

## Chapter 7. Turtle Positions and Coordinates

So far, we've moved the turtle around the screen using the direction commands: **FORWARD**, **BACK**, **LEFT**, **RIGHT**. Well...there are two other ways to move the turtle around that give you all sorts of new things to do. These new choices are turtle headings and x-y coordinates.



### **Turtle Headings**

When you want the turtle to move from Home up to the right, you can tell her go **RIGHT 45 FD 100 LEFT 45**.

Another way to do the same thing is to tell the turtle to set her heading to 45 and then go forward 100 steps and then set her heading back toward the top of the screen.



**SETHEADING 45 FD 100 SETHEADING 0**

## Turtle Positions and Coordinates

Or you can use shorthand. Clear the screen and type...

```
SETH 45 FD 100 SETH 0
```

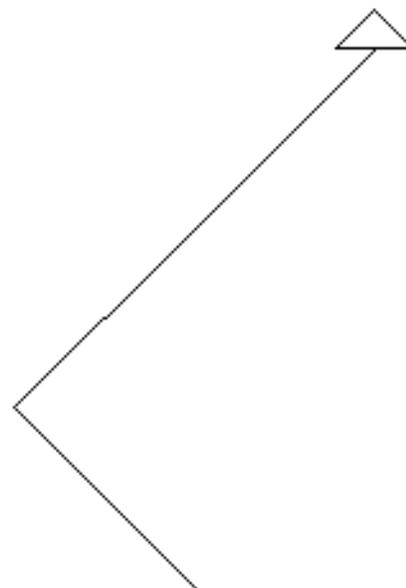
When you typed **SETH 45**, what direction did the turtle turn? She turned the same direction as when you told her **RIGHT 45**, correct?

If you typed **SETH - 45 FD 100 SETH 0**, what do you think would happen? Go ahead...try it and see. Remember to clear the screen first.

OK...now type **SETH 45 FD 100 SETH 0** again, but without clearing the screen. What happened?

Hmmm? This is interesting. Now type **SETH 45 FD 100 SETH 0** again. What happened this time?

Well, it seems like you can use the **SETH** command to turn the turtle. But it sure doesn't act like the **RIGHT** and **LEFT** commands.



If you ever want to know what heading the turtle is on, type...

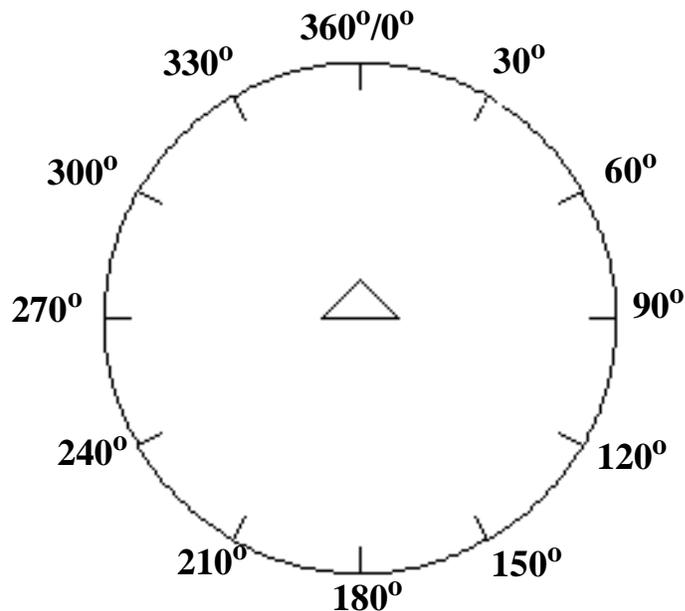
```
SHOW HEADING
```

**HEADING** is a new command. You can easily guess what that means. **SHOW** is another new command that is

a lot like **PRINT**. It shows something in the command window. You'll see more of **SHOW** later. In the meantime, let's try to make some sense out of this headings business.

Remember the string clock you made a while ago...the one you made with a piece of board and some nails? This is a good tool for learning about headings.

Well, let's look at that clock again.



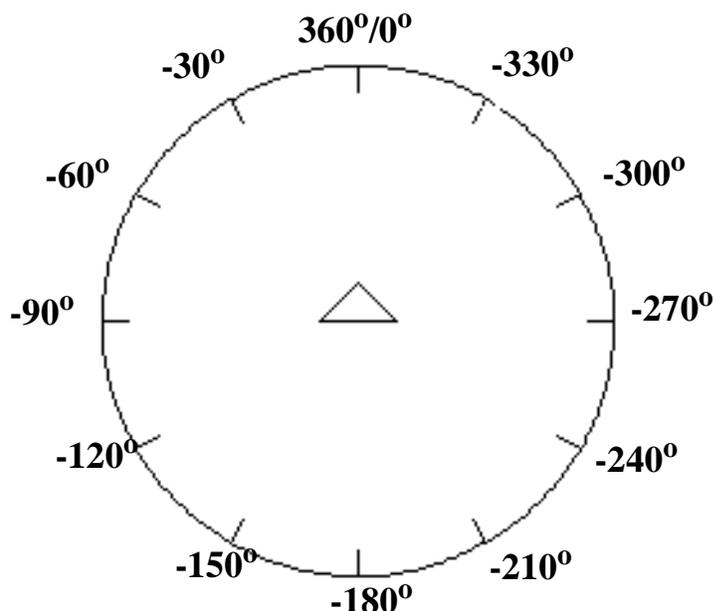
What heading is at 12:00 o'clock? 3:00 o'clock? 6:00 o'clock? 9:00 o'clock?

Get the idea? Turtle headings follow the hands of the clock. The heading at 12:00 o'clock is zero...at 3:00 o'clock, it's 90...at 6:00 o'clock, it's 180...and at 9:00 o'clock, it's 270.

This isn't to say that you can't move counter clockwise...to the left. You've already seen what happens when you typed...

**SETH - 45.**

## Turtle Positions and Coordinates



What time would it be at SETH -90? SETH -180?  
SETH -270?

---

### The Turtle's Compass

When you look at a clock like this, with all the hours marked off in degrees, what does it make you think of?

How about a compass?

Yep, that's it. It's like a compass.

Now...what do you think you can learn from a compass?

Morf's got a great game that will help you out. You'll learn about the procedures to draw this marked-off circle a little later in this chapter.

---

## **Rabbit Trail 18. The Turtle Rally**



Have you ever heard of a car rally?

Well...car rallies are special types of races. Only this type of race doesn't determine how fast you can drive. It determines how well you can follow directions.

Each car has a driver and a navigator. The team that can find their way through the course as close to the time limit is the winner.

In the Turtle Rally, you are given a compass and a list of directions. You'll need a digital watch that shows the seconds or a watch with a second hand so you can time yourself.

Here's how to play.

Have one person set up the course...maybe your teacher, your mom, or a person elected to be Rally Captain. This can be a course through the house, through the neighborhood, around the school yard, or anywhere you can set up a safe course to follow.

The directions will read something like this, only there should be many more.

1. You have 15 minutes to complete the race.
2. Start from Home. Set your heading to 0 and go forward for 40 seconds.
3. Set your heading to 90 and walk forward 120 steps.
4. Set your heading to 30 and run forward for 10 seconds.

You can make up any kind of directions you want. They can be simple. They can be silly. Or they can be tough. It makes the race more fun when you mix them up...walking, running, turning...and don't forget Back!

The Rally Captain times each team...the time each team starts and the time each team finishes. Each team starts a minute or two apart so that teams don't bump into each other along the path.

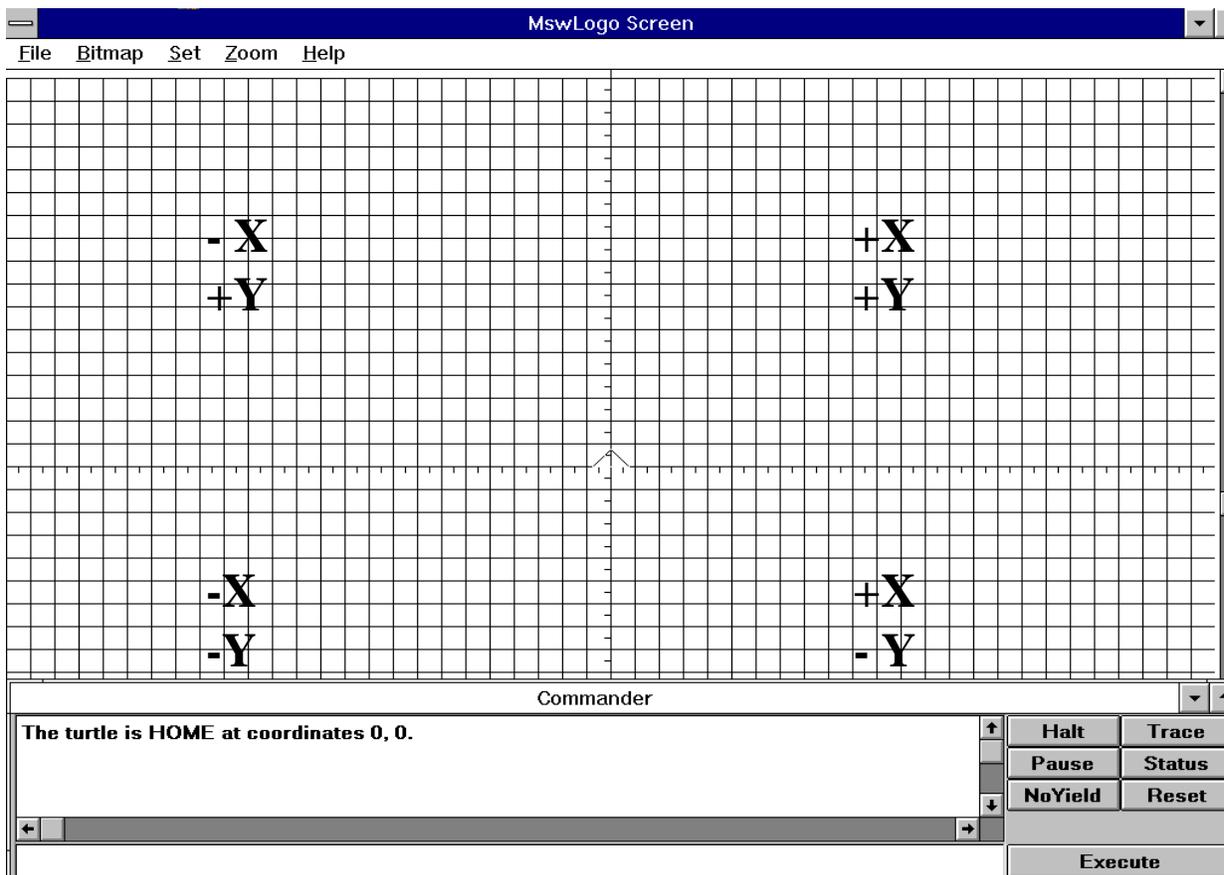
## Turtle Positions and Coordinates

It is a good thing to have checkpoints along the path where teams have to pick up something like a piece of paper and return it to the finish line. Maybe the paper can have a question each player has to answer correctly. Use your imaginations. The main thing is to have some fun while practicing with headings and directions.

After you've run your Turtle Rally, why not try one on the computer? You'd be surprised what you can create using headings and directions. In fact, that's just what you'll do a little later in this chapter.

## X - Y Coordinates

Imagine this screen with 1000 columns drawn on it and 680 rows. This is what shows up on a 14" SVGA monitor. Other monitors may display something different.



**Columns go up and down. Rows go from side to side. To find a position, you'd have to know which column and which row to pick.**

**This is exactly what you do when using the X- and Y-coordinates. X-coordinates count the number of columns from side-to-side. Y-coordinates count the number of rows up and down.**

**There are two lines that divide the screen in half, one divides the screen from top to bottom and one that divides the screen down the middle, from side to side. Where these lines meet is HOME...where X is 0 and Y is 0.**

**If you move up to the right, both the X-coordinate numbers and Y-coordinate numbers are positive...(+).**

**If you move up to the left, the X-coordinate is negative (-) but the Y-coordinate is positive.**

**If you move down to the left, both the X- and Y-coordinates are negative.**

**If you move down to the right, the X-coordinate is positive and the Y-coordinate is negative.**

**If you'd like to put the same type of grid on your computer screen, use this procedure...**

**TO GRID**

**HT CS CT PRINT [Let's take a look at x-y coordinates.]**

**WAIT 100 CS G GRAPH CT**

**PRINT [The turtle is HOME at coordinates 0, 0.]**

**WAIT 300 JB**

**END**

**TO G**

**REPEAT 50 [D] RT 90 REPEAT 50 [D]**

**END**

## Turtle Positions and Coordinates

```
TO D
RT 90 FD 10 LT 90 BK 5 FD 5 RT 90 FD 10 LT 90 FD 2
BK 7 FD 5
END
```

```
TO GRAPH
XLINE -500 LASTX
YLINE 340 LASTY
PU HOME PD ST
END
```

```
TO JB
PR [The X axis runs side to side,]
PR [each line is 20 columns apart.] WAIT 100
PR [The Y axis runs up and down,]
PR [each vertical line is 20 rows apart.] WAIT 100
PR [The X axis runs from -500 to the left,]
PR [to 500 on the right.] WAIT 100
PR [The Y axis runs from -340 below,]
PR [to 340 at the top of the screen.] WAIT 100
PR [Now...let's see how to set x-y coordinates.]
DUOTRI
END
```

```
TO LASTX
PU SETXY 500 340 PD SETXY 500 -340
END
```

```
TO LASTY
PU SETXY -500 -340 PD SETXY 500 -340
END
```

```
TO DUOTRI
CS CT ST
PR [You can even draw using]
PR [the coordinate commands.] WAIT 100
PR [SETX -100] SETX -100 WAIT 100
PR [SETY 100] SETY 100 WAIT 100
PR [SETXY 0 0] SETXY 0 0 WAIT 100
PR [SETXY 100 100] SETXY 100 100 WAIT 100
PR [SETY 0] SETY 0 WAIT 100
PR [SETX 0] SETX 0 WAIT 100
END
```

```
TO XLINE :SIZE
PU SETXY :SIZE 340 PD SETXY :SIZE -340
IF :SIZE = 500 [STOP]
XLINE :SIZE + 20 HT
END
```

```
TO YLINE :SIZE
PU SETXY -500 :SIZE PD SETXY 500 :SIZE
IF :SIZE = -340 [STOP]
YLINE :SIZE - 20 HT
END
```

---

## **XCOR and YCOR**

**HOME** is at x-coordinate 0 and y-coordinate 0. In Logo, we use the shorthand names...

**XCOR**

**YCOR**

Clear the screen. Then type **SHOW XCOR**. What happened?

## Turtle Positions and Coordinates

**Hmmmmm. Very interesting.**

**What do you think you need to do to send the turtle to another x-coordinate? Give yourself a Gold Star if you said SETX...not SET XCOR. Go ahead...try it. Type SETX and a number.**

**SETX \_\_\_\_\_**

**SETX \_\_\_\_\_**

**Now try moving the turtle up and down. What's the command? That one should be real easy.**

**SETY \_\_\_\_\_**

**SETY \_\_\_\_\_**

**I bet you've already guessed that you can combine these two commands. Sure...**

**SETXY \_\_\_\_\_ \_\_\_\_\_**

**That's right. You have to type both the X position and the Y position with the SETXY command. Remember that when using the SETXY command, you use numbers for the values of X and Y.**

**SETXY \_\_\_\_\_ \_\_\_\_\_**

**SETXY \_\_\_\_\_ \_\_\_\_\_**

**SETXY \_\_\_\_\_ \_\_\_\_\_**

**SETPOS uses a list of two numbers. Try a few.**

**SETPOS is another command that can move the turtle around the screen. SETXY takes two numbers. SETPOS takes a list of two numbers. There's an important difference.**

**Remember about lists? They must be put inside brackets. So...in the case of SETPOS, you have...**

**SETPOS [ *x-coordinate y-coordinate* ]**

**What are the x and y positions in this statement?**

**SETPOS [20 45]**

**The x-coordinate is 20 and the y-coordinate is 45.**

**Now...where were we? Was the pen down when you typed the SETX, SETY, SETXY, and SETPOS commands? What happened? Did you end up with a crazy drawing on the screen?**

**If the pen was up, put it down and then do a few more SETXY commands...to get some practice using the commands.**

**There...see what you can do with these coordinate and position commands? They were used to help make the marked-off circles you saw at the beginning of this chapter.**

**Here's the procedures.**

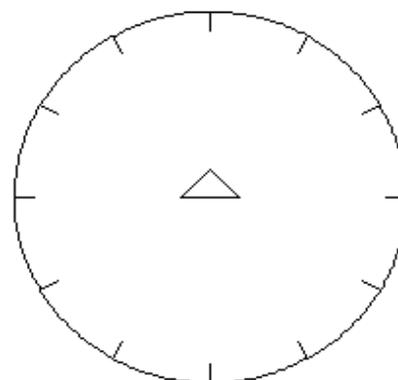
```
TO ARCL :CENTER :RADIUS :ANGLE  
LOCAL "AMT  
MAKE "AMT :RADIUS * PI / 180  
PU SETPOS :CENTER LT 90 FD :RADIUS LT 90 PD  
REPEAT :ANGLE [FD :AMT LT 1]  
END
```

```
TO ARCR :CENTER :RADIUS :ANGLE  
LOCAL "AMT  
MAKE "AMT :RADIUS * PI / 180  
PU SETPOS :CENTER LT 90 FD :RADIUS RT 90 PD  
REPEAT :ANGLE [FD :AMT RT 1]  
END
```

```
TO CIRCLEM :CENTER :RADIUS :ANGLE  
REPEAT 360 / :ANGLE [ARCR :CENTER :RADIUS  
:ANGLE MARK]  
PU SETPOS :CENTER PD  
END
```

```
TO MARK  
RT 90 FD :RADIUS/10  
BK :RADIUS/10 LT 90  
END
```

```
TO PI  
OUTPUT 3.14159  
END
```



**CIRCLEM** draws a circle as a series of arcs with a mark between the arcs.

To draw the individual arcs that are between the hours on the clock, type...

```
ARCR [100 20] 100 30
```

In these procedures, the variable **:CENTER** is a list representing the **x-** and **y-**coordinates. The turtle moves to the **x-**coordinate of 100 and the **y-**coordinate of 20. It then turns and moves to the edge of the arc and draws it one degree at a time.

This is quite different from the circles you did before. With those, you simply started at the edge and drew a circle. You had to tell the turtle how far to go forward and how many degrees to turn.

**REPEAT 360 [ FD \_\_\_\_\_ RT \_\_\_\_\_ ]**

In these procedures, you tell the turtle where the center of the circle is to be and what the radius is to be. You know what a radius is, don't you? It's the distance from the center of a circle to the edge. It's one-half the diameter of the circle. The diameter stretches from edge to edge through the middle of the circle.



You know how Morf likes to experiment. So here's one for you.

Change either of the ARC procedures so that they use the SETXY command rather than the SETPOS command.

What's that :AMT variable?

That's part of the mathematical formula for drawing a circle. It calculates the size of the circle based on the size of the radius you give it.

```
TO CIRCLE :CENTER :RADIUS  
LOCAL "AMT  
MAKE "AMT :RADIUS * PI/180  
PU SETPOS :CENTER  
SETX XCOR - :RADIUS SETH 0 PD  
REPEAT 360 [FD :AMT RT 1]  
PU SETPOS :CENTER PD  
END
```

## Turtle Positions and Coordinates

**Change the  $PI / 180$  to  $PI / 360$ . What happens?**

**Do you remember the relationship between the diameter of the circle and  $PI$ ?**

**Diameter \*  $PI$  = Circumference**

**So...maybe  $:RADIUS * PI / 180$  will make more sense if we write it as...**

**$(:RADIUS * 2) * PI / 360$**

**Radius \* 2 equals the diameter, right? Then we have the diameter \*  $PI$  that equals the distance around the circle. That's divided by 360. How does this fit in the line...**

**REPEAT 360 [FD :AMT RT 1]**

**That makes :AMT equal to 1. You end up with the more familiar command for drawing a circle...**

**REPEAT 360 [FD 1 RT 1]**

**Awfully simple? Or simply awful?**

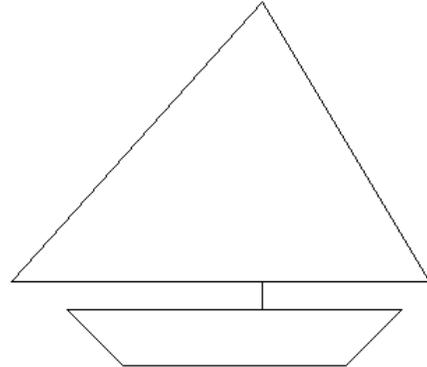
---

## Creative Coordinates

**You saw how coordinates can be used to create pictures in the GRID procedure. Here's a simple procedure that uses SETXY COMMANDS.**

```
TO SAILBOAT  
CS PU SETXY 20 -40 PD  
SETXY -160 -40
```

```
SETXY 20 160  
SETXY 140 -40  
SETXY 20 -40  
PU SETXY -120 -60 PD  
SETXY -80 -100  
SETXY 80 -100  
SETXY 120 -60  
SETXY 20 -60  
SETXY 20 -40  
SETXY 20 -60  
SETXY -120 -60  
END
```



Now...how about a challenge. Play around with this procedure to see what it draws. WHAT1, WHAT2, and WHAT3 are simply subprocedures to break a large procedure into pieces.

```
TO WHAT.IS.IT  
CS CT PU SETXY 5 25 PD  
WHAT1 WHAT2 WHAT3  
END
```

```
TO WHAT1  
SETXY 25 25 SETXY 30 25  
SETXY 35 25 SETXY 35 15  
SETXY 30 0 SETXY 35 -10  
SETXY 30 -15 SETXY 25 -5  
SETXY 25 -15 SETXY 15 -15  
SETXY 5 -5 SETXY 10 -15  
SETXY 0 -15  
END
```

## Turtle Positions and Coordinates

**TO WHAT2**

```
SETXY -5 -5 SETXY -5 -15  
SETXY -15 -15 SETXY -10 -5  
SETXY -15 5 SETXY -20 0  
SETXY -25 -10 SETXY -30 -15  
SETXY -35 -15 SETXY -40 -10  
SETXY -40 -5 SETXY -35 -10  
SETXY -35 -10 SETXY -30 -10  
SETXY -25 0 SETXY -25 10  
END
```

**TO WHAT3**

```
SETXY -30 15 SETXY -30 25  
SETXY -20 35 SETXY -15 35  
SETXY -10 30 SETXY -5 25  
SETXY -5 30 SETXY 0 30  
SETXY 5 25 SETXY 5 10  
SETXY -5 5 SETXY -10 5  
SETXY -10 10 SETXY -15 5  
PU HOME PD  
END
```

**Type this procedure on your computer to see what it draws. Or load it and run it. It came with MSW Logo. Here's a hint...it's a circus animal. Want to make a guess? You can see if you were right when you run the WHAT.IS.IT procedure.**

**Now let's dress this procedure up a bit.**

**TO START**

```
WHATIS 400  
END
```

**TO WHAT :X**

**SETXY :X + 25 25 SETXY :X + 30 20  
SETXY :X + 35 25 SETXY :X + 35 15  
SETXY :X + 30 0 SETXY :X + 35 -10  
SETXY :X + 30 -15 SETXY :X + 25 -5  
SETXY :X + 25 -15 SETXY :X + 15 -15  
SETXY :X + 5 -5 SETXY :X + 10 -15  
SETXY :X + 0 -15  
END**

**TO WHAT2 :X**

**SETXY :X - 5 -5 SETXY :X - 5 -15  
SETXY :X - 15 -15 SETXY :X - 10 -5  
SETXY :X - 15 5 SETXY :X - 20 0  
SETXY :X - 25 -10 SETXY :X - 30 -15  
SETXY :X - 35 -15 SETXY :X - 40 -10  
SETXY :X - 40 -5 SETXY :X - 35 -10  
SETXY :X - 35 -10 SETXY :X - 30 -10  
END**

**TO WHAT3 :X**

**SETXY :X - 25 0 SETXY :X - 25 10  
SETXY :X - 30 15 SETXY :X - 30 25  
SETXY :X - 20 35 SETXY :X - 15 35  
SETXY :X - 10 30 SETXY :X - 5 25  
SETXY :X - 5 30 SETXY :X + 0 30  
SETXY :X + 5 25 SETXY :X + 5 10  
SETXY :X - 5 5 SETXY :X - 10 5  
SETXY :X - 10 10 SETXY :X - 15 5  
PU HOME PD  
END**

## Turtle Positions and Coordinates

```
TO WHATIS :X
IF :X < - 400 [STOP]
PU SETXY :X + 5 25 PD
WHAT :X
MAKE "X :X - 70
WHATIS :X
END
```

What do you think this procedure will draw? You deserve First Prize if you said a row of circus animals. To draw our row of animals, we use variables with the SETXY command.

But, wait a minute...the x-coordinate is the only variable...why?

The reason for this is that the x-coordinate is the only coordinate that changes as each animal is drawn. The y-coordinates stay the same for each animal drawn.

Remember...y-coordinates are the lines going up and down. If you changed them, some of the animals might end up walking in mid-air. And that would sure look funny!

---

## A Target Game

Want to have some fun practicing with x- and y-coordinates? Then try this target game.

The turtle draws a small target on the screen. You are then asked to guess the x- and y-coordinates of the target. If you guess correctly, the CHEERS procedure runs. If you don't guess right, you get another chance.

Go ahead...it's fun.

```
TO GAME
CS CT TARGET ASK
END
```

**TO TARGET**

**PU HT SETXY ( RANDOM 900 ) - 450  
( RANDOM 680 ) - 340**

**MAKE “X XCOR + 12 MAKE “Y YCOR**

**PD REPEAT 36 [FD 2 RT 10]**

**PU HOME ST**

**END**

**TO ASK**

**PRINT [Can you hit the target?] WAIT 100**

**PRINT [Give me an X- and a Y-coordinate.]**

**WAIT 100**

**PRINT [X = ] MAKE “X1 READWORD**

**PRINT [Y = ] MAKE “Y1 READWORD**

**TEST OR (:X1 > ABS 480)(:Y1 > ABS 340)**

**IFTRUE [PRINT [Your number is too big. Try again.]**

**WAIT 100 ASK]**

**IFFALSE [PD ST SETXY :X1 :Y1]**

**IF OR ( :X1 > ( :X + 12 ) ) ( :X1 < ( :X - 12 ) )**

**[PU HOME PD ASK]**

**IF OR ( :Y1 > ( :Y + 12 ) ) ( :Y1 < ( :Y - 12 ) )**

**[PU HOME PD ASK]**

**CHEERS**

**END**

**TO CHEERS**

**CT REPEAT 24 [PRINT [Congratulations!]]**

**WAIT 100**

**PRINT [Want to play again? (Y/N) ]**

**MAKE “ANS RC**

**IF :ANS = “Y [GAME]**

**IF :ANS = “N [PRINT [Bye for now!]]**

**END**

## Turtle Positions and Coordinates

```
TO ABS :NUM  
OP IFELSE :NUM < 0 [-:NUM][:NUM]  
END
```

There's a couple of other new things you'll find in this game.

```
RANDOM :N
```

This command picks a number between 0 and the number you select... :N. Try it.

```
REPEAT 5 [PRINT RANDOM 10]
```

What happened? Bet you got numbers between 0 and 9. Let's look at the line in our game...

```
SETXY ( RANDOM 480 ) - 240 ( RANDOM 280 ) - 140
```

Seems complicated! Remember that **RANDOM** only picks numbers between 0 and the number you select. Since coordinates can be less than 0, we need random numbers that can put the target anywhere on the screen.

So let's read this line...

```
SETXY ( RANDOM 900 ) - 450 ( RANDOM 680 ) - 340
```

**SETXY** takes two inputs, the x and the y-coordinate. First, Logo looks for the x-coordinate. You know the MSW Logo screen is 1000 turtle steps wide. So, the x-coordinates can be from -500 on the left of the screen, to 500 on the right.

But since we don't want our target right on the edge, we make the biggest number **RANDOM** can select a little smaller than 500.

**( RANDOM 900 ) - 450**

If RANDOM selects the largest number...900...Logo subtracts 450 from it giving you an x-coordinate of 450.

If RANDOM selects the smallest number...0...Logo subtracts 450 giving you an x-coordinate of -450.

Once Logo has the x-coordinate, it reads to the right to find the y-coordinate.

**( RANDOM 680 ) - 340**

If we do the same arithmetic with the y-coordinate as we did with the x-coordinate, we can have a y-coordinate from 340 to -340. So...with this line, we can have a target just about anywhere on the screen.

**TEST OR (:X1 > ABS 450) (:Y1 > ABS 340)**

Here's another new command. Reading from the left, Logo first reads the word TEST. This is one of Logo's conditional commands. It tests a condition to see if it is true or false.

To find out what needs to be tested, Logo continues reading.

TEST OR...OR is one of those logical operators that asks if either of two conditions is true.

**(:X1 > ABS 450)**

**(:Y1 > ABS 340)**

**The first condition we find is...**

**(:X1 > ABS 450)**

**:X1 is greater than what?**

## Turtle Positions and Coordinates

**ABS stands for absolute. In MSW Logo, you need a procedure to define ABS.**

```
TO ABS :NUM  
OP IFELSE :NUM < 0 [-:NUM][:NUM]  
END
```

**What ABS says is that it doesn't care if the x-coordinate :X1 is a positive number (to the right of HOME) or a negative number (to the left of HOME). It drops the positive or negative symbol and just tests the random number.**

```
TEST OR (:X1 > ABS 450) (:Y1 > ABS 340)
```

**What this line says is to test if :X1 is greater than 450 (or less than -450) OR if :Y1 is greater than 340 (or less than -340).**

```
IFTRUE [PRINT [Your number is too big. Try again.]  
WAIT 20 ASK]
```

**If the test is true...the number you typed is too big... print "Your number is too big. Try again." If it is false, or within the proper range, then the turtle draws a line out to the coordinates you guessed.**

**Now, why not make up some of your own tests? Here's one we did...**

```
TO CHECK  
TEST HEADING = 0  
IFFTRUE [FORWARD 100]  
IFFALSE [HIDETURTLE]  
END
```

Another way to write this test condition is to use the IF command. In MSW Logo, use the IFELSE command...

```
IFELSE HEADING = 0 [FD 100][HIDETURTLE]
```

Some other versions of Logo allow you to write that line as follows...

```
IF HEADING = 0 THEN FD 100 ELSE  
HIDETURTLE
```

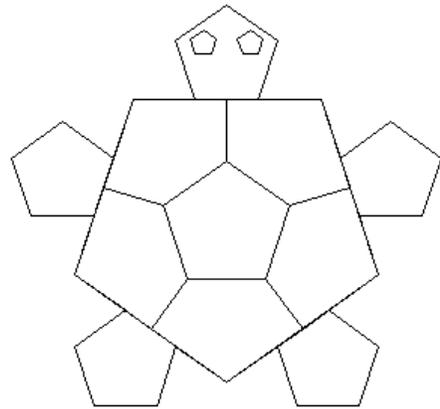
or...

```
IF HEADING = 0 [FD 100] [HIDETURTLE]
```

Here's one more use of turtle positions. This draws a turtle the size that you want it to be. Since TURTLE is already an MSW Logo primitive, the name was changed to YPLA.

```
TO YPLA :SIZE  
TURTLE1  
MAKE "P []  
REPEAT 6 [TURTLE2]  
TURTLE3  
END
```

```
TO TURTLE1  
CS PD HT RT 18 MAKE "TSIZE :SIZE / 3  
REPEAT 5 [FD :TSIZE LT 108 ~  
REPEAT 3 [FD :TSIZE RT 72] ~  
FD :TSIZE LT 108 FD :TSIZE RT 72]  
REPEAT 5 [FD :SIZE RT 72]  
END
```



## Turtle Positions and Coordinates

```
TO TURTLE2
FD :SIZE / 2 RT 90 FD :TSIZE
MAKE "Q POS
IF :P = [] [MAKE "P POS]
SETPOS :P SETPOS :Q
MAKE "P :Q BK :TSIZE LT 90
FD :SIZE / 2 RT 72
END
```

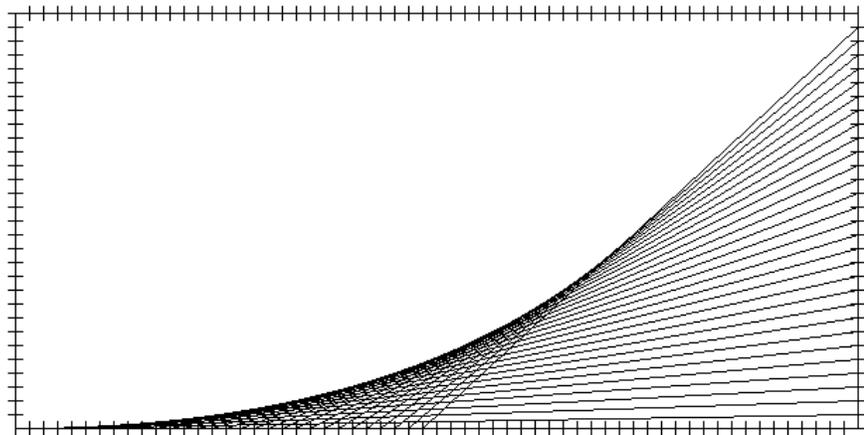
```
TO TURTLE3
PU FD :TSIZE LT 90 FD :SIZE / 4 LT 18 PD
REPEAT 5 [FD :SIZE / 12 RT 72]
PU RT 18 BK :SIZE / 4 RT 90
PU FD :TSIZE LT 90 FD :SIZE / 4 RT 18 PD
REPEAT 5 [FD :SIZE / 12 LT 72]
END
```

---

## More String and Wire Art

Remember when you did some string and wire art in Chapter 5? You never did get to the Logo procedures, did you?

Guess what? You're there now.



**BOXTOP** draws the Boxtop. You define the size of the short side and the number of marks to appear on that side. For example...

**BOXTOP 300 30**

The **CURVE** procedure looks complex. But it is simply the turtle doing the sewing that you did with a needle and colored yarn.

**CURVE 30 10 -300 0 300 0**

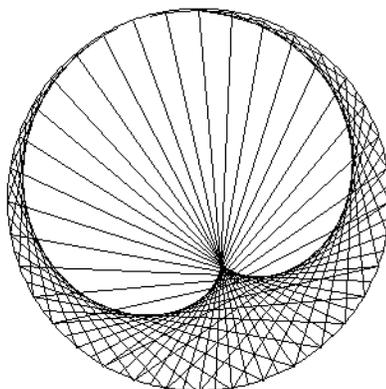
You are going to draw 30 lines that are 10 steps apart. You start at **:X1** in the lower left where the x-coordinate is **-300** and the **:Y1** is **0**. The turtle moves from **:X1, :Y1** to **:X2, :Y2**, and then back and forth 30 times.

Go ahead. Play with a few other combinations. Do the same things on the screen that you did with yarn. What else can you dream up?

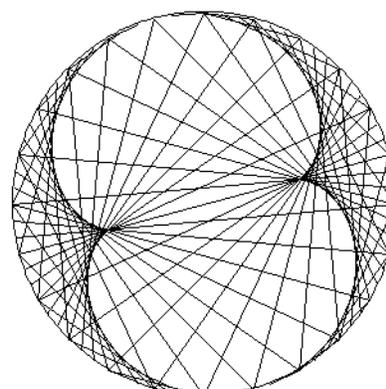
```
TO BOXTOP :DIST :MARK  
PU SETX :DIST - :DIST * 2 PD  
REPEAT 2 [MARKER :MARK RT 90 REPEAT 2  
  [MARKER :MARK] RT 90]  
END
```

```
TO CURVE :T :GAP :X1 :Y1 :X2 :Y2  
IF :T = 0 [STOP]  
PU SETXY :X1 :Y1 PD  
SETXY :X2 :Y2  
MAKE "X1 :X1 + :GAP  
MAKE "Y2 :Y2 + :GAP  
CURVE :T - 1 :GAP :X1 :Y1 :X2 :Y2  
END
```

## Turtle Positions and Coordinates



**STRING 150 2 2**



**STRING 150 3 2**

```
TO MARKER :MARK  
REPEAT :MARK [FD :DIST / :MARK MARKS]  
END
```

```
TO MARKS  
LT 90 FD 5 BK 10 FD 5 RT 90  
END
```

Once you've played with the BOXTOP, the following procedures make a bit more sense.

```
TO STRING :RADIUS :DIST :HEAD  
CS HT MAKE "N 1 PU SETX :RADIUS PD  
REPEAT 360 [FD :RADIUS * PI/180 LT 1] PU HOME  
REPEAT 36 * :HEAD [FD :RADIUS MAKE "P POS  
HOME HDG FD :RADIUS PD SETPOS :P PU  
HOME HDG1 MAKE "N :N +1]  
END
```

```
TO HDG  
SETH REMAINDER (:N * 5 * :DIST) 360  
END
```

```
TO HDG1  
SETH REMAINDER (5 * (:N-1)) 360  
END
```

```
TO PI  
OP 3.14159  
END
```

**Play with them for awhile, trying different variables. Not only do they create some beautiful patterns, they give you a look at how positions and headings can be used.**

**By the way, did you catch that REMAINDER command? That's part of the Great Math Adventure coming up in the next chapter.**

---

Turtle Positions and Coordinates

