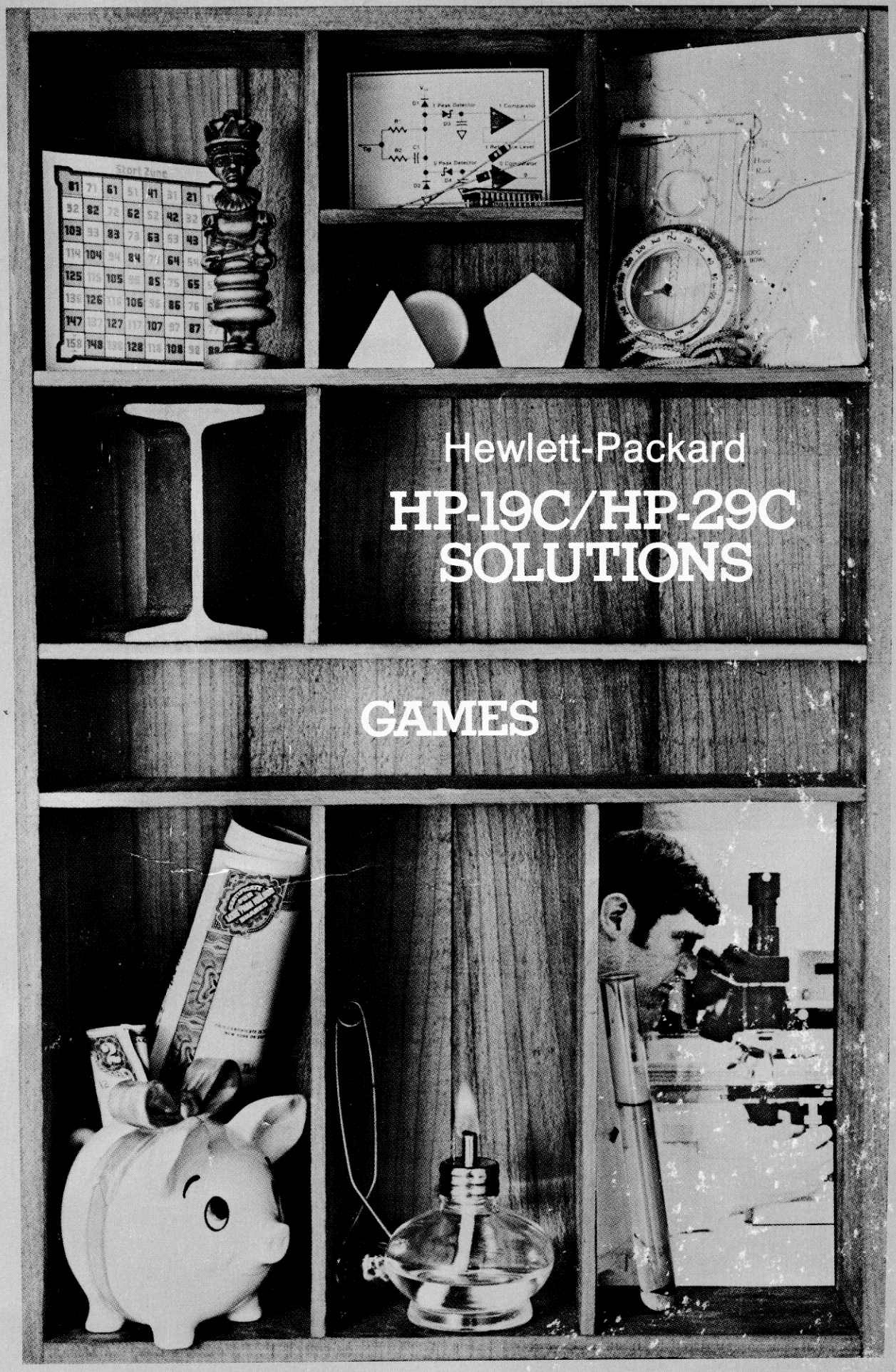


Hewlett-Packard

HP-19C/HP-29C SOLUTIONS

GAMES

INTRODUCTION

This HP-19C/HP-29C Solutions book was written to help you get the most from your calculator. The programs were chosen to provide useful calculations for many of the common problems encountered.

They will provide you with immediate capabilities in your everyday calculations and you will find them useful as guides to programming techniques for writing your own customized software. The comments on each program listing describe the approach used to reach the solution and help you follow the programmer's logic as you become an expert on your HP calculator.

You will find general information on how to key in and run programs under "A Word about Program Usage" in the Applications book you received with your calculator.

We hope that this Solutions book will be a valuable tool in your work and would appreciate your comments about it.

TABLE OF CONTENTS

RACETRACK

This game simulates a two-car race (or a one-car race against time) on a track of arbitrary shape. The track shown may be used, or other tracks of any shape may be designed. The program computes the velocities and positions of the cars.

Initially, both cars are at rest on the start-finish line. Car 1 is at (x,y) position (0,0) and car 2 is at position (5,0.). Player 1 starts. For each move, a player may accelerate in any direction or coast. To make a move, enter the direction (in degrees) and magnitude (0-9) of acceleration and press car number (1 or 2). To coast, enter any direction and a magnitude of zero. For a panic stop, enter a direction exactly opposite (180° away) from your present direction and use a magnitude of 9. The faster you go, the more moves it will take to stop completely.

After making your move, the display will show your car's velocity.

By rolling down the stack, the car's direction of travel, and it's (x,y) position may be displayed.

Traction limits the maximum rate of change of velocity to 9 meters per second per second. Note that this realistically limits the ability to turn when accelerating or declerating.

If the center points of the two cars get within 2 meters of each other, the cars collide, and the display shows flashing zeros. This destroys the two cars, and the game must be started over from the beginning.

It is most convenient to use a fresh sheet of graph paper with the track drawn on it for each game; then the positions of the cars may be plotted, and sequential positions joined by straight line segments. The players must decide after each move whether the car is off of the track, or if it had to go off of the track to travel between the last two positions.

EQUATIONS:

$$V_f = V_i + \Delta V$$

$$P_f = P_i + ((V_i + V_f)/2)t$$

where

V_i & V_f = Initial & final velocities (x & y)

ΔV = Velocity change due to 1 sec. accel.

P_i & P_f = Initial & final positions (x & y)

t = Time, seconds (1 sec. in this prog.)

NOTES:

The program halts, displaying zero if + 90° is used in P→R function; press "R/S" to continue. Direction of car's travel is with respect to the fixed frame of reference of car's starting position (origin) and the start-finish line (0°). See the sketch.

All directions are entered and displayed as A + or - Angle between 0° and 180°.

A collision of the two cars (flashing zeros) ends the game.

Each move advances the car one second in time.

REFERENCES:

Martin Gardner, Mathematical Games, Scientific American, Jan. 1973 and May 1973.

REFERENCES: (continued)

This program is adapted from HP-65 Users' Library program #04326A by Delmer D. Hinrichs.

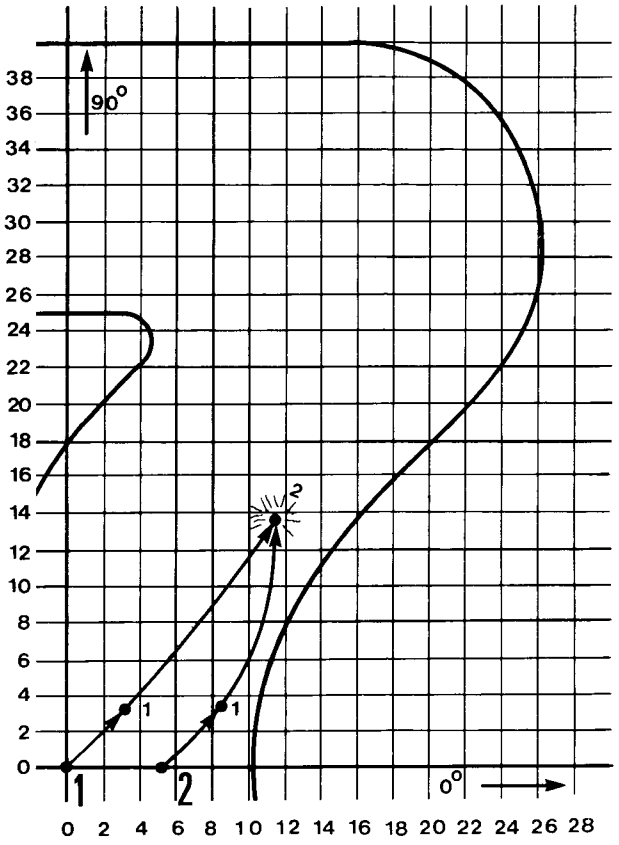


DIAGRAM OF SOLUTION

RULES:

1. A car which goes off the track, either at a plotted position or between plotted positions, loses the race.

2. A car which collides with the other one loses.

3. If both cars cross the start/finish line on the same move, the car which finishes farthest from the line wins.

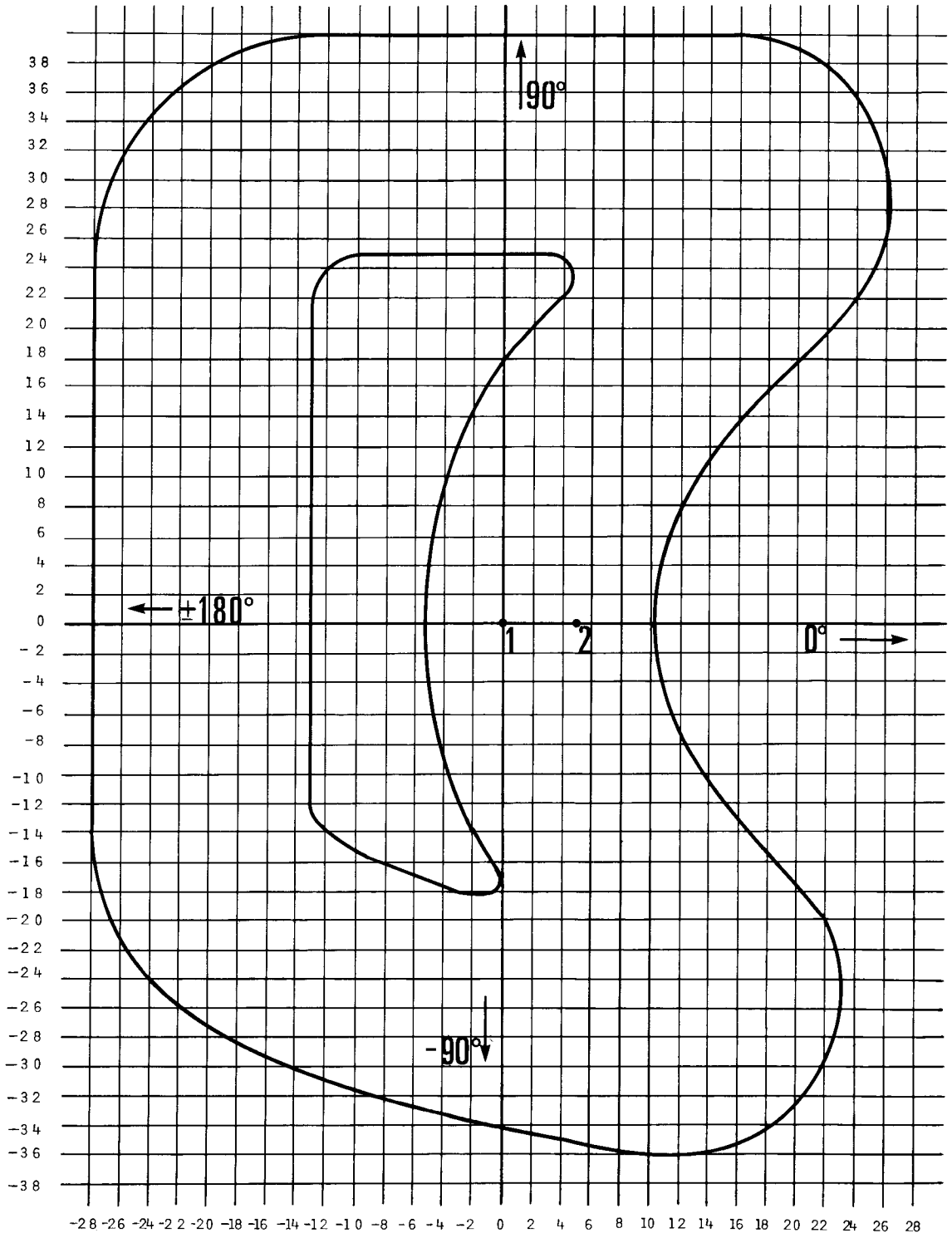
SOLUTIONS:

```
        GSB0
45.00 ENT↑
 9.00 GSB1
 9.00  ***   Car 1 velocity
       R↓
45.00  ***   Direction
       R↓
 3.18  ***   x-position
       R↓
 3.18  ***   y-position

45.00 ENT↑
 9.00 GSB2
 9.00  ***   Car 2 velocity
       R↓
45.00  ***   Direction
       R↓
 8.18  ***   x-position
       R↓
 3.18  ***   y-position

60.00 ENT↑
 9.00 GSB1
17.85  ***   Car 1 velocity
       R↓
52.50  ***   Direction
       R↓
11.80  ***   x-position
       R↓
13.44  ***   y-position

135.00 ENT↑
  9.00 GSB2  Car 2
  0.00  ***  (flashing)collision
             -- car 2 loses
```



RACETRACK

You may wish to copy this to play your game.

For variety, you may wish to draw your own track.

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS
1.	Key in the program				
2.	Initialize		GSB	0	5.00
3.	Enter parameters for car x*:				
	Direction of acceleration, degress	Angle	ENT↑		
	Magnitude of acceleration	Acc	GSB	n*	Velocity
	* n = 1 for car 1; n = 2 for car 2				
4.	Display car n's status:				
	Velocity, meters/sec				Velocity
	Direction of travel, degrees		R↓		Direction
	x-position, meters		R↓		x-pos.
	y-position, meters		R↓		y-pos.
5.	For next move, go to step 3.				
6.	For a new game, go to step 2				
7.	Flashing zeros indicate a collision and the end of the game.				

Program Listings

01 *LBL0	Initialize		50 *LBL6		
02 FIX2			51 RCLi		
03 DEG			52 +		
04 CLRG			53 STOi		New velocity, V_f
05 5	Car 2 x position		54 LSTX		Old velocity, V_i
06 STO6			55 +		
07 R/S			56 2		
08 *LBL1	Car 1		57 ÷		
09 1			58 ISZ		
10 STO0			59 ST+i		New position, P_f
11 GTO9			60 RTN		
12 *LBL2	Car 2		61 R/S		
13 5					
14 STO0					
15 *LBL9	Acc.				
16 R↓					
17 9					
18 X≤Y?	Limit acc. to 9 m/sec^2				
19 X≠Y					
20 R↓					
21 →R	x component y comp.				
22 GSB6	Calculate V_f and P_f				
23 X≠Y	y component				
24 ISZ	Calculate V_f and P_f				
25 GSB6					
26 RCL8					
27 RCL4					
28 -					
29 RCL6	Check for collision (i.e. < 2m.)				
30 RCL2					
31 -					
32 →P					
33 2					
34 X>Y?					
35 GTO5					
36 RCLi					
37 DSZ					
38 RCLi					
39 DSZ					
40 RCLi	Prepare display				
41 DSZ					
42 X≠Y					
43 RCLi					
44 →P	*** "Print Stack" may be inserted before "R/S".				
45 R/S					
46 *LBL5					
47 0	Blinking zero				
48 PSE					
49 GTO5					

REGISTERS

0 Pointer	1 x_1 vel	2 x_1 pos	3 y_1 vel	4 y_1 pos	5 x_2 vel
6 x_2 pos	7 y_2 vel	8 y_2 pos	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

PINBALL MACHINE

This game allows multiple scoring for 5 balls. The balls may be controlled with 4 flippers and tilting is also allowed. As the ball scores, the score is flashed and the running total is displayed. When the ball falls, the display blinks zero. New balls may be set up and played until the 5th ball falls; at that time the total score for the game is displayed as a negative number.

NOTE:

On very rare occasions, the machine will overflow (θ=90°...). In that event, store some other seed in R_2 or start a new game.

This program is adapted from HP-65 Users' Library program #03458A by Peter C. Wang.

SOLUTION:

	GSB0	Initialize
0.32147	STO2	*
	GSB1	Use flipper [1]
200.	***	Total score
	GSB3	[3]
		Flashing Ø-Ball 1 falls
	RCL0	4 balls left
4.	***	
	GSB2	[2]
500.	***	Score=300,total=500
	GSB2	[2]
1800.	***	Total
	GSB1	[1]
		Flashing Ø-Ball 2 falls
	RCL0	
3.	***	3 Balls Left
	GSB4	[4]
2100.	***	
	GSB2	
2700.	***	
	GSB1	
3300.	***	
	GSB1	
8000.	***	
	GSB4	
8800.	***	
	GSB1	
8900.	***	
	GSB2	
9400.	***	
	GSB3	
11300.	***	
	GSB4	
12000.	***	
	GSB4	
12800.	***	
	GSB4	
12800.	***	
	GSB4	
15500.	***	
	GSB4	
16600.	***	
	GSB2	
26200.	***	

* To reproduce this example, store 0.32147 in R2. When playing follow user instruction 2.

SOLUTION:

```
          GSB3
      26400.  ***
          GSB1
      27200.  ***  Score=800,Total=27,200
          GSB3
      27300.  ***
          GSB1
      27800.  ***
          GSB4
      28600.  ***
          GSB3
      29500.  ***
          GSB1
                   Flashing Ø-Ball 3 falls
          RCL0
          2.  ***  2 balls left

          GSB3

          RCL0
          1.  ***  1 ball left

          GSB2
      29500.  ***
          GSB3
      30000.  ***
          GSB1
      30900.  ***
          GSB4
      31000.  ***
          GSB1
      32200.  ***
          GSB1
      32700.  ***
          GSB3
      39800.  ***
          GSB2
      39800.  ***
          GSB2
      40200.  ***
          GSB3
      40900.  ***
          GSB2
     -40900.  ***  Last ball falls, total=
                     40,900
```

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS
1.	Key in the program				
2.	Initialize -- let initialization run		GSB	0	
	between 1-45 seconds; then halt by		Any		
	pressing any key		Key		
3.	Play the ball with any of the four		GSB	1	
	flippers. The score flashes and the total	or	GSB	2	
	is displayed	or	GSB	3	Score/total
		or	GSB	4	
4.	Repeat step 3 with the same ball until a				
	blinking zero appears, indicating the				
	ball has fallen.				
5.	When the ball has fallen, set up a new	Blinking	Any		
	ball and go to step 3 to play it.	zero	Key		0.
6.	The number of balls remaining may be		RCL	0	# of balls
	recalled at any time				Left
7.	When the fifth ball has fallen the game				
	is over and the total score is displayed				
	as a negative number				
8a.	You may "tilt" the machine at any time		GSB	5	
	(this operation changes the seed)				
8b.	Stop the tilting operation by pressing		Any		
	any key. Go to step 3.		Key		
9.	For a new game, go to step 2.				

Program Listings

01 *LBL0	Initialize	50 STO2	
02 CLRG		51 RCL4	
03 1		52 RCL7	θ > 85°?
04 STO2		53 X≤Y?	Ball falls(tanθ too large)
05 .		54 GTO8	
06 9		55 RCL2	0 < |tanθ| < 11.43
07 8		56 1	
08 5		57 0	
09 3		58 x	
10 STO3		59 INT	
11 EEX		60 RCL5	
12 2		61 x	"round" to 100's
13 STO5		62 ST+1	
14 8		63 PSE	Σ scores
15 5		64 RCL1	Display score
16 STO7		65 R/S	
17 5		66 *LBL8	Display total
18 STO0	i=# of balls=5	67 DSZ	Reduce # of balls
19 FIX0		68 GTO9	Indicate fall of ball
20 *LBL5	tilting operation	69 RCL1	
21 RCL3	$(.9853)^n$-->R_2	70 CLRG	
22 ST×2	n=#of loops	71 CHS	
23 GTO5		72 R/S	Display game total as a negative no.
24 *LBL1		73 *LBL9	
25 3	39=k for flipper[1]	74 0	Blinking zero
26 9		75 PSE	
27 GTO4		76 GTO9	
28 *LBL2		77 R/S	
29 5	53=k for flipper[2]		
30 3			
31 GTO4			
32 *LBL3			
33 RCL7	85=k for flipper[3]		
34 *LBL4			
35 RCL2			
36 RCL7	lastx = k for flipper [4]		
37 x			
38 +			
39 RCL3			
40 Y^x			
41 RCL3	(85) R_2 + K		
42 x			
43 X^2			
44 FRC			
45 RCL5			
46 x			
47 STO4	θ		
48 TAN			
49 ABS	|tanθ|		

REGISTERS

0 i=# of balls	1 Σ Scores	2 Seed	3 .9853	4 θ	5 100
6	7 85	8	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

15 BALL ROTATION POOL

The game proceeds with a random selection of the players' shots being scoring shots. The balls are pocketed in rotation (1 through 15). Player skills can be varied by selecting a skill factor between 1 and 20. This determines the relative number of scoring shots to total shots. The random sequences are variable by seed number selection. The program continously tallys each of two players scores.

NOTE:

This program is adapted from HP-65 Users' Library program #03427A by Robert A. Plack.

Guide Lines for Skill Factor Selection:

SKILL FACTOR:

2---	To pocket	15	balls	may	need	120	shots
7---	"	"	"	"	"	50	"
10---	"	"	"	"	"	30	"
13---	"	"	"	"	"	22	"
15---	"	"	"	"	"	20	"
17---	"	"	"	"	"	16	"

SOLUTION:

```
   11.00 STO6   Skill factor
1.2345987 STO7  Seed
         GSB1   Initialize
          R/S   Shoot
       1. ***   Sunk ball #1
          R/S   Shoot
       2. ***
          R/S   Shoot
       3. ***
          R/S   Shoot
       4. ***
          R/S   Shoot
       0. ***   Miss
          R/S   Player 2 shoots
       5. ***
          R/S   Shoot
       0. ***   Miss
         GSB2   Review
    4.01 ***    Score:  Player 1 has
                        sunk 4 balls;
                        player 2 has
                        sunk 1.
```

SOLUTION:

(after more play)

```
          R/S
      11. ***
          R/S
      12. ***
          R/S
      13. ***
         GSB2   Review
   10.03 ***    Score
          R/S   Shoot
       0. ***
          R/S   Shoot
       0. ***
          R/S   Shoot
       0. ***
          R/S   Shoot
      14. ***
          R/S   Shoot
   10.05 ***    Game over
                Player 1 wins
```

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS
1.	Key in the program				
2.	Enter skill factor (between 1 and 20)	S.F.	STO	6	
3.	Enter seed (any number)	Seed	STO	7	
4.	Initialize		GSB	1	-1.00
5.	First player shoots		R/S		Ball # or 0.
6.	A player continues to shoot until a zero		R/S		Ball # or 0.
	is displayed, indicating the shot was missed				
	(If the shot was made, the ball number is				
	displayed). It is then the opponent's turn.				
7.	The score is tallied automatically to				
	review the score.		GSB	2	$S_1S_1 \cdot S_2S_2$
	e.g., 7.02 is read player one has sunk 7				
	balls, player two has sunk 2.				
8.	When all fifteen balls have been sunk, the				
	game is over and the score is displayed.				
9.	For a new game, go to step 4 (or step 2				
	if desired).				

Program Listings

01 *LBL1	Initialize	50 *LBL2	Get score
02 0		51 RCL2	
03 STO1		52 1	
04 STO2		53 %	
05 STO8		54 RCL1	
06 1	Pointer	55 +	
07 STO0		56 FIX2	
08 CHS		57 R/S	Display score in format $S_1S_1 \cdot S_2S_2$
09 STO4		58 GTO0	
10 R/S	"Shoot"	59 R/S	
11 *LBL0			
12 FIX0			
13 RCL7	Seed		
14 Pi			
15 +			
16 X²			
17 FRC			
18 STO7			
19 EEX			
20 3			
21 ×			
22 FRC			
23 EEX			
24 2			
25 ×			
26 RCL6	Skill factor		
27 ÷			
28 INT			
29 6			
30 X≤Y?	RND > 6?		
31 GTO9	Miss		
32 1	Made the shot;		
33 ST+i	Increment score and		
34 ST+8	count		
35 1			
36 5			
37 RCL8			
38 X=Y?			
39 GTO2	Game over		
40 R/S	Display ball no.		
41 GTO0			
42 *LBL9			
43 RCL4			
44 CHS			
45 STO4	Switch pointer		
46 ST+0			
47 0	Display 0 to		
48 R/S	indicate a miss		
49 GTO0			

REGISTERS

0 i,pointer	1 Score (player1)	2 Score (player 2)	3	4 ± 1	5
6 Skill factor	7 Seed	8 Counter	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

ROULETTE

The player bets by entering the dollar amount of the bet and the number on which the bet is placed in the form B.##. For instance, $5 on #7 would be entered as 5.07 and $50 on #27 would be 50.27.

A winning number bet pays off at 32-to-1. A winning even-odd bet pays off at one-to-one.

In the "win" sequence, the player's total bankroll is displayed. In the "lose" sequence, the Roulette number is displayed, after which the total bankroll may be displayed by pressing R/S.

NOTES:

1. Bet only whole dollars.
2. The maximum bet is $99,999,999
3. If your winnings cause your bankroll to exceed 9.9999999×10^{99}, ERROR will be displayed.

This program is adapted from HP-65 Users' Library program #03076A by William A. Sholar.

SOLUTION:

	GSB5	
0.00	STO2	
0.00	STO7	Seed
500.27	GSB1	
0.28	***	#
	R/S	
-500.00	***	Bankroll
500.27	GSB1	
0.02	***	#
	R/S	
-1000.00	***	
500.27	GSB1	
0.14	***	#
	R/S	
-1500.00	***	
500.27	GSB1	
14500.00	***	Winner
500.27	GSB1	
30500.00	***	Another winner
	GSB2	Bet even
0.25	***	#
	R/S	
0.00	***	Total bankroll

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS
1.	Key in the program				
2.	Initialize		GSB	5	
3.	Store your bankroll	$	STO	2	
4.	Store seed (any number)	Seed	STO	7	
5.	Play one of the following:				
5a.	Enter bet and number in BB.## format	B.##	GSB	1	*
	and spin				
5b.	Enter bet on even number and spin	B.	GSB	2	*
5c.	Enter bet on odd number	B.	GSB	3	*
6.	Repeat step 4 as often as desired, or				
	go to step 3 to change your luck.				
	* If you win, your updated total is				
	displayed. If you lose, the spin is				
	displayed. To see your new total.		R/S		Total

Program Listings

01 *LBL5	Initialize	50 R/S	Display total
02 FIX2		51 *LBL9	
03 CLRG		52 RCL7	Seed
04 *LBL1		53 Pi	
05 INT		54 +	
06 STO1	Bet ---> R_1	55 X^2	
07 LSTX		56 FRC	
08 FRC		57 STO7	New seed
09 STO3	# ---> R_3	58 EEX	
10 GSB9	Spin	59 2	
11 RCL3		60 ×	
12 X≠Y?		61 3	
13 GTO0	You lose	62 ÷	
14 RCL1	You win	63 INT	
15 3		64 1	
16 2		65 %	
17 ×		66 STO6	Spin ---> 6
18 ST+2		67 RTN	
19 RCL2		68 R/S	
20 R/S	Display total		
21 *LBL2	Bet (even)		
22 0			
23 STO0			
24 GTO8			
25 *LBL3	Bet (odd)		
26 .			
27 5			
28 STO0			
29 *LBL8			
30 R↓	Bet, $		
31 STO1			
32 GSB9	Spin		
33 5			
34 0			
35 ×			
36 FRC	0 or .5		
37 RCL0			
38 X≠Y?			
39 GTO0			
40 RCL1	You lose		
41 ST+2	You win		
42 RCL2			
43 R/S	Display total		
44 *LBL0	Lose routine		
45 RCL1			
46 ST-2	Deduct bet		
47 RCL6			
48 R/S	Display spin ("pause" may replace "R/S")		
49 RCL2			

REGISTERS

0 0 or .5	1 Bet, $	2 Total	3 #	4	5
6 Spin	7 Seed	8	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

TIC - TAC - TOE

This program plays tic-tac-toe with the user. The keyboard of the machine is used as the playing board, with each digit representing one of the nine positions, as shown at right. The machine moves first, into a side position (position 2). The user may move into any of the eight remaining positions. As play continues, user may move into any unoccupied position for each move.

7	8	9
4	5	6
1	2	3

Tic-tac-toe can be won only if one player makes a mistake. This program takes advantage of user mistakes by completing a row of three, or by setting a trap to force a win. If all user moves are correct, a draw results. The side opening by the machine gives the user a better chance to avoid losing.

This program operates on a game tree look-up basis; a different register, containing the machine responses, may be selected for each of the eight possible user first moves.

NOTES:

1. Illegal moves (to occupied positions) gives erroneous results.
2. No win, lose, or draw signals are given; the user must keep track of the progress of the game.

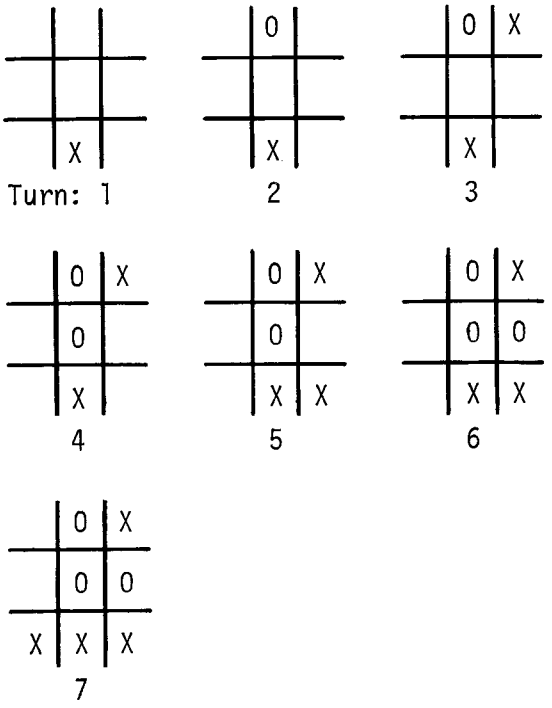
This program is adapted from HP-65 Users' Library program #03363A by Delmer D. Hinrichs.

REFERENCE:

Gardner, Martin, Mathematical Puzzles & Diversions, Simon and Schuster, New York, 1959 pages 37-46.

EXAMPLE:

Machine plays "X", User plays "O"



SOLUTIONS:

```
     1.5873649 ST01   Store constants
     3.5891467 ST02
       4.13598 ST03
     5.1374698 ST04
       6.31578 ST05
       7.13589 ST06
  0.154763657 ST07
       9.31587 ST08
```

(1)

```
        FIX0
        GSB1
 2.     ***    Machine's 1st move
 8.     R/S    Player's 1st move
 9.     ***
 5.     R/S
 3.     ***
 6.     R/S
 1.     ***    Machine Wins
```

(2)

```
        GSB1
 2.     ***
 9.     R/S
 3.     ***
 1.     R/S
 5.     ***
 8.     R/S
 7.     ***    Machine wins
```

(3)

```
        GSB1
 2.     ***
 1.     R/S
 5.     ***
 8.     R/S
 7.     ***
 3.     R/S
 6.     ***
 9.     R/S
 4.     ***    Machine Wins
```

(4)

```
        GSB1
 2.     ***
 8.     R/S
 9.     ***
 1.     R/S
 3.     ***
 6.     R/S
 5.     ***
 4.     R/S
 7.     ***    Machine wins
```

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1.	Enter program			
2.	Store constants:			
	1.5873649		STO 1	
	3.5891467		STO 2	
	4.13598		STO 3	
	5.1374698		STO 4	
	6.31578		STO 5	
	7.13589		STO 6	
	0.154763657		STO 7	
	9.31587		STO 8	
3.	Set display		f Fix	
			0	
4.	Start game		GSB 1	2
5.	Enter move (any number 1-9)	Player's Move	R/S	
6.	Repeat step 5 until the outcome is resolved (HP-29C/19C wins or a draw). The player must keep track of the progress of the game.			
7.	For a new game, go to step 4.			

Program Listings

01	*LBL1	Initialize	50	3	No,response is 3
02	EEX		51	R/S	Enter player's 3rd move
03	5	10^5	52	1	
04	STO9		53	X=Y?	Is it 1?
05	2	Machine's 1st move	54	6	Yes, 6 is response
06	R/S	Player's 1st move	55	R/S	No, 1 is response; end game
07	8		56	*LBL8	Response is 1
08	STO0	i=8	57	1	Enter player's move
09	X=Y?	Is it 8?	58	R/S	
10	GTO6	Yes	59	3	Is it 3?
11	R↓	No	60	X=Y?	Yes, response is 5
12	*LBL3	Find reg. which contains correct responses	61	5	
13	RCLi		62	R/S	No, response is 3; end game
14	GSB0		63	*LBL7	
15	DSZ		64	CLX	
16	GTO3		65	1	Is it 1?
17	*LBL0		66	X=Y?	Yes, use last 5 digits in R_7
18	INT		67	RCL9	
19	X=Y?	Match?	68	RCL7	Mult. R_7 by 1 or 10^5 same in lastx
20	GTO4	Hit	69	x	
21	R↓	Miss	70	+	Go find responses
22	RTN		71	GTO5	
23	*LBL4		72	R/S	
24	RCLi				
25	+	Save in last x			
26	*LBL5				
27	GSB9	Find response			
28	X=Y?	Is it correct?			
29	GSB9	No, get it this time			
30	R/S	Enter player's move			
31	GTO5				
32	*LBL9				
33	LSTX				
34	FRC				
35	1				
36	0				
37	x	~			
38	INT	Saves ~ in last x			
39	RTN				
40	*LBL6	Interactive strategy for 8			
41	9	Player's 2nd move			
42	R/S				
43	4	Is it 1 or 3?			
44	X>Y?	Yes			
45	GTO7				
46	CLX				
47	6	Is it 6 or 7?			
48	X≤Y?				
49	GTO8				

REGISTERS

0 Pointer	1 Library	2 Lib.	3 Lib.	4 Lib.	5 Lib.
6 Lib.	7 Lib.	8 Lib.	9 10^5	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

BELL FRUIT

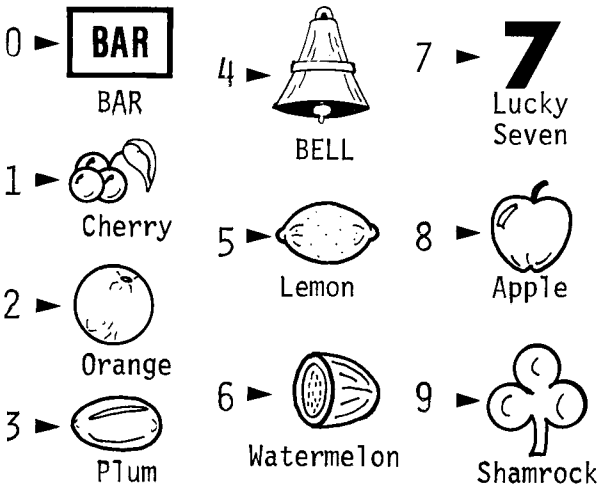
Contrary to popular belief, the "Auto-Bell" and "Bell Fruit" brands of slot machines are not rigged. However, the odds at getting a jackpot pattern are extremely low. For example, on a real slot machine, each 'wheel' contains 20 symbols, only one of which is a bar. Thus, with 3-wheels, a 3-bar combination (or 'Jackpot') comes up once every 8000 plays!

This program is more sporting.(depending on the seed used in initialization, this program can be down right generous). When GSB 1 is pressed the 'wheels' spin and a 3 digit decimal is arrived at. (The no. is to the right of the decimal point, ignore the '0' to the left). A dime is deducted from the 'pot' (R_1). If you win, the payoff amount is paid into the 'pot'. This may be reviewed at any time by pressing RCL 1. Any 3-of-a-kind (except cherries) wins $1.00. Any 2-of-a-kind (except cherries) followed by a 'bar', wins $1.00. A cherry in the first position wins 20¢. A cherry in the second position, when following the 1st cherry, wins an additional 30¢. All other combinations are "Fruit-Salad" and win you zilch! Good Luck.

NOTES:

It's best to key in a many digit decimal as a seed, as opposed to a small number (use a number like '251.0637948' instead of '3'). Due to the nature of the program, one is generally assured of winning 20¢ (one cherry) immediately following a jackpot (3-bars).

This program is adapted from HP-65 Users' Library program #03044B by Craig A. Pearce.



SOLUTION:

32147.000	GSB5	Seed
	GSB1	Play
0.174	***	$.20 winner
	GSB2	Pot now has $.10
0.10	***	
	GSB1	Play
0.994	***	No luck
	GSB2	As expected, pot even
0.00	***	
	GSB1	
0.067	***	Lost
	GSB1	
0.385	***	Lost
	GSB1	
0.905	***	Lost
	GSB2	Sure enough,$.30 in hole
-0.30	***	
	GSB1	
0.997	***	Lost
	GSB1	
0.120	***	Won $.20
	GSB1	
0.496	***	Lost
	GSB1	
0.722	***	Lost
	GSB1	
0.999	***	Won $1.00!
	GSB2	Let's stop while
0.40	***	we're ahead

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1.	Key in the program			
2.	Enter seed (any number below 10^6)	Seed	GSB 5	
	and initialize			
3.	Play		GSB 1	0.XYZ
	Winning combinations:			
	0.1nm pays 20¢			
	0.11n pays 30¢			
	0.XXX pays $1.00			
	0.XX0 pays $1.00			
	(where X ≥ 2)			
	0.000 pays $10.00			
4.	Recall pot of winnings (or losses) at		GSB 2	Pot
	any time			
5.	Repeat steps 3-4 any number of times			
6.	For a new game, go to step 2			
7.	For a new seed (at any time)	Seed	STO 8	

Program Listings

01 *LBL5	Seed & initialize	50 1	
02 CLRG		51 GSB0	
03 STO8		52 *LBL7	
04 DEG		53 RCL5	
05 R/S		54 .	
06 *LBL1		55 1	-10¢
07 FIX3		56 CHS	
08 RCL8		57 *LBL0	
09 EEX		58 ST+1	
10 3		59 R↓	
11 x		60 RTN	
12 COS		61 *LBL6	Cherry routine
13 ABS		62 .	
14 STO8	RND	63 2	Pay 20¢ for 1 cherry
15 EEX		64 GSB0	
16 6		65 RCL3	
17 +	Adjust format (0.XYZ)	66 X≠Y?	Second cherry?
18 LSTX		67 GTO7	
19 -		68 .	
20 FRC		69 3	Pay 30¢ for 2nd cherry
21 STO5		70 GSB0	
22 GSB9		71 GTO7	
23 STO2	X	72 *LBL9	Peel off digits
24 -		73 1	
25 GSB9		74 0	
26 STO3	Y	75 x	
27 -		76 ENT↑	
28 GSB9		77 INT	
29 STO4	Z	78 RTN	
30 1		79 *LBL2	RCL Pot
31 RCL2		80 RCL1	
32 X=Y?	A cherry?	81 FIX2	
33 GTO6		82 R/S	
34 RCL3			
35 X≠Y?	Do 1st two digits match?		
36 GTO7	Not a winner		
37 RCL4			
38 X≠Y?	Do 2nd two digits match?		
39 GTO8			
40 1	If not, test for Z=0 otherwise pay		
41 GSB0	$1. for 3-way match		
42 1	Set flag to show		
43 STO0	3-way match		
44 R↓			
45 *LBL8	Z=0? if not, no.		
46 X≠0?	is not a winner		
47 GTO7	Was no. a 3-way		
48 9	match? No, input a1		
49 DSZ	otherwise input a 9 for a jackpot		

REGISTERS

0 Flag	1 Pot	2 "X" digit	3 "Y" digit	4 "Z" digit	5 RND(.XYZ)
6	7	8 Seed	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

BLACKJACK (21)

The player places a bet and draws a card. Jack, Queens, and Kings count as 10 and Aces count as 1. The player keeps taking "hits" (drawing cards) until satisfied with the total. If his total exceeds 21 he loses. The machine draws until its total exceeds the player's.

If its total exceeds 21, you win.

If you win, your bet is added to your bankroll and should you lose, your bet is deducted.

The program uses a random number generator to pick the cards where the probabilitites of drawing a deuce through an ace are all equal.

NOTE:

This program is based on three HP-65 Users' Library programs: 237A by Duke Castle, 1296A by Gary D. Campbell, and 2024A by Mordecai Schwartz, M.D.

SOLUTION:

2.654	STO1	Enter seed
50.00	STO2	Enter your bankroll
15.00	GSB1	Enter your bet and draw 1st card
8.	***	Total count after 1st card
	R/S	Hit
18.	***	Total after second card
		Stick
	GSB2	Machine plays
		After each card, the following count totals are flashed: 7,8,9,14, 24
65.00	***	You win, bankroll is now $65.00
15.00	GSB1	Bet $15.00 on another game
1.	***	Total count after 1 card
	R/S	
3.	***	After 2 cards
	R/S	
12.	***	After 3 cards
	R/S	
14.	***	After 4 cards
	R/S	After 5 cards your total of 24 is flashed.
50.00	***	You lose-Bankroll is back to $50.00
20.00	GSB1	Now you bet $20.00
3.	***	
	R/S	
5.	***	
	R/S	
10.	***	
	R/S	
14.	***	
	R/S	5 card total is flashed:22
30.00	***	Loss reduces bankroll to $30.00
10.00	GSB1	You bet only $10.00
10.	***	
	R/S	
11.	***	
	R/S	
15.	***	Decide to stick with a 15 total
	GSB2	Machine wins again with totals of 10,16 and 20
20.00	***	You are left with $20.00

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS
1.	Key in the program				
2.	Initialize:				
	Store seed	Any number	STO	1	
	Store bankroll to play with	W_0	STO	2	
3.	Enter bet and draw a card	Bet	GSB	1	P_1, value of first card
4.	Take a "hit"		R/S		P_j, players total after ith card
5.	Repeat step 4 until satisfied with the				
	total (go to step 6)				
6.	"Stick". Let machine draw,machine draws		GSB	2	$M_1, M_2 \ldots W$
	until it's total:				
	a) exceeds 21 in which case you win and				
	your bet is added to your bankroll.				
	b) is greater than your total but less				
	than 21, in which case you lose and				
	your bet is subtracted from your				
	bankroll, W.				
7.	For another hand, go to step 3.				

Program Listings


```
01 *LBL1
02  STO6      Bet
03    0
04  STO0
05  STO4
06  STO5
07  FIX0
08 *LBL7
09  GSB0      Get card
10  ST+5
11    2
12    1
13  RCL5      Total
14   PSE
15  X=Y?
16  GTO9      You win
17  X>Y?
18  GTO8      You lose
19   R/S
20  GTO7
21 *LBL2      Machine draws
22  GSB0
23  ST+4
24    2
25    1
26  RCL4
27   PSE
28  X>Y?      You win
29  GTO9
30  RCL5      Your total
31   X≠Y
32  X>Y?
33  GTO8      You lose
34  GTO2      Machine draws
35 *LBL9        again
36    1       Set flag
37  STO0
38 *LBL9
39  RCL6      Bet
40   DSZ      Add winnings and
41   CHS        subtract losses
42   CHS
43  ST+2
44  FIX2
45  RCL2      Display total
46   R/S        (bankroll)
47 *LBL0        random number
48    1         generator
49    0
50  RCL1
51    Pi
52    +
53    X²
54   FRC
55  STO1
56    1
57    3
58    x
59    1
60    +
61   INT
62  X>Y?
63  X≠Y       10. for jack-king
64  RTN
65  R/S
```

REGISTERS

0 i	1 Seed	2 Bankroll	3	4 Mach. Tot	5 Player's Tot
6 Bet	7	8	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

CANNIBALS AND MISSIONARIES

The program completely simulates the classical cannibal-missionary river crossing problem in the following form:

3 missionaires and 3 partially-civilized cannibals must cross a river with a boat that can hold no more than 3 passengers. At no time may cannibals outnumber missionaries at either bank or on the boat lest the cannibals regress to an earlier mode of behavior! Further, cannibal(s) left aboard the boat alone will run off with it after launching.

Missionaries, cannibals and boat are all initially on the left bank. Press GSB 1 once for each cannibal you wish to put aboard and GSB 2 for each missionary. After each crossing (GSB 3) or return (GSB 4) the right bank distribution (those already across)is displayed in the form $0.C_RM_R$:

For example:
- 0.00 No one has crossed-initial condition
- 0.23 2 cannibals & 3 missionaries on Rt. bank
- 0.33 Successful simulation-everyone across

After a crossing (GSB 3 or GSB 4), improper operations are appropriately punished:

		DISPLAY
A.	Impossible crossing-boat on wrong bank	"Error"
B.	Boat adrift (no one on) or stolen (no miss. on)	1.00
C.	C's outnumber M's on boat	2.00
D.	Boat sinks - 4 or more aboard	4.00
E.	M's outnumbered at either bank	3.00

NOTE:

This program is adapted from HP-65 Users' Library program number 02286A by Mordecai Schwartz, M.D.

SOLUTION:

	GSB5	Initialize
	GSB1	Cannibal boards
	GSB2	Missionary boards
	GSB2	Missionary boards
	GSB2	Missionary boards
	GSB3	Left to right crossing
4.00	***	Boat overloaded-try again
	GSB5	Initialize
	GSB2	M
	GSB2	M
	GSB2	M
	GSB3	-------->
0.03	***	0 cannibals & 3 Miss.
	GSB2	M on right bank
	GSB2	M
	GSB4	<-----(right to left
3.00	***	C's outnumber M's on left bank
	GSB5	Initialize
	GSB1	C
	GSB2	M
	GSB3	-------->
0.11	***	1C + 1M on right bank
	GSB2	M
	GSB4	<-------
0.10	***	1C + 0M on right bank
	GSB1	C
	GSB1	C
	GSB2	M
	GSB3	-------->
2.00	***	Missionaries out-numbered on boat

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1.	Key in the program			
2.	Initialize: Cannibals, missionaries, and boat at left bank		GSB 5	0.00
3.	Load the boat:			
3a.	A cannibal boards		GSB 1	1.00
3b.	A missionary boards		GSB 2	1.00
4.	Repeat 3a/3b until all passengers are loaded			
5.	Cross the river (in the proper direction):			
5a.	Left to right --------->		GSB 3	$0.C_RM_R$*
5b.	Right to left <---------		GSB 4	$0.C_RM_R$*
	Output = $0.C_RM_R$			
	e.g. 0.23 means 2 cannibals and 3 missionaries on right bank			
6.	Repeat steps 3-5 until everyone is on right bank.			
7.	For a new game or after an improper operation *, go to step 2.			
	* Outputs after an improper operation:			
	a. Impossible crossing-boat on wrong side			Error
	b. Boat adrift (no one on) or stolen (only C's on)			1.00
	c. M's outnumbered on boat			2.00
	d. Boat overloaded - 4 or more aboard			4.00
	e. C's outnumber M's on either bank or too many M's called, e.g., 1 M on bank and 2 M's loaded aboard			3.00

If crossing involves multiple errors, the display hierarchy is as above.

Program Listings

01 *LBL5	Initialize	50 *LBL7	Safe crossing configuration
02 FIX2	0	51 0	
03 CLRG	R/S	52 STO3	
04 0		53 STO4	
05 R/S		54 RCL1	
06 *LBL1	A cannibal boards	55 EEX	
07 1		56 1	
08 ST+3		57 ÷	}
09 R/S		58 RCL2	
10 *LBL2	A missionary boards	59 EEX	
11 1		60 2	} Set up display
12 ST+4		61 ÷	
13 R/S		62 +	
14 *LBL3	Cross river (--→)	63 R/S	Display 0.$C_R M_R$
15 RCL5		64 *LBL4	Cross river (←)
16 X≠0?	Left bank?	65 RCL5	
17 GTO0		66 X=0?	Right bank?
18 1		67 GTO0	
19 STO5	Set flag	68 0	Clear flag
20 RCL3	}	69 STO5	
21 ST+1	New right bank Distribution	70 RCL3	}
22 RCL4		71 ST-1	New right bank Distribution
23 ST+2		72 RCL4	
24 *LBL9	River crossing over or back	73 ST-2	
25 1		74 GTO9	
26 STO0		75 *LBL0	
27 RCL4		76 0	Display "Error"
28 X=0?		77 ÷	
29 GTO8	i = 1	78 *LBL8	
30 ISZ		79 RCL0	Display error code
31 RCL3		80 R/S	
32 X>Y?			
33 GTO8	i = 2		
34 +			
35 4			
36 STO0			
37 X≤Y?			
38 GTO8	i = 4		
39 RCL2			
40 X=0?			
41 GTO7	Safe		
42 3			
43 X=Y?	Safe		
44 GTO7			
45 RCL1			
46 RCL2			
47 DSZ			
48 X≠Y?	i = 3		
49 GTO8			

REGISTERS

0 Error Code	1 C_R Right bank	2 M_R Right bank	3 C_B Boat	4 M_B Boat	5 Flag 1=set 0=clear
6	7	8	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

HUNT A MOVING SUBMARINE

Using your destroyer, you try to locate the position of the enemy submarine in a 10 x 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 GSB 2. If the submarine is in one of the 8 adjacent squares(or direclty 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. A blinking "1" indicates 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 (but it will not move off the grid).

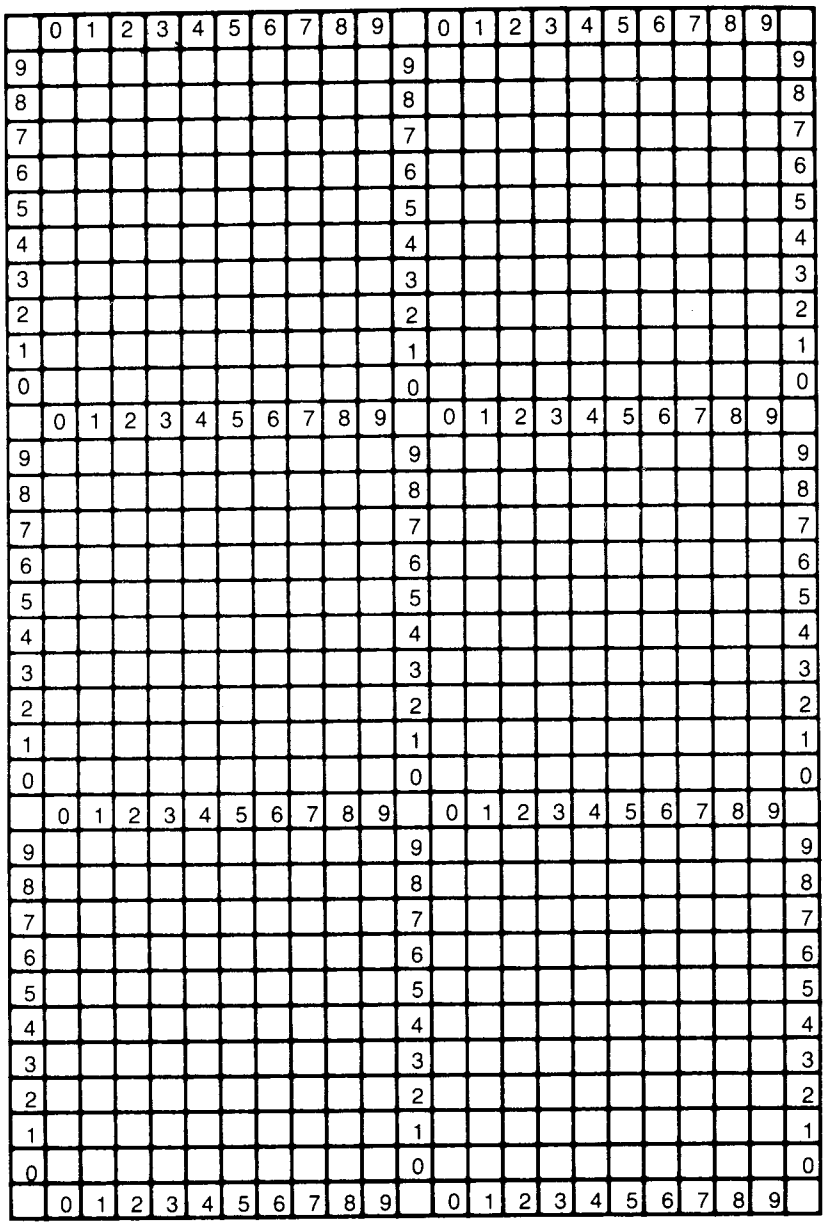
A depth charge has a range of 0.8. When you position your destroyer for a depth charge drop, (or when you prepare for a sonar reading), you may move anywhere on the board, not just to the center of a square. For instance, a depth charge dropped from a (2.5,6.5) location would destroy any submarine in the center of square (2,6)(2,7)(3,6) and (3,7). Note that sonar's range=1.8

NOTE:

This program is adapted from HP-65 Users' Library program number 01957A by Moshi M. Breiner.

SOLUTION:

17.20 GSB1	Seed
2.00 ENT↑	
2.00 GSB2	Search
0.00 ***	
4.50 ENT↑	
4.50 GSB2	Search
0.00 ***	
1.50 ENT↑	
6.50 GSB2	Search
1.00 ***	Within 1.8 units
2.00 ENT↑	
7.00 GSB2	Search
0.00 ***	
1.00 ENT↑	
5.00 GSB2	Search
0.00 ***	
3.00 ENT↑	
5.00 GSB2	Search
1.00 ***	Within 1.8 units
2.50 ENT↑	
5.50 GSB3	Fire depth charge
0.00 ***	Miss
2.50 ENT↑	
3.50 GSB2	Search
1.00 ***	Within 1.8 units
1.50 ENT↑	
3.50 GSB3	
1.00 ***	(flashing) Hit!



Playing board for Hunt a Moving Submarine.

You might wish to use copies of this page for your games.

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1.	Key in the program			
2.	Initialize and enter seed (any number			
	between 1 and 100)	Seed	GSB 1	1.00
3.	Take a sonar reading	R*	ENT↑	
	or	C*	GSB 2	0 or 1
	Fire a depth charge	R	ENT↑	
		C	GSB 3	0 or
				blinking 1
4.	Repeat step 3 until the submarine is			
	destroyed (blinking 1.)			
5.	For a new game, go to step 2.			
	* R = Row			
	C = Column			

Program Listings

01 *LBL1	Initialize	49 5	
02 CLRG		50 -	
03 STO2	Seed	51 ENT↑	
04 DEG		52 ABS	
05 FIX2		53 ÷	± 1 positive if RND ≥ 5 neg if RND ≤ 4
06 GSB0		54 DSZ	0 or 3 results
07 STO4	C_S	55 RCL4	$R_S \pm 1$ or $C_S \pm 1$
08 GSB0		56 +	
09 STO1	R_S	57 9	If $P_i+x = 0$ or $P_i+x = 10$, the new coordinate is $P_i-x =$ $\|\|P_i+x-9\|-9\|$
10 1		58 -	
11 STO5	Clear flag 1	59 ABS	
12 R/S		60 9	
13 *LBL3	Charge fired	61 -	
14 0		62 ABS	
15 STO5	Set flag 1	63 ISZ	i = 1 or 4
16 R↓		64 STOi	
17 *LBL2		65 RCL5	Flag 1
18 1		66 X=0?	
19 STO0	i = 1	67 STO3	Zero out R_3
20 R↓		68 1	
21 RCL4		69 STO5	
22 -	$x_2 - C_S$	70 RCL3	Clear flag 1
23 X≠Y		71 R/S	
24 RCL1		72 *LBL0	Result
25 -	$x_1 - R_S$	73 1	
26 →P	δ,Euclidean dist.	74 PSE	Blink 1.
27 .		75 GTO8	
28 8		76 *LBL0	Random number generator
29 -	$\delta - .8$	77 2	
30 X>0?		78 RCL2	
31 GTO9	Continue flag 1	79 +	
32 RCL5		80 STO2	
33 X=0?		81 COS	
34 GTO8	Charge within .8 units	82 ABS	
35 R↓		83 EEX	
36 *LBL9		84 5	
37 1		85 x	
38 STO3		86 FRC	
39 X≤Y?	$\delta \geq 1.8$?	87 1	
40 ST-3	Zero out R_3	88 0	
41 GSB0		89 x	
42 4		90 INT	
43 X≤Y?	4 < RND?	91 RTN	RND = 6th figure of cos (R_2)
44 STO0	i=4; C_S will change and R_S will remain the same	92 R/S	
45 RCL1		NOTE: P_i = Any row or column location for sub.	
46 GSB0			
47 4			
48 .			

REGISTERS 0=set 1=clear

0 i	1 R_S	2 Seed	3 Result	4 C_S	5 Flag 1
6	7	8	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

ARTILLERY GAME

In this game, the gun has a maximum range of 10,000 meters. At the start of a game, the target is at a random direction (0-360°) and random distance (5,000-10,000 meters) from the gun. To locate the target, fire a shell in an arbitrary direction and elevation (i.e., 90° direction and 45° elevation). The display then shows the direction and distance from the shell hit to the target. From this, it is possible to determine where the target was. The target randomly changes its direction from the gun by up to ± 5°, and randomly moves towards the gun by 0-1,000 meters after each shell hits. Taking into account the expected movement of the target, a new direction and elevation are estimated, and another shell fired. Corrections are made and shells fired until either the target is blown up (shell hits within 100 meters of target) or the gun is destroyed (target gets within 1,000 meters of gun without being hit).

Note that due to the way that the target moves, the closer it gets to the gun, the easier it is to hit.

A difficulty factor of 10 is stored in register 4. This limits the direction change of the target to ± 5°. This may be changed to any real number; for example, to 0 to eliminate zig-zags, or to 20 for a more difficult game. The factor 10 is automatically restored to register 4 at the beginning of the next game.

Independently, the "EEX,3" in steps 62, 63 of the program controls the distance (0-1,000 meters) that the target moves towards the gun after each shell is fired. This may be replaced by, for example, "EEX,2" to make a much easier game.

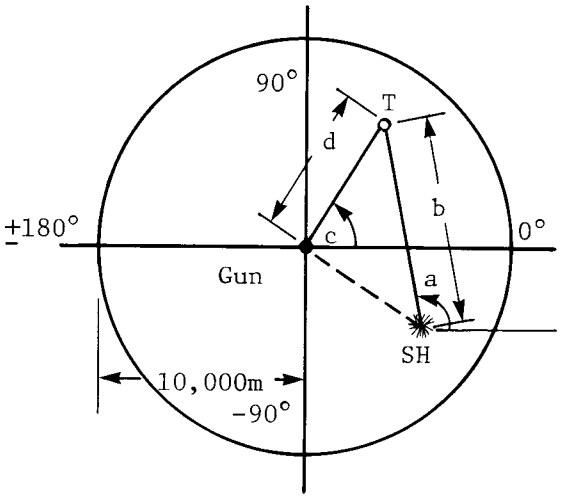
NOTE:

This program is adapted from HP-65 Users' Library program #03320A by Delmer D. Hinrichs.

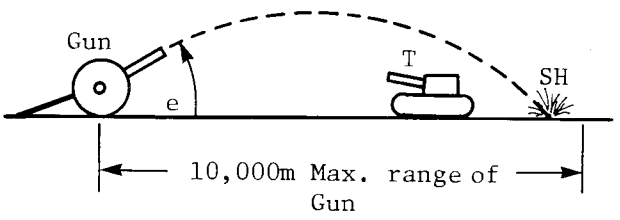
The range of the gun is given by the formula

Range = Sin 2θ • Maximum Range,

where θ = elevation angle



T = Target
SH = Shell Hit
a = Angle, shell hit to target
b = Distance, shell hit to target
c = Angle, gun to target
d = Distance, gun to target
e = Elevation, angle of gun



SOLUTION: (1)

0.00 GSB1	Enter seed of 0
90.00000 ENT↑	Locate target
45.00000 GSB2	
-75.14467 ***	Target was -75° and 14467 meters from shell impact
-45.00000 ENT↑	Shoot
20.00000 GSB2	
141.01832 ***	You're closer
-47.00000 ENT↑	
12.00000 GSB2	Shoot
92.00286 ***	
-46.00000 ENT↑	
10.00000 GSB2	
-66.00207 ***	
-47.00000 ENT↑	
10.00000 GSB2	
137.00527 ***	The target is getting further away from your shell
-48.00000 ENT↑	
7.50000 GSB2	
59.00192 ***	
-47.00000 ENT↑	
6.50000 GSB2	
125.00167 ***	Your shell hit only 167 meters away from target
-46.00000 ENT↑	
5.00000 GSB2	
0.00000 ***	(flashing)-Gun destroyed You lose

SOLUTION: (2)

63 2	2 is put in step 63 to improve chances of victory
5.00000 GSB1	Seed = 5
90.00000 ENT↑	
45.00000 GSB2	Find target
-142.02440 ***	Target location
102.74932 ENT↑	
30.01001 GSB2	Shoot
11.00588 ***	
98.85965 ENT↑	Shoot
29.70000 GSB2	
-176.00229 ***	Missed by 229 m.
100.37360 ENT↑	Shoot
29.58000 GSB2	
1.00000 ***	(flashing) Target destroyed You win

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1.	Key in the program			
2.	Enter seed (any number from 0 to 100			
	inclusive) and initialize.	Seed	GSB 1	10.00000
3.	Enter direction and elevation (both in	Direction	ENT↑	
	degrees) and fire.	Elevation	GSB 2	DDD.MMMMM
	Display shows direction and distance, shell			
	hit to target, in format: degrees. meters			
4.	Repeat step 3 until:			
4a.	Target is destroyed --flashing 1.			
4b.	Gun is destroyed--flashing 0.			
5.	For a new game, go to step 2.			

Program Listings

01 *LBL1		50 X≷Y	θ
02 DEG		51 INT	
03 CLRG	Initialize	52 X<0?	
04 STO7		53 GSB7	
05 3		54 ST+3	
06 6		55 RCL4	
07 0		56 GSB0	
08 GSB0	Starting angle	57 RCL4	
09 STO1		58 2	
10 5		59 ÷	
11 EEX		60 -	
12 3		61 ST+1	New angle
13 STO8		62 EEX	
14 GSB0		63 3	
15 RCL8		64 GSB0	1000
16 +		65 ST-2	
17 STO2	Starting distance	66 RCL2	New distance
18 1		67 EEX	
19 0		68 3	
20 STO4		69 X>Y?	New distance < 1000
21 FIX5	Set display	70 GTO8	
22 R/S		71 RCL3	
23 *LBL2	Elev. Dist.	72 R/S	Degrees • meters
24 2		73 *LBL8	
25 x		74 0	Display blinking 0
26 SIN		75 GTO9	
27 EEX		76 *LBL7	
28 4	10000	77 RCL3	
29 x		78 CHS	
30 →R	x_1y_1	79 STO3	
31 CLΣ		80 R↓	
32 Σ+		81 RTN	
33 0		82 *LBL6	
34 ST.0		83 1	Display blinking 1
35 RCL1		84 *LBL9	
36 RCL2		85 PSE	Blink generator
37 →R	x_2y_2	86 GTO9	
38 Σ-		87 *LBL0	
39 $\bar{x}$	x_2-x_1 y_2-y_1	88 RCL7	Random number generator
40 →P	rθ	89 Pi	
41 EEX		90 +	
42 2		91 X^2	
43 X>Y?	r < 100?	92 FRC	
44 GTO6		93 STO7	
45 R↓	r	94 x	
46 EEX		95 RTN	
47 5		96 R/S	
48 ÷			
49 STO3			

REGISTERS

0	1 Angle	2 Distance	3 Display	4 Δθ	5
6	7 Seed	8 5000	9	.0	.1
.2	.3	.4	.5	16	17
18	19	20	21	22	23
24	25	26	27	28	29

NOTES

In the Hewlett-Packard tradition of supporting HP programmable calculators with quality software, the following titles have been carefully selected to offer useful solutions to many of the most often encountered problems in your field of interest. These ready-made programs are provided with convenient instructions that will allow flexibility of use and efficient operation. We hope that these Solutions books will save your valuable time. They provide you with a tool that will multiply the power of your HP-19C or HP-29C many times over in the months or years ahead.

Mathematics Solutions
Statistics Solutions
Financial Solutions
Electrical Engineering Solutions
Surveying Solutions
Games
Navigational Solutions
Civil Engineering Solutions
Mechanical Engineering Solutions
Student Engineering Solutions

HEWLETT hp PACKARD

Reorder No. 00029-14006 Printed in U.S.A. 00029-90010