increment score 1
125	+	
126	STOA	
127	*LBL7	— — — — — — — —
128	0	Remove seeds from bin
129	STOI	
130	ISZI	Step to next bin
131	7	If this is bin 7 or 13
132	RCLI	Then done
133	X=Y?	Else
134	GTO6	Try to capture from this bin
135	1	
136	3	
137	X=Y?	
138	GTO6	
139	GTO7	— — — — — — — — — — — —
140	*LBLe	Automatic player 2
141	1	
142	STOI	I←1
143	*LBL2	Begin loop 2
144	RCLI	
145	X=0?	If C(I) = 0
146	GTOb	Then next I
147	3	Else if C(I) = 1 or 2
148	X>Y?	Then
149	GTO4	Search for attacking bin
150	*LBLb	Else next I
151	ISZI	I←I + 1
152	RCLI	
153	7	
154	X≠Y?	While I < 7
155	GTO2	Repeat loop 2
156	*LBLc	— — — — — — — —
157	RCLE	Move randomly
158	Pi	
159	+	
160	8	
161	Y^x	
162	FRC	
163	STOE	
164	6	
165	x	
166	1	
167	+	
168	+P	
169	INT	
170	STO0	
171	6	
172	+	
173	STOI	
174	RCLI	
175	X=0?	If bin is empty
176	GTOc	Then try again
177	RCL0	Else
178	PRTX	make move
179	GSBE	
180	RTN	— — — — — — — — — —
181	*LBL4	Search for attacking bin
182	RCLI	
183	STO6	Save pointer from loop 2
184	1	
185	2	
186	STOI	Initialize new pointer
187	*LBLd	
188	RCLI	
189	X=0?	
190	GTO8	
191	1	
192	1	
193	÷	
194	FRC	
195	1	
196	1	
197	x	
198	RCLI	
199	RCL6	
200	-	
201	X=Y?	If capture is possible
202	GTO9	Then go to 9
203	*LBL8	
204	DSZI	Decrement pointer
205	RCLI	
206	6	
207	X=Y?	If search is complete
208	GTO8	Then go to 8
209	GTOd	
210	*LBL8	
211	RCL0	Restore pointer for
212	STOI	loop 2
213	GTO4	
214	*LBL9	Capture possible
215	RCLI	
216	6	
217	-	
218	PRTX	
219	GSBE	
220	RTN	

LABELS

A Player 1	B Display	C Start	D Score	E Player 2
a	b Random move	c Random loop	d Loop	e Auto player 2
0 Loop	1 Loop	2 Loop	3 Loop	4 Used
5 Used	6 Used	7 Used	8 Used	9 Used

FLAGS

0	P1 vs. P2
1	Player 1
2	
3	

SET STATUS

FLAGS	ON	OFF
0	☒	☐
1	☐	☒
2	☐	☒
3	☐	☒

TRIG	
DEG	☒
GRAD	☐
RAD	☐

DISP	
FIX	☒
SCI	☐
ENG	☐
n	2

Racetrack

001	*LBLC	START
002	CLRG	
003	P≠S	
004	CLRG	
005	1	
006	7	
007	STOI	Initialize loop 9
008	*LBL9	
009	RCLI	
010	1	
011	-	
012	4	Compute starting position
013	÷	
014	5	
015	×	
016	5	
017	5	
018	+	
019	STOi	
020	RCLI	Decrement pointer
021	4	
022	-	
023	STOI	
024	X>0?	While pointer is positive,
025	GTO9	repeat loop 9
026	RTN	
027	*LBLA	Identify cars
028	1	
029	GTO0	
030	*LBLB	
031	2	
032	GTO0	
033	*LBLC	
034	3	
035	GTO0	
036	*LBLD	
037	4	
038	GTO0	
039	*LBLE	
040	5	
041	*LBL0	
042	PRTX	Print car ID
043	1	Compute a pointer
044	-	
045	4	
046	×	
047	1	
048	+	
049	STO0	
050	STOI	
051	R↓	
052	9	
053	X≤Y?	If |v| exceeds 9
054	X⇄Y	Then use 9 instead
055	R↓	
056	→R	
057	ISZI	
058	ISZI	
059	RCLi	
060	+	
061	STOi	Increment v_x
062	GSB2	
063	DSZI	
064	DSZI	
065	RCLi	
066	+	
067	STOi	Increment x
068	X⇄Y	
069	ISZI	
070	ISZI	
071	ISZI	
072	RCLi	
073	+	
074	STOi	Increment v_y
075	GSB2	
076	DSZI	
077	DSZI	
078	RCLi	
079	+	
080	STOi	Increment y
081	RCL0	
082	STOI	
083	RCLi	Initialize loop 8
084	STO8	
085	ISZI	
086	RCLi	
087	STOC	
088	1	
089	7	
090	STOI	
091	*LBL8	Test for crash
092	RCL6	
093	X=Y?	If same car
094	GTO0	Then skip test
095	RCLi	
096	RCL0	
097	-	
098	ISZI	
099	RCLi	
100	RCLC	
101	-	
102	→P	
103	2	
104	X≤Y?	
105	DSZI	
106	X≤Y?	If distance ≥ 2
107	GTO0	Then no crash
108	RCLI	
109	2	
110	-	
111	4	
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 Used	C Used	D	E	I Used

113	1	
114	+	
115	PSE	Display ID of car in crash
116	PSE	
117	PSE	
118	0	Set v_y and v_x to zero for both cars
119	ISZI	
120	STOi	
121	ISZI	
122	STOi	
123	RCL6	Save pointer
124	X⇄I	
125	X⇄Y	
126	ISZI	
127	ISZI	
128	STOi	
129	ISZI	
130	STOi	
131	X⇄Y	
132	3	
133	-	
134	X⇄I	Restore pointer
135	*LBL0	
136	RCLI	Decrement pointer
137	4	
138	-	
139	STOI	
140	X>0?	WHILE pointer is positive
141	GTO8	repeat loop 8
142	RCLB	
143	5	CHECK INSIDE FENCE
144	0	
145	÷	
146	GSB7	
147	RCLC	
148	7	
149	0	
150	÷	
151	GSB7	
152	+	
153	1	
154	X>Y?	If off track
155	GSB5	Then penalize
156	RCLB	
157	8	CHECK OUTSIDE FENCE
158	0	
159	÷	
160	GSB7	
161	RCLC	
162	EEX	
163	2	
164	÷	
165	GSB7	
166	+	
167	1	
168	X>Y?	If on track
169	GTO6	THEN don't penalize
170	*LBL5	PENALIZE
171	RCL0	
172	2	
173	+	
174	STOI	
175	0	Set v_y and v_x to zero
176	STOi	
177	ISZI	
178	STOi	
179	*LBL6	Set up display
180	RCL0	
181	3	
182	+	
183	STOI	
184	RCLi	
185	DSZI	
186	RCLi	
187	DSZI	
188	→P	
189	RCLi	
190	DSZI	
191	RCLi	
192	PRST	
193	RTN	
194	*LBL2	Subroutine to compute ΔS
195	LSTX	
196	+	
197	2	
198	÷	
199	RTN	
200	*LBL7	Subroutine to compute $y^{2.5}$
201	ABS	
202	2	
203	.	
204	5	
205	y^x	
206	RTN	
207	*LBLa	Display position ID→y,x
208	PRTX	
209	1	
210	-	
211	4	
212	x	
213	2	
214	+	
215	STOI	
216	RCLi	
217	PRTX	
218	DSZI	
219	RCLi	
220	PRTX	
221	RTN	

LABELS

A 1	B 2	C 3	D 4	E 5
a ID→y,x	b	c START	d	e
0 Used	1	2 Subroutine	3	4
5 Off track	6 Display	7 Subroutine	8 Crash loop	9 Start loop

FLAGS

0	
1	
2	
3	

SET STATUS

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

Teaser

001	*LBLC	Initialize
002	1	
003	STO5	
004	CHS	
005	STO1	
006	STO2	
007	STO3	
008	STO4	
009	STO6	
010	STO7	
011	STO8	
012	STO9	
013	STOB	
014	1	
015	0	
016	STOA	
017	GTOD	
018	*LBLA	Store move
019	STOI	
020	RCLI	Generate a negative 1 or display "Error"
021	X<0?	
022	GTOe	
023	CHS	
024	GTOi	Go to appropriate routine to change board
025	*LBL1	
026	STx1	
027	STx2	
028	STx4	
029	STx5	
030	GTOD	
031	*LBL2	
032	STx1	
033	STx2	
034	STx3	
035	GTOD	
036	*LBL3	
037	STx2	
038	STx3	
039	STx5	
040	STx6	
041	GTOD	
042	*LBL4	
043	STx1	
044	STx4	
045	STx7	
046	GTOD	
047	*LBL5	
048	STx2	
049	STx5	
050	STx8	
051	STx4	
052	STx6	
053	GTOD	
054	*LBL6	
055	STx3	
056	STx6	
057	STx9	
058	GTOD	
059	*LBL7	
060	STx4	
061	STx5	
062	STx7	
063	STx8	
064	GTOD	
065	*LBL8	
066	STx7	
067	STx8	
068	STx9	
069	GTOD	
070	*LBL9	
071	STx5	
072	STx6	
073	STx8	
074	STx9	
075	*LBLD	Set up display
076	F0?	
077	SPC	
078	FIX	
079	DSP3	
080	RCL9	
081	X<0?	
082	0	
083	RCLA	
084	÷	
085	RCL8	
086	X<0?	
087	CLX	
088	+	
089	RCLA	
090	÷	
091	RCL7	
092	X<0?	
093	CLX	
094	+	
095	RCLA	
096	÷	
097	F0?	
098	PRTX	
099	EEX	
100	6	
101	÷	
102	STO0	
103	RCL6	
104	X<0?	
105	0	
106	RCLA	
107	÷	
108	RCL5	
109	X<0?	
110	CLX	
111	+	
112	RCLA	

REGISTERS

0	1	2	3	4	5	6	7	8	9
Used	±1	±1	±1	±1	±1	±1	±1	±1	±1
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
10	# of moves				

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

If flag 0
then clear flag
else set flag and
display 1

Clear flag
display 0

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

Golf

001	*LBLb	
002	RCLA	Random #.
003	9	u_{i+1}
004	9	
005	7	
006	×	
007	FRC	
008	STOA	u_i
009	.	
010	5	
011	-	$u_i - .5$
012	RTN	----------
013	*LBLA	Woods distance.
014	GSBb	
015	1	
016	+	
017	4	
018	0	
019	×	W
020	STOC	----------
021	0	Clear registers.
022	STO5	
023	STO3	
024	STO2	
025	P⇄S	
026	STO5	
027	STO3	
028	STO2	
029	P⇄S	----------
030	GSBb	Hole yardage.
031	9	
032	×	
033	7	
034	+	
035	5	
036	0	
037	×	Y_h
038	STOD	----------
039	ST-2	Yo_1
040	P⇄S	
041	ST-2	Yo_2
042	P⇄S	----------
043	2	Par.
044	1	
045	5	
046	÷	
047	3	
048	+	
049	INT	Par.
050	ST+9	Σ Par.
051	P⇄S	
052	ST+9	Σ Par.
053	1	----------
054	ST+0	Hole number. h
055	P⇄S	
056	ST+0	h

057	RCL6	Print.
058	RCLD	
059	R↑	
060	RCLC	
061	PRST	Print h, Y_h, Par, W.
062	RTN	----------
063	*LBLD	Club distance.
064	1	Putter
065	9	
066	GTOC	
067	*LBLE	Wood
068	4	
069	-	
070	2	
071	×	
072	*LBLC	Iron
073	1	
074	ST+1	S′ = total strokes.
075	R↓	
076	2	
077	0	
078	-	
079	1	
080	0	
081	×	
082	CHS	
083	STO6	Mean distance m.
084	0	
085	R/S	Enter s.
086	STOB	s
087	2	Calculate Y_c
088	1/X	
089	GSBb	
090	-	
091	LSTX	.5
092	X⇄Y	
093	RCL7	H
094	5	
095	6	
096	÷	
097	×	
098	-	
099	RCL8	m
100	5	
101	÷	
102	×	
103	RCL8	m
104	RCLB	s
105	×	
106	+	
107	ABS	Y_c
108	STOE	----------
109	GSBb	Calculate θ_c.
110	3	
111	×	
112	RCL7	H

REGISTERS

0 h	1 S_1'	2 Y_1	3 x_1	4 D_{h1}	5 ϕ_{h1}	6	7 H_1	8 m	9 Σ Par
S0 h	S1 S_2'	S2 Y_2	S3 x_2	S4 D_{h1}	S5 ϕ_{h2}	S6	S7 H_2	S8	S9 Σ Par

A u_i	B s, θ_c	C W	D Y_h	E Y_c	I

113	3	
114	6	
115	+	
116	/X	
117	x	
118	STOB	θ_c
119	RCL3	Is ball holed?
120	ABS	
121	RCLC	W
122	X≤Y?	$W \leq \lvert x_{n-1} \rvert$?
123	GSB9	Yes—in woods.
124	RCL4	No—escape woods.
125	4	
126	+	
127	RCLE	Y_c
128	X>Y?	$Y_c > D_h + 4$?
129	GTO4	Yes—missed hole.
130	RCL4	No—in hole?
131	5	
132	1/X	
133	-	
134	X>Y?	$D_h - .2 > Y_c$?
135	GTO4	Yes—missed hole.
136	LSTX	No—in hole?
137	RCL4	D_h
138	÷	
139	TAN⁻¹	γ: hole window
140	RCLB	θ_c
141	ABS	
142	X>Y?	$\lvert\theta_c\rvert > \gamma$?
143	GTO4	Yes—missed hole.
144	8	No—in hole!
145	DSP9	
146	PRTX	Print 0.000000000.
147	RCL1	S′ = total strokes.
148	RCL7	H
149	1	
150	8	
151	÷	H/18
152	RCL0	h
153	x	h (H/18)
154	-	S′ – h (H/18)
155	RCL9	Σ Par
156	-	[S′ –h(H/18)] – Σ Par
157	DSP0	
158	PRTX	Print S.
159	RTN	
160	*LBL4	Missed hole.
161	RCLB	θ_{cn}
162	RCL5	$\phi_{h\ n-1}$
163	+	
164	RCLE	Y_{cn}
165	→R	$Y_c, \theta + \phi \rightarrow y', x'$
166	RCL2	y_{n-1}
167	+	
168	STO2	y_n
169	X⇄Y	x'
170	RCL3	x_{n-1}
171	+	
172	STO3	x_n
173	CHS	
174	X⇄Y	
175	CHS	
176	→P	$-y_n, -x_n \rightarrow D_{h_n}, \phi_n$
177	STO4	D_{h_n}
178	X⇄Y	
179	STO5	ϕ_{h_n}
180	RCLE	Y_c
181	RCLB	θ_c
182	RCL2	y_n
183	RCL3	x_n
184	PRST	Print Y_c, θ_c, y, x
185	ABS	In woods?
186	RCLC	W
187	X≤Y?	$W \leq \lvert x \rvert$?
188	GTO6	Yes: In woods
189	GTO7	No: No woods
190	*LBL0	Woods display
191	DSP6	
192	*LBL7	Print D_h, ϕ_h
193	RCL4	D_h
194	PRTX	Print D_h
195	RCL5	ϕ_h
196	PRTX	Print ϕ_h
197	DSP0	
198	RTN	
199	*LBL9	In woods.
200	RCLB	θ_c
201	ABS	
202	RCL5	ϕ_h
203	ABS	
204	SIN	sin $\lvert\phi\rvert$
205	3	
206	x	3 sin $\lvert\phi\rvert$
207	RCL3	x
208	ABS	
209	RCLC	W
210	-	$\lvert x \rvert - W$
211	÷	3 sin $\lvert\phi\rvert/(W - \lvert x \rvert)$
212	TAN⁻¹	β = woods window
213	ABS	
214	X>Y?	$\lvert\beta\rvert > \lvert\theta\rvert$?
215	RTN	Yes—escape woods
216	RCLE	No—hit trees
217	9	
218	÷	
219	STOE	$Y_c = Y_c'/9$
220	RCLB	θ'
221	9	
222	x	
223	STOB	$\theta = 9(\theta')$
224	RTN	

LABELS

A HOLE	B WOOD	C IRON	D PUTTER	E
a	b RAN, #	c	d	e
0	1	2	3	4 No hole
5	6	7 Prt D_h, ϕ_h	8 Wood disp.	9 In woods

FLAGS

0	
1	
2	
3	

SET STATUS

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

The Dealer

001	*LBLC	Deal n cards.
002	SPC	
003	*LBLE	
004	STOA	
005	GSBA	
006	PRTX	
007	RCLA	
008	1	
009	-	
010	X>0?	
011	GTO6	
012	RTN	
013	*LBLA	Generate pseudo-random number.
014	RCLB	
015	9	
016	9	
017	7	
018	×	
019	FRC	
020	STOB	
021	RCLC	Scale pseudo-random number according to number of cards.
022	×	
023	INT	
024	5	Calculate register containing selected card.
025	÷	
026	STOI	
027	FRC	Compute multplier necessary to move selected card up to decimal point.
028	1	
029	0	
030	×	
031	10ˣ	
032	STOD	
033	RCLi	Remove digits to left of decimal.
034	×	
035	FRC	
036	EEX	Select card and store card number in R_E.
037	2	
038	×	
039	INT	
040	STOE	
041	RCLI	Take card from bottom of deck.
042	RCLC	
043	5	
044	÷	
045	STOI	
046	FRC	
047	1	
048	0	
049	×	
050	10ˣ	
051	1/X	
052	LSTX	
053	RCLi	
054	×	
055	FRC	
056	EEX	
057	2	
058	×	
059	INT	
060	RCLE	Place selected card at bottom of deck.
061	-	
062	EEX	
063	2	
064	÷	
065	×	
066	ST-i	
067	R↑	Place card from bottom of deck in place of selected card.
068	STOI	
069	LSTX	
070	RCLD	
071	÷	
072	ST+i	
073	RCLC	Move bottom of deck up one card.
074	1	
075	-	
076	STOC	
077	X<0?	If all cards have been dealt, shuffle.
078	GSBc	
079	RCLE	If bingo flag is set, go to zero.
080	*LBL5	
081	F1?	
082	GTO0	
083	1	Calculate suit.
084	3	
085	÷	
086	1	
087	+	
088	INT	
089	LSTX	
090	FRC	Calculate card.
091	1	
092	3	
093	×	
094	1	
095	.	
096	5	
097	+	
098	INT	
099	10ˣ	
100	×	Combine for output.
101	SCI	
102	RTN	
103	*LBL0	Bingo output.
104	1	
105	+	
106	FIX	
107	RTN	
108	*LBLc	Shuffle: Store 51 as number in deck for cards or 74 for bingo.
109	2	
110	3	
111	ENT↑	
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	I Hole

113	1	
114	F1?	
115	+	
116	STOC	
117	CLX	
118	RTN	
119	*LBLe	Set bingo flag.
120	SF1	
121	GTO0	
122	*LBLa	Clear bingo flag.
123	CF1	
124	*LBL0	
125	1	Load cards as two digit numbers; five per register.
126	8	
127	STOI	
128	.	
129	0	
130	5	
131	0	
132	5	
133	0	
134	5	
135	0	
136	5	
137	0	
138	5	
139	CHS	
140	ENT↑	
141	ENT↑	
142	ENT↑	
143	.	
144	9	
145	5	
146	9	
147	6	
148	9	
149	7	
150	9	
151	8	
152	9	
153	9	
154	STO9	
155	P⇄S	
156	*LBL7	
157	+	
158	STOi	
159	DSZI	
160	GTO7	
161	+	
162	STO0	
163	.	
164	5	Store seed.
165	2	
166	8	
167	4	
168	1	
169	6	
170	3	
171	STOB	
172	GSBc	Store number of cards.
173	RTN	
174	STOB	Store user seed.
175	RTN	
176	*LBLE	Set registers for review of cards dealt.
177	SPC	
178	RCLC	
179	GSBc	
180	R↓	
181	RCLC	
182	STOD	
183	X⇄Y	
184	STOC	
185	*LBL4	Pull values out of deck in order previously dealt for review.
186	RCLD	
187	5	
188	÷	
189	STOI	
190	FRC	
191	1	
192	0	
193	×	
194	10^x	
195	RCLi	
196	×	
197	FRC	
198	EEX	
199	2	
200	×	
201	INT	
202	GSB5	
203	PRTX	Is review complete?
204	RCLC	
205	RCLD	
206	1	
207	-	
208	STOD	
209	X≠Y?	
210	GTO4	Display last value from review.
211	R↓	
212	R↓	
213	RTN	

LABELS

A →Hit	B	C n→Deal	D	E Review
a Cards; seed	b	c Shuffle	d	e Bingo; seed
0 Used	1	2	3	4 Review
5 Suit/Card	6 Deal	7 Start	8	9

FLAGS

0	
1	Bingo
2	
3	

SET STATUS

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

Bowling Scorekeeper

001	*LBLa	Start
002	CLRG	Clear primary registers.
003	P=S	
004	9	
005	STOI	
006	EEX	
007	4	
008	CHS	
009	*LBL9	
010	STOi	Store .0001 in secondary
011	DSZI	registers 0 to 9.
012	GTO9	
013	STOi	
014	CLX	
015	RTN	
016	*LBLA	Entry for other than spare
017	INT	or strike.
018	LSTX	
019	FRC	
020	ENT↑	
021	ENT↑	
022	1	
023	0	
024	×	
025	FRC	
026	X≠0?	Error if extra digits in entry
027	GTOB	e.g. 1.08.
028	R↓	
029	1	
030	0	
031	÷	
032	+	Original entry
033	STOI	Store and decrement for
034	DSZI	indirect address.
035	CF0	NOP
036	FRC	
037	EEX	
038	2	
039	×	Pin count
040	P=S	
041	RCLi	Recall secondary register
042	P=S	(CS.S1S2 PS).
043	ENT↑	
044	INT	CS = current sum
045	ENT↑	
046	R↑	
047	+	New CS
048	1	
049	1	
050	X>Y?	Less than 11?
051	GTO0	Continue
052	*LBLB	else error
053	ISZI	
054	RCLI	
055	GSBD	
056	GTOb	
057	*LBLB	Spare
058	*LBLC	Strike
059	INT	Remove extra digits.
060	.	
061	1	
062	+	
063	STOI	Store and decrement for
064	DSZI	indirect address.
065	CF0	NOP
066	P=S	
067	RCLi	Recall secondary register.
068	P=S	
069	ENT↑	
070	INT	
071	1	
072	0	10 pins
073	ENT↑	
074	*LBL0	
075	R↓	Pins down
076	STOA	New CS to A
077	R↓	
078	GSB1	Shift, get S1
079	STOB	S1 to B
080	GSB1	Shift, get S2
081	STOC	S2 to C
082	GSB1	Shift, get PS
083	STOD	PS to D
084	RCLi	Recall primary register.
085	CF1	Clear 2nd ball flag.
086	X<0?	If reg. neg.
087	SF1	set 2nd ball flag.
088	ABS	Remove sign.
089	EEX	
090	2	
091	×	
092	INT	
093	EEX	
094	2	
095	÷	
096	STOE	SSS.FS to E
097	RCLC	
098	X≠0?	If strike 2 balls back
099	GSB2	GSB2.
100	RCLB	Strike 1 ball back
101	STOC	now 2 balls back.
102	0	No strike 1 ball back.
103	STOB	
104	F1?	If 2nd ball
105	GTO3	GTO 3
106	RCLD	Add count
107	X≠0?	
108	GSB2	
109	0	
110	STOD	0 to PS
111	RCLA	
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	I ID.P

113	0	
114	X≠Y?	Strike?
115	GTO4	No strike
116	STO6	Store strike
117	GTO5	- - - - - - - - - - - -
118	*LBL4	2nd ball
119	SF1	
120	GTO6	- - - - - - - - - - - -
121	*LBL3	Spare?
122	CF1	
123	RCLA	
124	1	
125	0	
126	X≠Y?	CS ≠ 10?
127	GTO7	
128	STO0	
129	GTO5	
130	*LBL7	Add count
131	GSB2	
132	*LBL5	
133	0	
134	STOA	0 to CS
135	RCLi	Increment frame count.
136	ABS	
137	EEX	
138	4	
139	CHS	
140	+	
141	STOi	
142	*LBL6	- - - - - - - - - - - -
143	RCLD	Ready score and frame
144	EEX	display, PS.
145	2	
146	÷	
147	RCLC	S2
148	+	
149	EEX	
150	2	
151	÷	
152	RCLB	S1
153	+	
154	EEX	
155	2	
156	÷	
157	RCLA	CS
158	+	CS.S1S2PS
159	P⇄S	
160	STOi	Update sec. reg.
161	P⇄S	
162	RCLi	
163	ABS	Remove sign.
164	EEX	
165	2	
166	x	
167	FRC	.NF
168	EEX	
169	2	
170	x	
171	INT	NF.
172	RCLI	
173	FRC	Last ball
174	+	NF.LB
175	EEX	
176	4	
177	÷	.00NFLB
178	RCLE	
179	+	SSS.FSNFLB
180	F1?	If flag set.
181	CHS	Make negative.
182	STOi	Update print register.
183	PRTX	Print display.
184	RTN	- - - - - - - - - - - -
185	*LBL1	Subroutine to shift and
186	-	take integer part.
187	EEX	
188	2	
189	x	
190	ENT↑	
191	INT	
192	RTN	- - - - - - - - - - - -
193	*LBL2	Scoring subroutine
194	RCLB	S1
195	RCLC	S2
196	+	
197	RCLD	PS
198	+	
199	RCLA	CS
200	+	
201	RCLE	SSS.FS
202	ABS	
203	+	
204	EEX	
205	2	
206	CHS	.01
207	+	Increment FS
208	STOE	Update E
209	FRC	Compare FS to .1
210	-	
211	1	
212	X>Y?	If less than .1
213	RTN	Continue
214	RCLE	Otherwise
215	ABS	
216	STOi	Update print register.
217	PRTX	Print and stop.
218	R/S	- - - - - - - - - - - -
219	*LBLD	Recall score display.
220	1	
221	-	
222	STOI	
223	RCLi	
224	RTN	

LABELS

A ID.P entry	B Spare	C Strike	D Recall score	E
a Start	b Error	c	d	e
0 Used	1 Shift sub.	2 Score sub.	3 Spare	4 2nd
5 Add frame	6 Display	7 Add count	8 Error	9 Initialize

FLAGS

0	
1	2nd ball
2	
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	4

Biorhythms

001	*LBL2	Increment pointer by 3.
002	ISZI	
003	ISZI	
004	ISZI	
005	RTN	
006	*LBLA	
007	STOA	Store Birthdate
008	RTN	
009	*LBLc	FUNCTION OF DATE
010	ENT↑	
011	INT	
012	STO3	N (M, D, Y)
013	-	
014	EEX	
015	2	
016	x	
017	ENT↑	
018	INT	
019	STO5	
020	-	
021	EEX	
022	4	
023	x	
024	STO6	
025	3	
026	6	
027	5	
028	STO4	
029	x	
030	2	
031	RCL3	
032	X>Y?	
033	GTO8	
034	x	
035	CLX	
036	RCL6	
037	1	
038	-	
039	STO6	
040	GTO1	
041	*LBL0	
042	:	
043	4	
044	x	
045	.	
046	3	
047	+	
048	+	
049	INT	
050	-	
051	RCL6	
052	*LBL1	
053	CLX	
054	RCL5	
055	+	
056	RCL3	
057	1	
058	-	
059	3	
060	1	
061	x	
062	+	
063	RCL6	
064	4	
065	÷	
066	INT	
067	X≷Y	
068	+	
069	RTN	
070	*LBLC	+1 loop
071	RCLC	
072	STO9	
073	*LBL7	
074	1	
075	ST+9	
076	GSBb	
077	GTO7	
078	*LBLB	BIO DATE CYCLE
079	STOB	Store Bio date
080	RCLA	
081	GSBc	Days between dates
082	STO2	
083	LSTX	
084	STO0	
085	RCLB	
086	GSBc	
087	LSTX	
088	ST-0	
089	CLX	
090	RCL2	
091	-	
092	RCL4	
093	2	
094	÷	
095	ST÷0	
096	X≷Y	
097	STOC	
098	STO9	
099	2	Store cycles bases.
100	3	
101	STO3	
102	2	
103	8	
104	STO2	
105	3	
106	3	
107	STO1	
108	*LBL6	PRINT CYCLE
109	3	for a day
110	STOI	
111	DSP2	
112	*LBL6	

REGISTERS

0	1	2	3	4	5	6	7	8	9
1 or 2	33	28	23	Days	Days	Days	Used		Δ days
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
Birth date	Bio date	Δ days			

113	RCL9	
114	RCLi	
115	÷	
116	FRC	
117	F1?	
118	GTO8	
119	2	
120	×	
121	PI	
122	×	
123	SIN	
124	RND	
125	GTO1	
126	*LBL8	
127	RCLi	
128	×	
129	GSB2	
130	STOi	
131	X≠I	
132	3	
133	-	
134	X≠I	
135	DSZI	
136	GTO6	
137	RTN	
138	*LBL1	
139	GSB9	
140	*LBL8	
141	DSZI	
142	GTO6	
143	SPC	
144	RTN	
145	*LBL9	----------
146	F0?	PRINT/PAUSE
147	PRTX	
148	F0?	
149	RTN	
150	R/S	
151	RTN	
152	*LBLe	----------
153	DSP0	PRINT TOGLE
154	F0?	
155	GTO8	
156	1	
157	SF0	
158	RTN	
159	*LBL0	
160	6	
161	CF0	
162	RTN	
163	*LBLd	----------
164	.4	DAYS UNTIL NEXT MAX
165	STO7	
166	1	
167	RCLC	
168	GTO0	
169	*LBLD	DAYS UNTIL NEXT MIN
170	4	
171	ENT↑	
172	3	
173	÷	
174	STO7	
175	1	
176	RCLC	
177	GTO6	----------
178	*LBLE	DAYS UNTIL NEXT CRITICAL
179	2	
180	STO7	
181	RCLC	----------
182	*LBL0	
183	STO9	
184	X≠Y	
185	STO0	
186	SF1	
187	GSB6	
188	CF1	
189	3	
190	STOI	
191	*LBL3	
192	GSB2	
193	RCLi	
194	X≠I	
195	3	
196	-	
197	X≠I	
198	RCLi	
199	RCL7	
200	÷	
201	-	
202	CHS	
203	X<0?	
204	GTO3	
205	*LBL4	
206	PRTX	
207	*LBL3	
208	RCLi	
209	RCL0	
210	÷	
211	+	
212	RCL1	
213	X≠Y	
214	X>Y?	
215	GTO5	
216	GTO4	
217	*LBL5	
218	SPC	
219	DSZI	
220	GTO3	
221	RTN	

LABELS

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

FLAGS

0	
1	
2	
3	

SET STATUS

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

Timer

001	*LBLA	INITIALIZATION
002	CLRG	
003	P⇄S	
004	CLRG	
005	5	C_b (MINIMUM INTERVAL CALIBRATION CONSTANT)
006	1	
007	3	
008	4	
009	1/X	
010	STOE	
011	5	
012	.	C_a (5 SECOND PRIMARY)
013	7	
014	8	
015	3	
016	0	
017	STOB	
018	.	K_a (5 SECOND SECONDARY)
019	1	
020	6	
021	7	
022	5	
023	STOC	
024	6	C_c (COUNT-UP)
025	0	
026	6	
027	0	
028	1/X	
029	STOD	
030	6	C_d (COUNT-DOWN)
031	7	
032	9	
033	0	
034	1/X	
035	STO5	
036	0	
037	RTN	
038	*LBL5	5 SECOND TIMER LOOP
039	PSE	
040	HMS+	
041	7	
042	2	
043	0	
044	1/X	0:00:05 HRS
045	+	
046	STOA	TIME
047	RCLB	C_a
048	RCLI	
049	+	
050	STOI	
051	*LBL1	FAST LOOP
052	DSZI	
053	GTO1	
054	RCLA	TIME
055	→HMS	
056	GTO5	

057	*LBLa	CALIBRATE 5 SECOND TIMER
058	R↓	
059	GSB4	
060	R↑	
061	HMS+	
062	X⇄Y	
063	HMS+	
064	−	
065	LSTX	
066	RCLC	K_a
067	×	
068	÷	
069	RCLB	C_a
070	+	
071	PRTX	
072	RTN	
073	*LBLC	COUNT-UP TIMER
074	GSBA	
075	R/S	
076	*LBL7	FAST LOOP
077	ISZI	
078	GTO7	
079	*LBLc	CONVERT TO TIME
080	RCLI	
081	RCLD	
082	×	
083	→HMS	
084	RTN	
085	P⇄S	SPLITS
086	RCL1	
087	STOI	
088	RCL2	
089	STO1	
090	RCL3	
091	STO2	
092	RCL4	
093	STO3	
094	RCL5	
095	STO4	
096	RCL6	
097	STO5	
098	RCL7	
099	STO6	
100	RCL8	
101	STO7	
102	RCL9	
103	STO8	
104	RCL0	
105	STO9	
106	P⇄S	
107	RCL1	
108	P⇄S	
109	STO0	
110	P⇄S	
111	RCL2	
112	STO1	

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	I

113	RCL3	
114	STO2	
115	RCL4	
116	STO3	
117	RCL5	
118	STO4	
119	RCL6	
120	STO5	
121	GTOc	
122	*LBLE	
123	RCL5	
124	STO6	
125	RCL4	
126	STO5	
127	RCL3	
128	STO4	
129	RCL2	
130	STO3	
131	RCL1	
132	STO2	
133	P⇄S	
134	RCL0	
135	P⇄S	
136	STO1	
137	P⇄S	
138	RCL9	
139	STO0	
140	RCL8	
141	STO9	
142	RCL7	
143	STO8	
144	RCL6	
145	STO7	
146	RCL5	
147	STO6	
148	RCL4	
149	STO5	
150	RCL3	
151	STO4	
152	RCL2	
153	STO3	
154	RCL1	
155	STO2	
156	RCLI	
157	STO1	
158	4	
159	:	C_e SPLIT CONSTANT
160	4	
161	2	
162	+	
163	STOI	
164	P⇄S	
165	GTO7	
166	*LBLB	MINIMUM INTERVAL VISIBLE TIMER
167	GSBA	
168	R/S	
169	*LBL6	FAST LOOP
170	PSE	
171	RCLE	
172	HMS+	
173	GTO6	
174	*LBLb	CALIBRATE MIN. INTERVAL TIMER
175	R↓	
176	GSB4	
177	X⇄Y	
178	R↓	
179	RCLE	
180	GTO2	
181	PRTX	
182	RTN	
183	*LBL2	CALIBRATE SUBROUTINE
184	X⇄Y	
185	HMS+	
186	×	
187	X⇄Y	
188	HMS+	
189	÷	
190	1/X	
191	PRTX	
192	RTN	
193	*LBLd	CALIBRATE COUNT-UP TIMER
194	GSB4	
195	GSBc	
196	R/S	
197	X⇄Y	
198	RCLD	
199	GTO2	
200	*LBLD	COUNT-DOWN TIMER
201	GSBA	
202	R/S	
203	STO4	START TIME t_s
204	HMS+	
205	RCL5	
206	÷	
207	STOI	
208	RCL4	
209	R/S	
210	*LBL3	FAST LOOP
211	DSZI	
212	GTO3	
213	0	
214	RTN	
215	*LBLe	CALIBRATE COUNT-DOWN TIMER
216	GSB4	
217	RCL4	START TIME t_s
218	X⇄Y	
219	RCL5	C_d
220	GTO2	
221	*LBL4	CALCULATE SWEEP SECOND HAND TIME INTERVAL T_c
222	CHS	
223	HMS+	
224	RTN	

LABELS

A 5 SEC	B MIN INT	C COUNT-UP	D COUNT-DN	E SPLITS
a 5 SEC CAL	b MIN CAL	c C-U to TIME	d CAL C-U	e CAL C-D
0	1 5 SEC LP	2 CAL SUB	3 C-U LOOP	4 T_c
5 5 SEC LP	6 MIN LOOP	7 C-U LOOP	8	9

FLAGS

0
1
2
3

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
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.

HEWLETT hp PACKARD

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

00097-90134

● B C D E