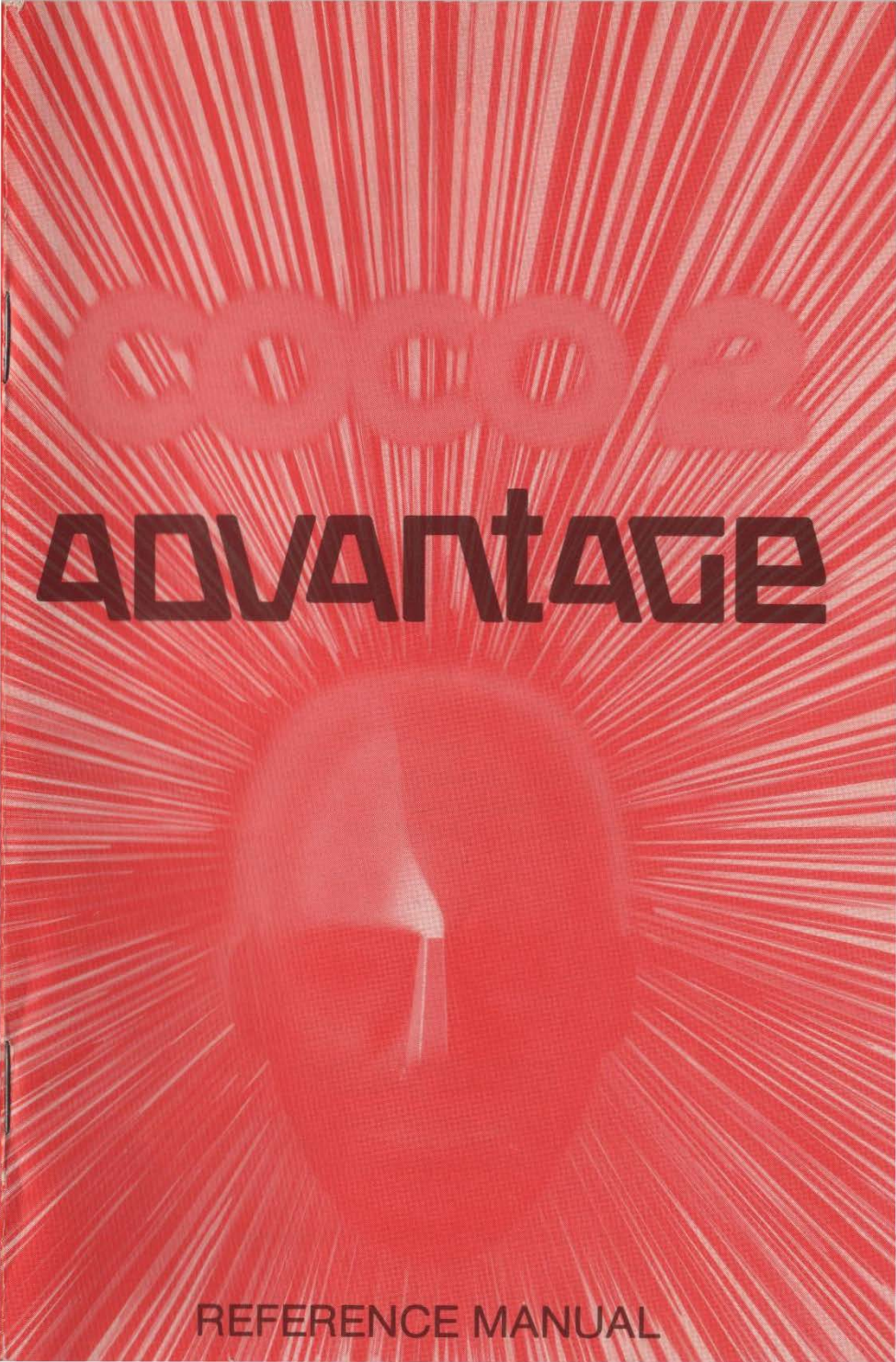

ADVANTAGE
REFERENCE MANUAL

HOW TO LOAD YOUR COCO 2 ON YOUR COMPUTER WITH CASSETTE RECORDER

1. Insert the Joystick Port Adaptor into Joystick Port #1. (Must be in at all times except when playing games or drawing objects with the Joystick.)
2. Insert the COCO 2 cassette into your cassette recorder and rewind the cassette.
3. Press SHIFT/RUN or type LOAD "COCO2", 1.1. Your computer will say PRESS PLAY ON TAPE.
4. Press only the PLAY key on the recorder. FOUND COCO 2 will appear on the screen.
5. If loaded with SHIFT/RUN, COCO 2 will begin to run by itself after it is finished loading.
6. If loaded with the LOAD statement, type RUN.
7. Press STOP on your cassette recorder when COCO 2 begins to run.
8. Be Patient. COCO 2 is a big program; it will take several minutes to load.

WITH DISKETTE DRIVE

1. Insert Joystick Port Adapter into Joystick Port #1. (Must be in at all times except when playing games or drawing objects with the Joystick.)
2. Insert your COCO 2 diskette into your drive and close the door.
3. Type LOAD "COCO2", 8,1.
4. After COCO 2 is loaded type RUN.

WARNING! TURN OFF COMPUTER BEFORE INSERTING THE JOYSTICK ADAPTOR

COCO 2™

REFERENCE MANUAL

Every effort has been made to ensure that this manual accurately documents the operation of COCO 2™. However, due to the ongoing improvement and update of the computer software, ACA cannot guarantee the accuracy of printed material after the date of publication, nor can ACA accept responsibility for errors or omissions. Revised manuals and update sheets will be published as needed and may be purchased by writing to:

ADVANTAGE

ADVANTAGE COMPUTER ACCESSORIES
1020 MEYERSIDE DRIVE, UNIT 8
MISSISSAUGA, ONTARIO
L5T 1J4

February 1983

TABLE OF CONTENTS

INTRODUCTION

WELCOME TO **COCO 2™!**

Why is this program called **COCO 2™**? Because there is a **COCO™**. The first **COCO™** teaches computer techniques, computer programming, a basic language and problem solving.

COCO 2™ allows you to create your own computer games. But **COCO 2™** really does much more. By the time you have played and programmed with **COCO 2™** you will have learned:

— Computer Graphics
— Computer Animation
— Real-Time Programming
— Computer-Human Interaction
— Logic

And you thought you were just having fun!

This manual is a very important part of **COCO 2™**. In it you will find all you will need to know to use **COCO 2™**. It is suggested that you read at least Section 1 before you try to use **COCO 2™**.

Whenever you load **COCO 2™** the sample game will be ready for you to play. You can play it, change it or erase it and write your own game.

COCO 2™ is menu driven. That means that there is a list of things you can do with **COCO 2™**; each thing has a number, and you simply select

the number of what you want to do and press return. In the first section of the manual you will learn about each of the functions of **COCO 2™**.

Section 2 contains the details of **COCO 2™**'s programming language.

Section 5 contains a sample game and the steps used to create a game.

You have seen hundreds of different games that are available to use at home on a television, or in a "hand held" game or in arcades. Each of these games uses different kinds of equipment but they have one major thing in common. They are all based on a microprocessor (computer) and therefore had to be programmed. **COCO 2™** will teach you how games work and how to program games to work the way you want them to.

There are many things you have to program in games that may not be obvious.

You have to draw the objects in the game, give them a color, give them sound, give them weapons. But this is only the beginning. You have to tell the objects how to act, how to move, how to shoot and how to react to each other. And there is more. How many objects does a player get before the game ends? How do you score and do you get an extra player during the game?

These are the things that **COCO 2™** will teach you while you are having fun doing it.

SECTION 1 MENU SELECTIONS

This section of the manual will present each of the menu selections, their purpose and how to use them. Each menu selection (except the programming language) is fully described here. Section 2 discusses the programming language.

COCO 2

1. DRAW OBJECTS
2. PROGRAM THE GAME
3. SET SCORING
4. SAVE/LOAD GAME
5. RUN GAME
6. DIRECTORY

SELECTION:

1.1 DRAW OBJECTS

PURPOSE

This function allows you to create (or change) the two objects in the game, one for you (the player) and one for your opponent (the computer). The object can be anything that you want; plane, tank, cannon, helicopter, boat, alien, spaceship, robot, car . . . anything. You have to decide the color of your objects, their sound, and where they are located on the screen when the game starts.

When you select this menu selection the following will display on your screen:

DRAW OBJECTS

1. PLAYER
2. OPPONENT

SELECTION:

Type '1' to draw the player object or '2' to draw your opponent.

Once you have selected the object you wish to draw, a screen will display that will allow you to create the object you have selected. In the following example is the screen to create the OPPONENT. The PLAYER screen is the same except for the heading.

OPPONENT

COLOR	: ***R***
SOUND	: ***S***
START POSITION X	: ***C***
START POSITION Y	: ***T***

These are the questions that you must answer about the object that you are drawing. **COCO 2™** will always display the questions and the last answer saved. If you have just loaded **COCO 2™** then the answers will be those of the sample game, as they are in this example.

You must answer these questions as follows:

COLOR : Select the color that you want for the object. You can choose the color by typing the first letter of the color. For example, to choose RED type R.
WHITE, RED, CYAN, PURPLE, GREEN, BLUE, YELLOW

SOUND : Select a sound for the object. Type the first letter of the sound you want.
TANK, PLANE, SAUCER, WALK, BUZZ, FUNNY, HELICOPTER, CAR

START POSITION X : In this selection you must specify the position (from left to right on your screen) you want the object to appear when you start a new game or after a previous object has been destroyed. Type the first letter of your selection.
CENTER, LEFT, RIGHT

START POSITION Y : Select the start position of the object (from top to bottom). Type the first letter of your selection.
TOP, MIDDLE, BOTTOM

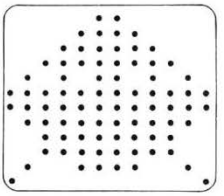
Once you have finished specifying these characteristics for the object you can now go on to the next step, drawing the object.

You can always come back and change anything about an object. If you have already created the object then the old characteristics will be displayed and you can change them.

As soon as you press RETURN the characteristics are saved in memory and the DRAW OBJECT screen will display.

DRAWING THE OBJECT

Once you have defined the characteristics of the object the following screen will display:



OPPONENT

You can move the black box (cursor) to any position by using the joystick or using the equivalent keys on the keyboard (see the table under MOVE instruction). To draw your object you simply turn the asterisks on or off using the fire button on the joystick or the space bar on the keyboard.

As you turn the asterisks on and off your object will appear below the asterisk drawing. Remember that if the color of your object is the same as the screen color then your object will be invisible.

You can modify an existing object or create a new one for a new game using this same method.

When you are finished drawing simply press RETURN and **COCO 2™** will save the object in memory and will keep it there until you save it on diskette or tape.

Note: The SHIFT and HOME/CLR keys together will clear the old object from the screen.

WARNING: If you turn off your computer before saving the game or press the ← key you will lose your object and its characteristics and will have to do it again.

1.2 PROGRAM THE GAME

PURPOSE

This selection allows you to program a game. You can program the actions of the player's object and the actions of the opponent's object. This section will introduce the idea of programming a game. Sections 2 and 3 contain more details on programming. Section 5 contains the details for the sample Game.

When you select ***PROGRAM THE GAME***, **COCO 2™** will display the following menu:

PROGRAM THE GAME

1. PLAYER
2. OPPONENT

SELECTION:

In order to program a complete game you must write a program for both the player and the opponent. Select '1' if you want to program the PLAYER or '2' to program the OPPONENT.

After you have indicated which program you want, the following screen will appear:

PROGRAM THE PLAYER

INSTRUCTION

_____ This is current instruction area.

_____

_____ The rest of your program will appear here.

ENT/CHG/DEL/NEW:

This is where you can enter new programs and change existing ones.

When you first load **COCO 2™** the sample game will also be loaded. If you want to clear that game and enter your own there are some commands you must know.

The ***'ENT/CHG/DEL/NEW'*** area is where you would specify the action you wish to perform. The following is the list of valid actions and their meaning. To use the command simply type the first letter of the word.

ENT This action tells **COCO 2™** that you wish to ENTER an instruction after the top line on the screen. You can enter up to 99 lines for each program.

CHG This action tells **COCO 2™** that you wish to CHANGE an instruction. When you type C you can change the instruction at the top of the screen.

The ***cursor control key allows you to scroll*** (move the next line of your program into the current area of the screen). The screen can only display part of a program at a time. The cursor key allows you to display more of your program. Type SHIFT and cursor to scroll your program down.

DEL This action tells **COCO 2™** that you wish to DELETE the instruction appearing at the top of the screen. The computer will make a warning sound if you try to delete and there are no lines to delete.

NEW This action tells **COCO 2™** to clear the program work area to make room for a new program.

WARNING: Be careful — this erases your current program.

COCO 2™ will check the command that you enter. If the command does not exist, **COCO 2™** will make a warning sound and wait for you to enter a valid command.

Press ← to return to the Main Menu.

WARNING: If you turn off your computer before saving the game, the program will be lost and you will have to enter it again.

NOTE: By Typing a ***P*** at the ***'ENT/CHG/DEL/NEW'*** area your program will be printed on your printer.

1.3 SET SCORING

PURPOSE

This selection allows you to define the scoring for your game. In addition you can define the number of player objects and the points awarded for a hit on the opponent.

The scoring is divided into 5 levels. The game progresses from level A through level E as the player accumulates points. Each level has associated with it a score limit, hit value, and an extra player indicator. As your score exceeds each limit you will be awarded the new ***HIT VALUES*** and ***EXTRA PLAYERS*** if they are allowed.

When you select this function the following screen will display.

The cursor will stop at each number that you can change. Type in your change and press RETURN to continue or just press RETURN if there is no change required.

SET SCORING

START # OF PLAYERS: 3

LVL	SCORE LIMIT	HIT VALUE	EXTRA PLAYER
A	0	5	N
B	100	5	Y
C	200	10	Y
D	300	15	Y
E	400	20	Y

When you first load **COCO 2™** this table already exists, and you can change it to the way you want it to appear for your game.

Look at each area of the table:

START # OF PLAYERS: Specify the number of objects the player gets at the beginning of the game. This is the number of objects that must be destroyed before the game ends. For example 3 means that the game will end when 3 player objects are destroyed.

SCORE LIMIT This is the lower limit of the scoring range. Level A always has a limit of zero. The highest limit is 9999. The score limit for each level must be higher than the previous level.

HIT VALUE This is the number of points that are awarded to the player for every hit on the opponent object. A different value can be used in each level. The lowest value is 1 and the highest is 20.

EXTRA PLAYER This indicates whether the player receives an extra object when he reaches this level. A 'Y' is indicated if an extra object is awarded; and 'N' if it is not. Level A always has an 'N'.

The example above shows that level B has a score limit of 100, a hit value of 5 and 'Y' for an extra object.

This means that (when playing a game with this kind of scoring) the player, when he exceeds a score of 100, will get 5 points for every hit and an extra object to use. The player will continue to get 5 points per

hit until he reaches level C, which in this case is 200 points. When he reaches level C, the player will then get 15 points for every hit and an extra object.

Later on in Section 2 you will see how you can check to see what level the player has achieved and make the game progressively more difficult and challenging for better players.

Each game that you create will have its own scoring table.

Press RETURN to return to the Main Menu.

WARNING: Remember to save the game before you turn your computer off or the table will be lost and you will have to create it again.

1.4 SAVE/LOAD GAME

PURPOSE

This selection allows you to save the games you have created onto diskette or cassette tape. When you select ***SAVE/LOAD GAME*** the following menu will appear.

SAVE/LOAD GAME

1. SAVE GAME
2. LOAD GAME

SELECTION:

Select '1' if you want to save a game you have just written or one that you have just changed. Select '2' to load a game that you have previously saved.

If you select ***SAVE*** the following screen will appear.

SAVE GAME

DISK, TAPE? :
NAME :

Answer the questions as follows:

DISK/TAPE? : Type 'D' for disk or a 'T' for tape depending on what type of storage unit you have.

NAME: Type up to a 6 character name that you want to call this game. When you load this game again **COCO 2™** will find it by its name.

Remember if you don't save your game before you turn off the computer, the game will be lost. The SAVE feature allows you to build a complete library of games that you can load and play any time.

When you save a game **COCO 2™** will save the Opponent Program, Player Program, Score Table and the two objects under the game name.

WARNING: If you are using a diskette drive and you save a game with a name you have already used then the game will not be saved. If you are using cassette tape, it is recommended that you do not save more than one game on each side of the cassette.

If the SAVE was successful **COCO 2™** will display:

PRESS RETURN TO CONTINUE

If the SAVE was not successful **COCO 2™** will display:

(ERROR MESSAGE IF DISKETTE DRIVE IS USED) PRESS RETURN TO CONTINUE

If you select ***LOAD*** the following screen will display.

LOAD GAME

DISK, TAPE :

NAME :

Answer the questions as follows:

DISK, TAPE : Type 'D' if you have a disk drive, a 'T' if you have a cassette drive.

NAME: Type the name (up to 6 characters) of the game that you wish to load or press return to load the first game.

If the game you have tried to load has not been saved then **COCO 2™** will display an error message. If you get this message check your spelling, you may have made a mistake.

If you have forgotten the name of the game you want to load read about the game DIRECTORY in Section 1.6.

COCO 2™ will display 'LOADED RETURN TO CONTINUE' if the Load was successful and 'NOT LOADED' if the Load was not successful. If you are using a diskette drive then an error message will be displayed.

Press RETURN, ESC or ← to return the Main Menu.

WARNING: You will lose the current game if you LOAD a new one.

1.5 RUN THE GAME

PURPOSE

This is the selection that you will probably use the most. This function really has two purposes. One is to play the games you have programmed. The other allows you to run the game in a monitor mode. This means that you can execute the game, watch the program instructions display on the screen and run the game step by step to analyze your program in detail.

The monitor is both a learning tool and debugging device. When you are in the monitor mode the program instructions will be displayed at the bottom of the screen as they are executed.

You can run either the opponent program through the monitor or the player program, or both.

In a normal play mode you can play your games using either a joystick or the keyboard. These methods of play are completely interchangeable and you can use either at any time. See Section 2 for the equivalence between the keyboard and the joystick.

When you select ***RUN THE GAME*** the following screen will display:

RUN THE GAME

1. PLAY
2. MONITOR

SELECTION:

If you select '1' for PLAY then the screen will clear and the game will begin.

If you select '2' for MONITOR then you must answer the following questions:

STEP-BY-STEP? If you answer ***'Y'*** then the monitor will execute your game program step by step stopping after each instruction to allow you to examine instruction and the effect. When you are in the monitor you must press return to execute the next instruction.

If you answer ***'N'*** then the monitor will execute the game without stopping but the program instructions will be displayed.

PLAYER, OPPONENT, BOTH? When you select monitor, **COCO 2™** wants to know if you want to monitor the player program (type ***P*** for player), the opponent program (type ***O*** for opponent) or both (type ***B*** for both programs).

To return to the Main Menu press the ← key or the ESC key (some computers have an ESC and some have an ←).

1.6 DIRECTORY

PURPOSE

This selection allows you to keep track of the games you have saved on diskette.

When you select ***DIRECTORY***, **COCO 2™** will search the diskette and display the names of your games on the screen. If there are more names than can appear on the screen at one time type 'Y' in answer to the MORE? question.

Press RETURN, ESC or ← to return to the MAIN MENU.

SECTION 2 PROGRAMMING LANGUAGE REFERENCE

This section presents the **COCO 2™** programming instructions. With one exception, (MISSILE) all instructions can be used in the player program or the opponent program.

A program consists of a group of instructions. Each instruction is assigned a number (the first instruction is #1 and the second is #2 and so on). When the game is played the instructions are executed in the order of their number, unless control of the program is transferred to another line number. The GOTO, PERFORM and IF instructions can change the flow of a program.

The instructions are of 3 basic kinds:

1. ***Object Movement Instructions***

   These instructions are used to move both the player and the opponent object around on the screen and to control their speed.

2. ***Attack Instructions***

   These instructions can be used to fire on and destroy objects.

3. ***Program Control Instructions***

   These instructions control the flow of execution of the program.

This section will also discuss the use of 4 very special things. These things are variables. A variable is given different values at different stages of the program. The four variables are SCORE, R, V and J.

SCORE: This variable keeps track of the player's score in the game. As the player makes hits on the opponent the score is increased.

R: This is a random number generator. This means that whenever you use R, the computer will select a random number for you. The computer will use a different range of numbers for selecting the random number depending on how you use R.

For example, if you used R to generate a direction in a movement instruction then R will be a number from 1 through 9. If you used R in a GOTO instruction, (or where R means an instruction #) R can be from 1 through 99.

V: This is a general purpose variable. You give V a value by using it in a "LET V = " instruction. V will keep the same value until you change it again in your program.

J: This variable is set by the joystick or keyboard. When it is used **COCO 2™** will go and check the joystick (or the keyboard) for the direction of movement. In this way you can control the direction of movement of the objects. J can also be used to designate a distance in the MOVE instruction or even an instruction # in the program. J can have a value of 1 through 9. See the MOVE and IF J instructions for more information.

If you use J in an instruction and neither the joystick is activated nor the equivalent key is pressed on the keyboard, then that instruction will be ignored.

OBJECT MOVEMENT INSTRUCTIONS

You have the capability of moving objects anywhere on the screen. You will have to program these movements. These movements are either direct program movements or program response to the joystick/keyboard. The player object will usually be controlled by the joystick or keyboard.

The objects always move four pixels at a time. What's a pixel, you may ask? A pixel is part of a character. Each letter or number displayed on a screen is made up of pixels. The computer makes characters by turning the pixels on or off in a specific pattern. The following diagram shows how a computer would make the letter F.

Each square in this diagram is a pixel. Some of them have been turned on to make the letter F.

Your object, when it is instructed to move, will move at least four pixels at a time. Since a pixel is such a very small distance your object will move very smoothly across the screen. That is ***ONE*** [***HALF A***] ***CHARACTER*** at a time.

When you use the movement instruction you must specify the number of [half] characters to move.

Sometimes you may want to program the movements of objects so that they move randomly. To get a random number use the letter R in the instruction instead of a number. Every time an instruction using R is executed, R will have a new random value and this way you will never be able to predict the movement of objects.

You can control the distance that the object will move by using J variable and then get the distance from the joystick during the game. You can set the variable 'V' to a value and then use V in your movement instruction, or use 'R' to generate a random distance.

You must also specify the direction of movement. You can do this by specifying the direction using a number or by checking the direction of the joystick.

The movement instructions can be used in both the opponent program and the player program.

MOVE

Function This instruction moves the object in a straight line in the specified direction for the specified number of pixels. The direction and distance can be selected by number, determined by the joystick or can be generated randomly.

Syntax MOVE ***z n*** (***z*** can be 1-9, V, R or J)
(***n*** can be 1 through 99, V, R or J)

z specifies the direction of the movement. The directions are as follows:

Value	Direction	Keyboard
1	DOWN/LEFT	C
2	DOWN	V
3	DOWN/RIGHT	B
4	LEFT	D
5	TRACK	SPACE
6	RIGHT	G
7	UP/LEFT	E
8	UP	R
9	UP/RIGHT	T

You can use R to generate a random direction or assign V a value and use it in place of z in the MOVE instruction.

You can also use the joystick to control the direction of the MOVE.

The variable J, when used in the MOVE, will be assigned a value by the joystick. The Value column in the table above indicates the relationship between the directions and the numbers 1 through 9.

A value of 5 will cause the object to track the other object. This means that you can 'aim' your object and follow the opponent or the opponent can follow the player.

n specifies the number of [½] characters [4 pixels] you want the object to move.

Example MOVE 1 5 This instruction causes the object to move diagonally down and to the left, five [½] characters [20 pixels].

MOVE 4 12 This instruction causes the object to move left, 12 characters [48 pixels].

MOVE 8 J This instruction causes the object to move up and it will get the distance to move from the joystick. If the joystick is pushed DOWN then the object will move 2 characters [8 pixels] (see table above).

MOVE R R This will cause the object to move in a random direction for a random distance.

MOVE J 5 This will cause the object to move in the direction indicated by the joystick (UP on the joystick will cause the object to move up) for a distance of 5 characters [20 pixels].

SPEED

Function This instruction sets the speed of the object.

Syntax SPEED ***n*** (***n*** can be 1-9 or V, R or J, where 1 is slow). If speed is specified by J, the user can change the speed with the joystick.

Example SPEED 9 Specifies that this program is to execute at the highest speed.

ATTACK INSTRUCTIONS

These instructions are used to destroy the objects. These instructions cause the object to fire its weapons. The firing of the opponent's weapons will be controlled in the program. Firing by the player can be programmed and controlled by the keyboard or joystick fire button. An object (Player or Opponent) cannot have more than 5 items (bombs or shots) on the screen at any time.

MISSILE

Function This instruction can only be used in the opponent's program. It is not valid in the player's program. This instruction will cause a missile to be launched by the opponent. The missile will pursue the player until the missile is destroyed. The missile cannot leave the screen. There can only be one missile on the screen at one time. If another MISSILE command is encountered before a previous missile is destroyed then the instruction will be ignored.

Syntax MISSILE

SHOOT

Function This instruction causes a shot to be fired. The shot can be destroyed by another shot or bomb and when it goes off the screen it is gone. This instruction can be used by both the player and the opponent. The direction of the shot can be specified by a number from 1 through 9 or it can be random (R) on a specified value (V). It can also be directed by the joystick. See the MOVE instruction for a definition of directions.

Syntax SHOOT ***n*** (where ***n*** specifies a direction and can be 1-9, R, V or J)

Example SHOOT 1 (will fire a shot to the left and down.)
SHOOT J (will fire a shot in the direction specified by the value in J. J is set by the joystick.)

BOMB

Function This instruction causes a bomb to be released. The bomb explodes on impact with another object and when it goes off the screen it is gone. The direction of the bomb can be specified by a number from 1 through 9 or it can be random (R) or a specific value (V). It can also be directed by the joystick. See the MOVE instruction for a definition of directions.

Syntax BOMB ***n*** (where ***n*** specifies a direction and can be 1-9, R, V or J)

Example BOMB 5 (will release a bomb in the direction of the other object at the time of release.)
BOMB J (will fire a shot in the direction specified by the value in J. J is set by the joystick.)

PROGRAM CONTROL INSTRUCTIONS

These instructions control the flow of your program. You can check values of SCORE, R or J and change the action within your program. These instructions control the logic of the program. Using these instructions you can program 'smart' games; games that can react to situations and are not just mindless shoot-em-ups.

LET V =

Function This instruction allows you to set variable V to specific values.

Syntax LET V = ***x*** (where ***x*** can be 1-99 or V, R or J)

Example LET V = 5 (this gives V a value of 5 until you change it.)

PERFORM

Function This instruction allows you to execute, from different places in your program, a group of instructions specified a number of times. Control will be passed on to the line number specified in the PERFORM and when a RETURN is encountered in the PERFORM group, control will return to the instruction following the PERFORM. You can have up to

five PERFORM instructions before a RETURN is required.

Syntax PERFORM ***z x*** (where ***z*** represents the number of times the PERFORM is to be done and can be 1-9, V, R or J)
(where ***x*** is an instruction # and can be 1-99 or V, R or J)
NOTE: 'V' can have a value of up to 99 in the PERFORM.

Example PERFORM 5 20 (will PERFORM the group of instructions [beginning at instruction #20] 5 times.)

```
•
•
•
05 PERFORM 2 10      In this example the PER-
•                    FORM will execute the group
•                    of instructions beginning at
•                    instruction #10, twice.
•
10 MOVE 5 J          This is a PERFORM group
11 BOMB J            which can be executed from
12 RETURN            any place in the program.
•
•
```

RETURN

Function **This instruction must appear at the end of a PERFORM group. Control will be returned to the instructions after the PERFORM.**

Syntax **RETURN**

GOTO

Function Transfer control to the instruction number specified in the GOTO.

Syntax GOTO ***x*** (***x*** can be 1 through 99, or V, R or J)

Example GOTO 30 (will transfer program control to instruction #30.)

IF J =

Function This instruction checks the joystick, which assigns the value of J, (if there is no joystick the keyboard will be checked). If the joystick value is the same as the expression after the equal sign (***z***) then control will be transferred to the instruction number specified by ***x***.

Syntax If J = ***z x*** (***z*** can be 1-9, V, R or J)
(***x*** can be 1-99, V, R or J)

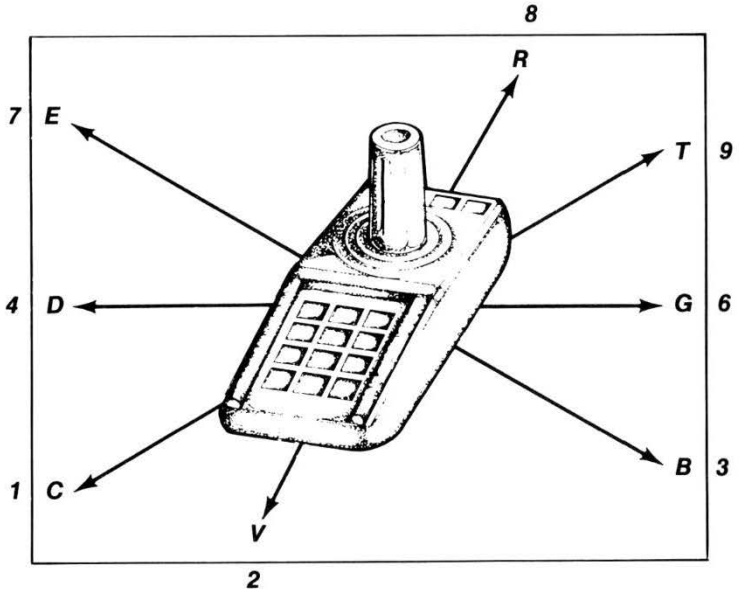
Example IF J = 5 20 (control will be transferred to instruction #20 if the value entered by the joystick is 5.)

The following table shows the values of the joystick, the corresponding action and the keyboard equivalent.

POSSIBLE VALUES OF J	JOYSTICK POSITION	KEYBOARD EQUIVALENT
1	DOWN/LEFT	C
2	DOWN	V
3	DOWN/RIGHT	B
4	LEFT	D
5	***FIRE***	***SPACE BAR***
6	RIGHT	G
7	UP/LEFT	E
8	UP	R
9	UP/RIGHT	T

The following diagram shows the relationship between the keyboard and the positions of the joystick. Notice that when you hold the joystick in the UP position then J will have the value of 8 and you can simulate the same thing by typing R on the keyboard.

The number outside the box represents the values of J when the joystick is held in the corresponding direction. Inside the box are the letters of the keyboard that have the same result.



IF SCORE =

Function This instruction will check the value of the score (according to the levels you set up in ***SET SCORING***) and transfer control to the instruction specified if the expression is true.

Syntax IF SCORE = ***y*** ***x*** (***y*** can be B, C, D, E [levels]), (***x*** can be 1-99, V, R or J)

Example IF SCORE = B 20 (This statement says that if the player's score is at level B then transfer program control to instruction 20.)

IF R =

Function This instruction generates a random number in R and compares it to the number after the equal sign (***z***). If the expression is true then control is transferred to the instruction specified by ***x***.

Syntax IF R = ***z x*** (***z*** can be 1-9, V, R or J)
(***x*** can be 1-99, V, R or J)

Example IF R = 1 10 (This statement says that if the value of R [the random number] is equal to 1 then transfer control to instruction 10.)

IF R = V 10 (If R is equal to the value in V then control is transferred to instruction 10.)

IF R >

Function This instruction generates a random number in R and compares it to the number after the greater than sign (***z***). If the expression is true then control is transferred to the instruction specified by ***x***.

Syntax IF R > ***z x*** (***z*** can be 1-9, V, R or J)
(***x*** can be 1-99, V, R or J)

Example IF R > 5 20 (This statement says that if R is greater than 5 go to line 20. If it is not, go to the next instruction.)

SECTION 3
PROGRAMMING EXAMPLES

Example 1 ***Shooting Gallery***

Before a game can be created, it is helpful to know what kind of game is to be created.

For example, consider a game which might be called "SHOOTING GALLERY". In order to create a shooting gallery, you need something to shoot with (a gun) and a target to shoot at.

The target can move back and forth across the top of the screen. Targets do not shoot back, so this program will not have any attack commands for the opponent.

The gun can sit at the center of the bottom of the screen. The gun does not need to move, but it needs to shoot. The fire button on the joystick will be used to tell the gun to fire.

Next, the game needs a scoring table. In this sample game, the various score levels will not be used. In addition, the opponent cannot destroy the player, so extra players are not necessary. An example of a score table might be as follows:

LVL	SCORE LIMIT	HIT VALUE	EXTRA PLAYER
A	0	5	N
B	200	5	N
C	400	5	N
D	600	5	N
E	800	5	N

Now that the game is defined, the computer must be told how to run the game. The first step is to load and run the **COCO 2™** program. Use selection 1 (DRAW OBJECTS) to draw the opponent (target) and player (gun). The target can be set to start in the upper left hand corner and the gun can start in the lower center of the screen.

Next, selection 3 is used to define the score table.

Selection 2 is used to enter programs to tell the computer what the target and the gun can do. The following programs are examples:

OPPONENT PROGRAM (TARGET)

1. MOVE 6 21 Move across the top of the screen to the right corner.
2. MOVE 6 21

3. MOVE 4 21 Move back to the top left corner.
4. MOVE 4 21

5. GOTO 1 Go back to the beginning of the program and start over.

PLAYER PROGRAM (GUN)

1. IF J = 5 3 If the fire button is pressed, GOTO statement 3.

2. GOTO 1 Go back to the beginning.

3. SHOOT 8 Fire a shot straight UP.

4. GOTO 1 Go back to the beginning.

Now the computer knows how to play the "SHOOTING GALLERY" game. At this point, the game can be played by using selection 5, but it is best to save the game first so that it is not lost when the computer is turned off. Use selection 4 if you want to save the game.

Example 2 ***Sentry***

Next, consider a more difficult game about a robot sentry and a spy. Like the target in "SHOOTING GALLERY", the robot is going to move back and forth across the top of the screen. However, this program will allow the robot to move at different speeds to make him harder to hit.

This program is going to let the spy move also. In addition, the spy will throw a bomb occasionally instead of shooting when the fire button is pressed.

As in the previous example, the first thing to do is to use selection 1 to draw the player and opponent objects. In this case, the player is the spy and the opponent is the robot.

Next, selection 3 is used to set up the scoring table. The following example might be used:

LVL	SCORE LIMIT	HIT VALUE	EXTRA PLAYER
A	0	1	N
B	10	5	N
C	50	10	N
D	1000	15	N
E	2000	20	N

Once this is done, selection 2 is used to enter the player and opponent programs.

OPPONENT PROGRAM

1. ***IF SCORE = E 14***
2. ***IF SCORE = D 11***
3. ***LET V = 4*** Set up for level C.
4. ***IF SCORE = C 8***
5. ***LET V = 2*** Set up for level B.
6. ***IF SCORE = B 8***
7. ***LET V = 1*** Level A, slow speed.
8. ***SPEED V*** Set speed.
9. ***PERFORM 1 23*** Subroutine 23, once.
10. ***GOTO 1*** Loop for levels A, B and C.
11. ***SPEED 6*** High speed.
12. ***PERFORM 1 23*** Subroutine 23.
13. ***IF SCORE = D 11*** Goto 11 if at D.
14. ***IF R > 3 20*** Select a random speed for the robot.
15. ***IF R > 3 18***
16. ***SPEED 9*** Highest Speed.
17. ***GOTO 21***
18. ***SPEED 8***
19. ***GOTO 21***
20. ***SPEED 7***
21. ***PERFORM 1 23*** Subroutine 23.
22. ***GOTO 14*** Goto 14 and start over.
23. ***MOVE 6 21***
24. ***MOVE 6 21***
25. ***MOVE 4 21***
26. ***MOVE 4 21***
27. ***RETURN***

PLAYER PROGRAM

1. ***IF J = 5 4*** Check for fire button.
2. ***MOVE J 1*** Move in the direction of the joystick.
3. ***GOTO 1*** Go back and check again.

4. ***IF R > 6 7*** Choose between shoot and bomb.
5. ***SHOOT 8*** Shoot up.
6. ***GOTO 1***
7. ***BOMB 8*** Bomb up.
8. ***GOTO 1***

Now that this game is finished, it may be saved or played as explained in the "SHOOTING GALLERY" game.

Example 3 ***Kamikaze***

Another example is the game of "KAMIKAZE" between a plane and a battleship. The player's object is the battleship and shoots when the fire button is pressed. The shot is automatically aimed toward the plane. In addition, holding the joystick in one of the up positions (7, 8 or 9) fires a bomb in the direction of the joystick. Moving the joystick left or right moves the battleship left or right.

The plane chases the battleship and attempts to ram it. In addition, the plane moves faster as the score level increases.

To create this program:

1. Draw the opponent object (PLANE). It should start in upper left corner.
2. Draw the player object (BATTLESHIP). It should start in the center of the bottom of the screen.
3. Enter the scoring table. An example might be:

LVL	SCORE LIMIT	HIT VALUE	EXTRA PLAYER
A	0	1	N
B	10	3	N
C	50	5	N
D	100	10	Y
E	1000	20	Y

4. Enter the opponent program.
   1. ***SPEED 1***
   2. ***IF SCORE = B 8***
   3. ***IF SCORE = C 10*** Set the speed based on the score.
   4. ***IF SCORE = D 12***
   5. ***IF SCORE = E 14***
   6. ***MOVE 5 1*** Move toward the ship.
   7. ***GOTO 2***
   8. ***SPEED 3***
   9. ***GOTO 6***
   10. ***SPEED 5***
   11. ***GOTO 6***
   12. ***SPEED 7***

13. GOTO 6

14. SPEED 9

15. GOTO 6

5. Enter the Player program.

1.	***IF J = 5 8***	Check for a shot.
2.	***IF J = 4 10***	Check for movement.
3.	***IF J = 6 10***	
4.	***IF J = 7 12***	Check for a bomb.
5.	***IF J = 8 12***	
6.	***IF J = 9 12***	
7.	***GOTO 1***	Ignore other joystick positions.
8.	***SHOOT 5***	Shoot at plane.
9.	***GOTO 1***	
10.	***MOVE J 1***	Move in joystick direction.
11.	***GOTO 1***	
12.	***BOMB J***	Fire a bomb in joystick direction.
13.	***GOTO 1***	

6. Save the game.
7. Play the game.

Example 4 ***B-1 Attack***

In the game of "B-1 ATTACK", the player's object is a B-1 Bomber and the opponent controls a fortress on the ground. To make it more interesting, the fortress is allowed to move along the bottom of the screen. As the score level increases, the fortress can fire more weapons and move faster.

Pressing the fire button on the joystick causes the B-1 Bomber to fire in the direction it is traveling. The B-1 can also drop a bomb by holding the joystick down, down and right or down and left. The bomber always moves from left to right across the screen. The B-1 is steered by holding the joystick to the right or left positions.

To create this program:

1. Draw the opponent object (FORTRESS). It should start in the center of the bottom of the screen.
2. Draw the player object (B-1 BOMBER). It should start in the upper left corner of the screen.
3. Enter the scoring table. An example might be:

LVL	SCORE LIMIT	HIT VALUE	EXTRA PLAYER
A	0	1	N
B	10	3	Y
C	100	5	Y
D	500	10	Y
E	2000	20	Y

4. Enter the opponent program.

```
1.  SPEED 2
2.  IF R > 4 5          Choose between moving left or moving
3.  MOVE 4 R            right by a random distance.
4.  GOTO 6
5.  MOVE 6 R
6.  IF SCORE = B 13     Select action to be taken based on score
7.  IF SCORE = C 11     level.
8.  IF SCORE = D 16
9.  IF SCORE = E 15
10. IF R > 4 2
11. SHOOT 8             Shoot upwards.
12. GOTO 2
13. IF R > 7 2
14. GOTO 11
15. MISSILE             Fire a missile.
16. BOMB 5              Bomb toward the B-1.
17. SHOOT 8             Shoot upwards.
18. GOTO 2
```

5. Enter the player program.

```
1.  LET V = 6           Set up the initial direction for the B-1
                        bomber.
2.  IF J = 5 14         Check for a shot.
3.  IF J = 1 12         Check for a bomb.
4.  IF J = 2 12
5.  IF J = 3 12
6.  IF J = 4 10         Check for a move.
7.  IF J = 6 10
8.  MOVE V 5            Move in the last indicated direction.
9.  GOTO 2
10. LET V = J           Set new indicated direction.
11. GOTO 8
12. BOMB 2              Bomb down.
13. GOTO 8
14. SHOOT 5             Shoot in direction the B-1 is flying.
15. GOTO 8
```

6. Save the game.
7. Play the game.

SECTION 4 GAME EXECUTION

Writing the programs for the PLAYER and the OPPONENT is very much like directing a play or a movie. You as the director have to visualize every move of both sides. Remember that many of the moves of the PLAYER will be controlled by the joystick but the OPPONENT will be entirely under computer control.

Once you have written both programs and you decide to play the game, the computer executes both programs.

The computer seems to run both programs at the same time. But this only seems to happen, the computer can only do one thing at a time.

There are a few rules that you have to know about how **COCO 2™** executes your programs. These rules will be followed in both the PLAY mode and the MONITOR mode.

COCO 2™ will execute the program in the following sequence:

A. Up to five instructions from the opponent program or until a movement instruction is encountered.

B. Opponent missile will be fired if the MISSILE instruction has been used.

C. Opponent shots will be fired if the SHOOT instruction has been used.

D. Opponent bombs will be released if the BOMB instruction has been used.

E. Up to five PLAYER instructions unless a movement is detected.

F. PLAYER shots will be fired if the SHOOT instruction is detected.

G. PLAYER bombs will be released if the BOMB instruction has been used.

Steps A, B and E will be executed multiple times depending on the relation in speed between the player and the opponent. For example if the OPPONENT is to be twice as fast as the PLAYER then A and B will be executed twice as many times as E.

After any movement by any of the objects (Player, Opponent, Shots, Bombs and Missiles) **COCO 2™** will automatically check for any collisions. If a collision is detected all objects involved will explode. If the opponent object is involved in a collision then the player score will increase. If the player object is involved in a collision then the player will lose an object.

Objects that are not involved in a collision will continue on course. A maximum of five items can be on the screen for each object. For example, the player can have a bomb and four shots active on the screen at one time and so can the opponent.

When the last player object is destroyed the game is over.

To return to the Main Menu at the end of the game type any key or activate the joystick.

SECTION 5 I.T. AND THE UFO

COCO 2™ comes with a preprogrammed game. Whenever you load **COCO 2™** the sample game is available for you to play, change or erase and start over.

All of the components of the game have been created for you.

When you develop your own games you have to do the following things.

1. Draw PLAYER object
2. Draw OPPONENT object
3. Set score table and difficulty levels
4. Program the PLAYER
5. Program the OPPONENT
6. SAVE the GAME if you want to keep it.

Let's analyze the sample game now so that you can see exactly how **COCO 2™** works. Here is a summary of the game.

THE OBJECT OF THE GAME

This game is a battle between ***I.T. the "Intra-Terrestrial" and the UFO***.

The object of the game is for I.T. to destroy the UFO. As the score increases the UFO becomes more dangerous and harder to destroy. But don't worry, I.T. gets more weaponry, moves faster and even begins to fly as your score goes up. You get more I.T.'s to play with when your score goes up.

Oh, by the way. **Watch out for the guided missiles!**

PLAYER PROGRAM SUMMARY

1. The speed varies with the level of difficulty. As the player's score reaches each higher level (see SET SCORING) the game speeds up (refer to program instructions 1 through 9).
2. During difficulty levels A and B the player can move left and right only and he cannot release a bomb (refer to program instructions 10 through 21).
3. During difficulty levels C and D the player can move in any direction and the shot or bomb will be released at random when the fire button is depressed (refer to program instructions 22 through 30).
4. During difficulty level E the player can move in any direction and can only shoot (refer to program instruction 25).

OPPONENT PROGRAM SUMMARY

1. Speed varies with the level of difficulty attained. This is done by setting the variable "V" to a value of 1 through 6 depending on the score.
2. Difficulty level A - the opponent can move left or right randomly (refer to program instructions 54 through 57).

3. Difficulty level B - the opponent moves left or right and shoots after every 9 moves (refer to program instructions 6 through 10).
4. Difficulty level C - the opponent releases a bomb, moves, left or right randomly 8 times and shoots (refer to program instructions 11 through 17).
5. Difficulty level D - same as level C, but with more randomness (refer to program instructions 18 through 24).
6. Difficulty level E - releases a shot, and then totally randomly either moves, releases a bomb, shoots or releases a missile (refer to program instructions 25 through 42).
7. Note that the logic between program instructions 43 through 53 is used over and over again by the levels of difficulty A through D.

The following is a listing of the ***PLAYER*** program and a description of the action.

INSTRUCTION	EXPLANATION
1. LET V = 9	Give V a value of 9, set up player at level E.
2. SPEED V	Set speed according to score level.
3. IF SCORE = E 22	Check the player's score level, if level E go to instruction 22.
4. LET V = 8	Set up for level D.
5. IF SCORE = D 22	Check the player's score level, if level is D go to instruction 22.
6. LET V = 6	Set up V with a value of 6 for level C.
7. IF SCORE = C 22	If player is at level C go to instruction 22.
8. LET V = 4	Set up V with a value of 4 for level B.
9. IF SCORE = B 11	If player is at level B go to instruction 11.
10. LET V = 2	Set up V with a value of 2 for level A.
11. IF J = 5 15	Go to 15 if fire button is active.
12. IF J = 4 20	Go to 20 if joystick direction is left.
13. IF J = 6 20	Go to 20 if joystick direction is right.
14. GOTO 11	Joystick is inactive or invalid.
15. IF SCORE = B 18	If B, allow directional shots.
16. SHOOT 8	Else shoot up only.
17. GOTO 2	
18. SHOOT J	Directional shot.
19. GOTO 2	
20. MOVE J 1	Move player in joystick direction.
21. GOTO 2	
22. IF J = 5 25	Go to 25 if fire button is activated.

23.	***MOVE J 1***	Move player in joystick direction.
24.	***GOTO 2***	
25.	***IF SCORE = E 27***	On E only shoot (no bombs).
26.	***IF R > 7 29***	Allow a bomb (odds 2:9).
27.	***SHOOT J***	Shoot in joystick direction.
28.	***GOTO 2***	
29.	***BOMB 5***	Release bomb in direction of opponent.
30.	***GOTO 2***	

This is the end of the PLAYER program.

The following is a listing of the ***OPPONENT*** program and an explanation.

INSTRUCTION		EXPLANATION
1.	***IF SCORE = E 25***	Logic for level E.
2.	***IF SCORE = D 18***	Logic for level D.
3.	***IF SCORE = C 11***	Logic for level C.
4.	***IF SCORE = B 6***	Logic for level B.
5.	***GOTO 54***	Logic for level A.
6.	***LET V = 2***	Used for speed and movement.
7.	***SPEED V***	
8.	***PERFORM 9 49***	Move randomly left or right 9 times.
9.	***SHOOT 2***	Shoot downwards.
10.	***IF SCORE = B 8***	Continue with level B logic.
11.	***LET V = 3***	
12.	***SPEED V***	
13.	***MOVE 2 V***	Move opponent down 3 characters [12 pixels].
14.	***BOMB 2***	Drop a bomb downwards.
15.	***PERFORM 8 43***	Move randomly left or right 8 times.
16.	***SHOOT 5***	Shoot in direction of player.
17.	***IF SCORE = C 13***	Continue with level C logic.
18.	***LET V = 4***	
19.	***SPEED V***	
20.	***MOVE R V***	Move in a random direction.
21.	***BOMB R***	Release a bomb in a random direction.
22.	***PERFORM 5 43***	Move randomly left or right 5 times.
23.	***SHOOT 5***	Shoot in direction of player.
24.	***IF SCORE = D 20***	Continue with level D logic.
25.	***LET V = 6***	Setup for level E.
26.	***SPEED 7***	
27.	***MOVE R R***	Move in a random direction, a random # of characters [pixels].

28. SHOOT 2	Level E immediately releases a shot down.
29. IF R > 4 36	Move or shoot (odds 5:9).
30. IF R = 1 41	Go release a missile (odds 1:9).
31. IF R > 3 34	Bomb or shoot (odds 1:3).
32. BOMB R	Release a bomb in a random direction.
33. GOTO 29	
34. SHOOT 5	Shoot in direction of player.
35. GOTO 29	
36. IF R > 1 39	Track or random move (odds 1:9).
37. MOVE 5 R	Track player at a random # of characters [pixels].
38. GOTO 29	
39. MOVE R V	Move in a random direction.
40. GOTO 29	
41. MISSILE	
42. GOTO 29	
43. IF R > 5 49	Move LEFT more often than RIGHT (odds 5:9).
44. IF R > 7 47	Move RIGHT or LEFT (odds 7:9).
45. MOVE 6 V	Move RIGHT, variable # of characters [pixels].
46. RETURN	Return to instruction after the PERFORM.
47. MOVE 4 V	Move LEFT, variable # of characters [pixels].
48. RETURN	
49. IF R > 7 52	Move LEFT or RIGHT (odds 7:9).
50. MOVE 4 V	
51. RETURN	
52. MOVE 6 V	
53. RETURN	
54. LET V = 1	Level A logic.
55. SPEED V	
56. PERFORM 1 44	
57. GOTO 1	

This is the end of the OPPONENT program.

We have tried to use all of **COCO 2™**'s instructions in these programs so that you can see how they work.

You should now run these programs step-by-step with the monitor and examine them in detail so that you can be completely familiar with the language.

The better your understanding of the language the faster you can write your games and move on to playing.

[ ] USE FOR VIC 20 ONLY

SECTION 6 ERROR MESSAGES

COCO 2™ is an interactive program. That means that the program responds to your commands or instructions immediately.

Wherever possible **COCO 2™** diagnoses errors and displays an error message as soon as you make an error. This technique ensures that you get results from **COCO 2™** as soon as possible.

There are a few errors that **COCO 2™** cannot detect until you run your game.

All error messages will have the following format:

ERROR CODE, Player or Opponent code, instruction #.

ERROR CODE

ERROR 1: A PERFORM instruction encountered with five previous PERFORMS unresolved.

ERROR 2: A RETURN instruction encountered without a matching PERFORM.

ERROR 3: Control transferred to non-existing instruction number.

PLAYER OR OPPONENT: If the error occurred in the PLAYER program, a ***'P'*** will be displayed. An ***'O'*** will be used to designate OPPONENT.

INSTRUCTION #: The number of the instruction where the error occurred.

SECTION 7 KEYBOARD TOUR

RETURN key

- You press RETURN at the end of each line of instruction. Pressing this key tells the computer to enter the line, or to execute the instruction(s). Sometimes it helps to think of RETURN as an ENTER key because this key actually enters the information or instruction into the computer.

CLR/HOME key

- Press this key and the cursor moves to the top left-hand corner of the screen (the "home" position). If you hold down the SHIFT key and press this key, the cursor still moves to the home position, but you also clear the screen. This key will return you to **COCO 2™**'s original color when you are in a menu screen.

INST/DEL key

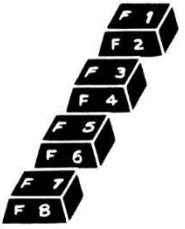
- You can insert and delete characters from the line you are typing by pressing this key. When you press the key by itself (delete), the character that was immediately to the left of the cursor disappears. If you're in the middle of a line, the character to the left is deleted and the characters to the right automatically move in to close up the space. Holding down SHIFT and pressing this key, opens up a space in the line so you can insert a new character. This is very powerful for editing and correcting mistakes!

CRSR key

- With the CRSR key, you can easily move program instructions up and down on the program screen. The CRSR key has a repeat feature that keeps the cursor moving until you release the key. The key has arrows that tell you the directions the key controls - up and down. To scroll down, you simply press the key. To scroll up, you must hold down the SHIFT key while pressing the CRSR key.

  The left/right CRSR key allows you to move the cursor sideways in ENTER or CHANGE mode of the PROGRAM THE GAME function.

COLOR CONTROL keys - The top 2 tan keys on the right side of the keyboard control the color of the display. F1 changes the color of the background. F3 changes the color of the border. F5 and F7 are not used. [F5 will change foreground scenes. F7 is not used.]



SECTION 8
GLOSSARY AND INSTRUCTION SUMMARY

SYNTAX — A word which means the exact spelling and structure of an instruction.

PROGRAM — A group of instructions that together form a job that a computer can do.

PIXEL	A small portion of a character on the computer screen. This is the smallest addressable object on a screen. A character is made up of pixels arranged in a specific pattern.
MENU	A list of functions that can be performed by the computer. Each function is numbered.
MICROPROCESSOR	Computer on a chip. Your computer has a microprocessor in it.
OBJECT	A drawing made up of pixels which is used to play a game.
CURSOR	The little black square. When you type something into the computer it will appear at the CURSOR position.
HIT	A collision between a shot, a bomb or a missile and an object. If the player hits the opponent with a shot or bomb the player's score is increased. If a player object is hit, it is destroyed.
DEBUGGING	A function performed by the programmer to correct his program. Usually performed by running the game and finding the errors and then making the necessary changes when the result is wrong.
DIRECTORY	The list of the games you have saved on diskette.
LOAD	The act of copying a game from diskette or tape into memory.
SAVE	The act of saving a game from memory to diskette or tape.
FLOW	The order of execution of a program.
VARIABLE	A variable can be thought of as a box in which a value may be stored.
NAME	A group of characters used to identify a game.

INSTRUCTION SUMMARY

INSTRUCTION	PARAMETER 1	PARAMETER 2
MISSILE		
BOMB	1-9,R,V,J	
SHOOT	1-9,R,V,J	
MOVE	1-9,R,V,J	1-99,R,V,J
PERFORM	1-9,R,V,J	1-99,R,V,J
RETURN		
LET V =		1-99,R,V,J
GOTO		1-99,R,V,J
IF SCORE =	B,C,D,E	1-99,R,V,J
IF J =	1-9,R,V,J	1-99,R,V,J
IF R =	1-9,R,V,J	1-99,R,V,J
IF R >	1-9,R,V,J	1-99,R,V,J

J = represents the current direction of the Joystick.

R = random number generator

V = general purpose variable

SCORE = the player's score

B,C,D,E = difficulty levels based on the score

INDEX

IMPORTANT WARRANTY INFORMATION

LIMITED 90 DAY WARRANTY ON ACA PERSONAL COMPUTER PRODUCTS

ADVANTAGE COMPUTER ACCESSORIES ("ACA") warrants to the original consumer purchaser that this ACA Personal Computer Product (not including computer programs) shall be free from any defects in material or workmanship for a period of 90 days from the date of purchase. If any such defect is discovered within the warranty period, ACA's sole obligation will be to repair or replace, at its election, the Computer Product free of charge on receipt of the unit (charges prepaid, if mailed or shipped) with proof of date of purchase satisfactory to ACA.

YOU MUST RETURN DEFECTIVE COMPUTER PRODUCTS TO ACA FOR IN-WARRANTY REPAIR.

This warranty shall not apply if the Computer Product: (i) has been misused or shows signs of excessive wear, (ii) has been damaged by being used with any products not supplied by ACA, or (iii) has been damaged by being serviced or modified by anyone other than ACA.

ANY APPLICABLE IMPLIED WARRANTIES, INCLUDING WARRANTIES OR MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE HEREBY LIMITED TO NINETY DAYS FROM THE DATE OF PURCHASE. CONSEQUENTIAL OR INCIDENTAL DAMAGES RESULTING FROM A BREACH OF ANY APPLICABLE EXPRESS OR IMPLIED WARRANTIES ARE HEREBY EXCLUDED. Some jurisdictions do not allow limitations on how long an implied warranty lasts or do not allow the exclusion or limitation of incidental or consequential damages, so the above limitations or exclusions may not apply to you.

This warranty gives you specific legal rights and you may also have other rights which vary from province to province.

DISCLAIMER OF WARRANTY ON ACA PROGRAMS: All ACA programs are distributed on an "as is" basis without warranty of any kind. The entire risk as to the quality and performance of such programs is with the purchaser. Should the programs prove defective following their purchase, the purchaser and not the manufacturer, distributor, or retailer assumes the entire cost of all necessary servicing or repair.

ACA shall have no liability or responsibility to a purchaser, customer, or any other person or entity with respect to any liability, loss, or damage caused directly or indirectly by computer programs sold by ACA. This disclaimer includes but is not limited to any interruption of service, loss of business or anticipatory profits or consequential damages resulting from the use or operation of such computer programs.

ACA warrants to the original consumer purchaser that the ACA Home Computer Cassette, Cartridge, and/or Diskette ("Computer Media"), not including computer programs, shall be free, from any defects in material or workmanship for a period of 90 days from date of purchase.

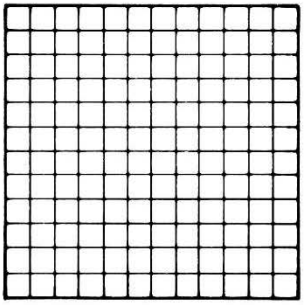
Any ACA Home Computer Media which is found to be defective during the warranty period will be replaced by ACA. Computer Media returned for in-warranty replacement must have the ACA label still intact, must be accompanied by proof of date of purchase satisfactory to ACA, and must be delivered or shipped no later than one (1) week after the end of the warranty period, shipping charges prepaid.

IMPORTANT: If you ship your ACA computer product, package it securely and ship it, charges prepaid and insured, by Parcel Post or United Parcel Service to:

ADVANTAGE COMPUTER ACCESSORIES
1020 MEYERSIDE DRIVE, UNIT 8
MISSISSAUGA, ONTARIO
L5T 1J4

TEACHERS and STUDENTS of **COCO 2™** use copies of this page as a worksheet.

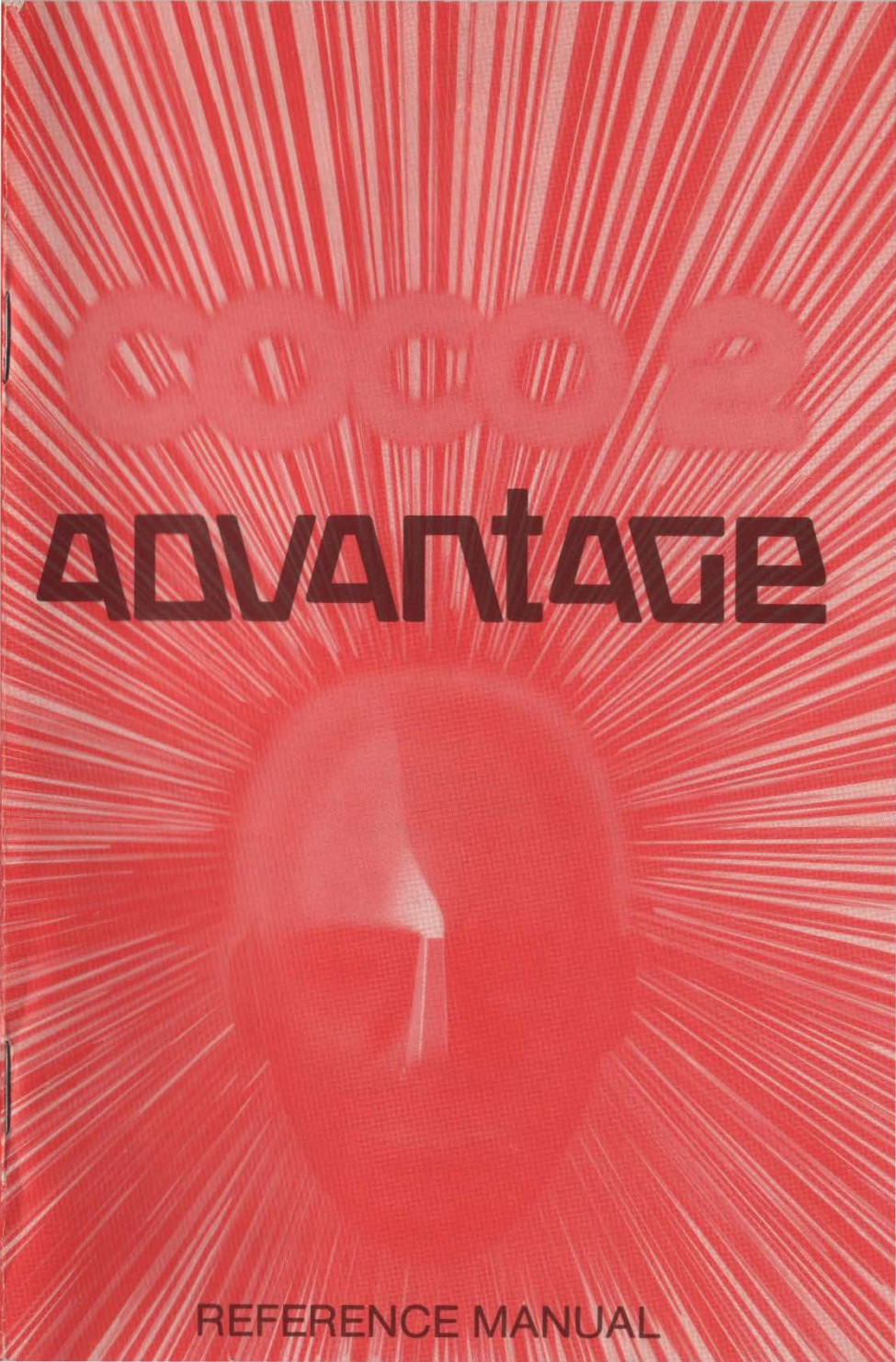
DRAW OBJECT



PROGRAM INSTRUCTIONS

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	

FOR VIC-20 ONLY


ADVANTAGE
REFERENCE MANUAL