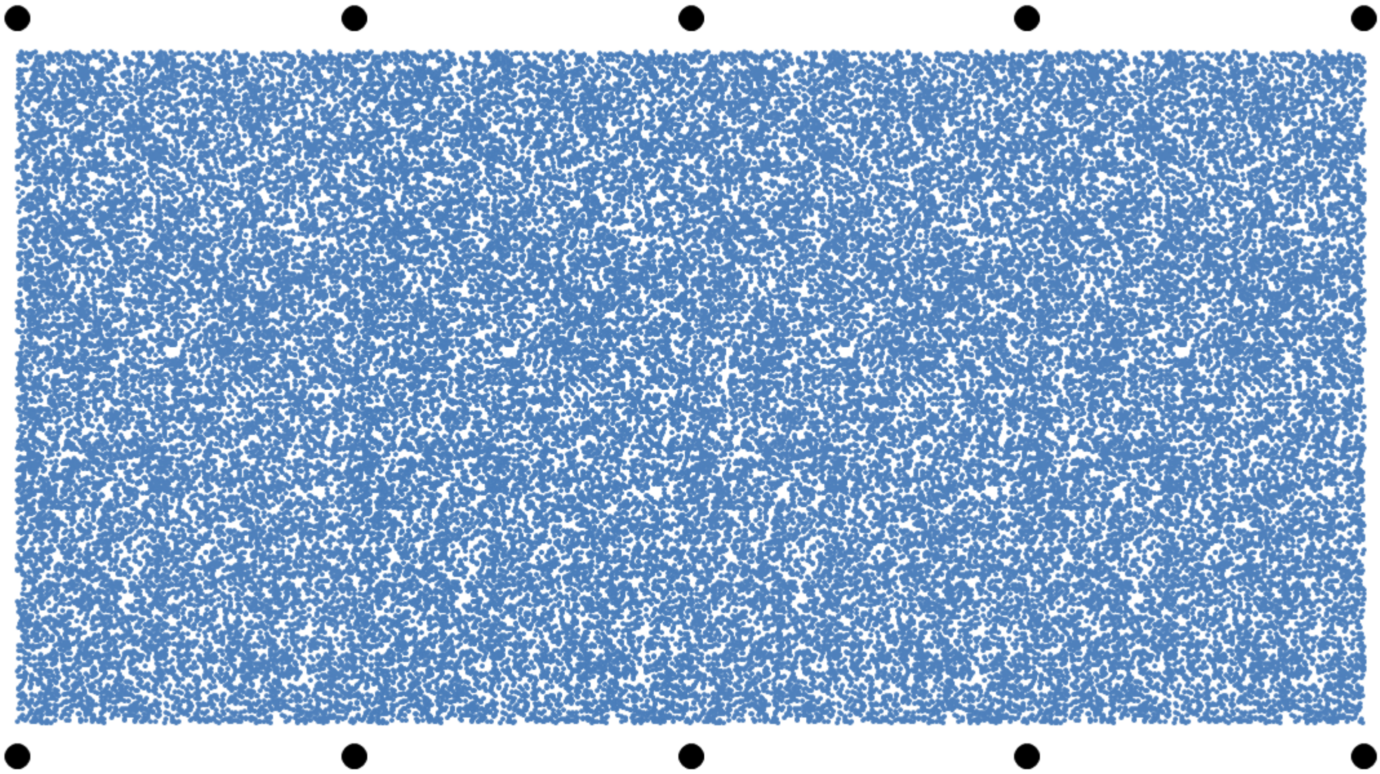
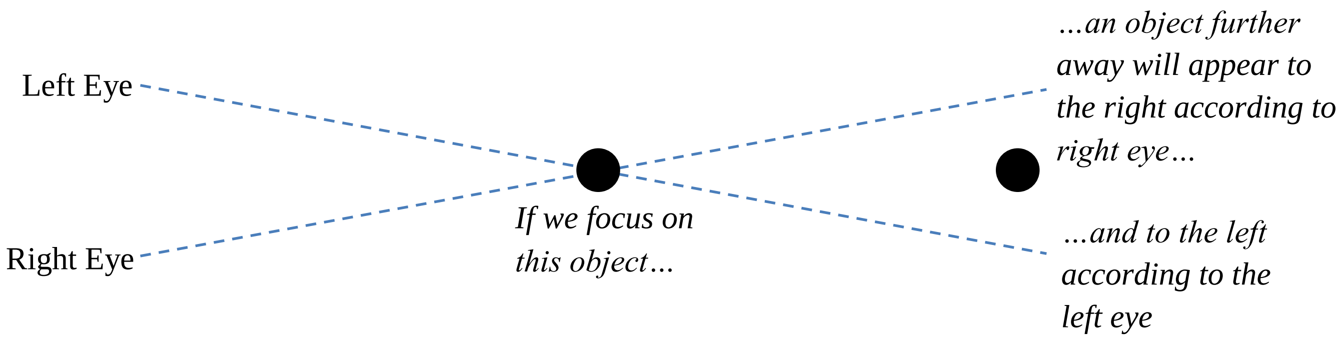
Creating Stereograms in Excel

Cross your eyes so that two adjacent black dots perfectly overlap. Relax until your eyes become focused. Look at the blue pattern and you should see something pop out of the page. What do you see?

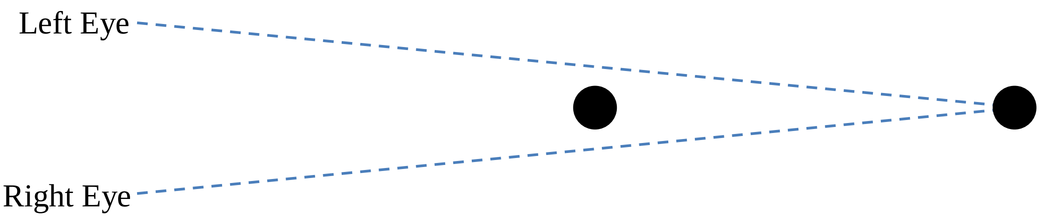


Stereograms work using the following principle:


Left Eye
Right Eye
If we focus on this object…
…an object further away will appear to the right according to right eye…
…and to the left according to the left eye

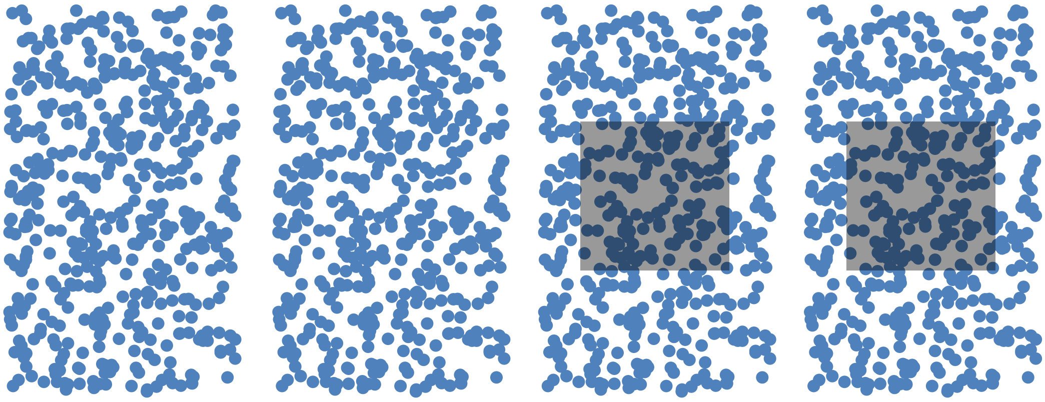
The object further away appears in a different position to each eye, so each eye sees a different version of it.

Alternatively, if we focus on the object which is further away the closer object will appear to the right according to the left eye and to the left according to the right eye.



Suppose we wish to create a stereogram which produces a square. We can do this using the following method:

Create four strips of randomly placed dots. The position of each dot in each strip is identical except for dots which lie inside the image we wish to produce in the two strips on the right. These dots are given a small horizontal offset.



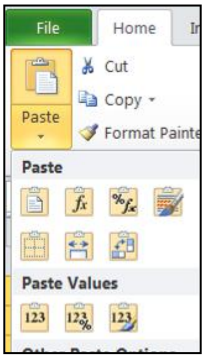
Note that, unlike the diagram above, there should be no space in between each strip of dots.

Use Excel to create a Stereogram of a shape or object of your choosing.

- Each strip should contain 15,000 dots and be 50 units wide and 100 units high.
- Each dot should be randomly placed to 1 decimal place.
- Dots in the two strips on the right which lie inside the shape you wish to create should be moved 0.5 units to the right.
- Include black dots above and below the stereogram to help the user focus. The spacing between these black dots should be equal to the width of one strip of dots (= 50 units).

Tip: Your stereogram should be created using a scatterplot with four sets of data. The black dots can be created using one more set of data.

Use conditional logic to determine whether the horizontal offset should be applied to each dot.



Create a report of your work using Word (or similar). Make sure you include:

- an introduction
- the theory behind stereograms
- an explanation of the formula used in Excel to create the Stereogram
- an explanation of why the viewer sees a 3-dimensional image when he/she views the stereogram correctly
- at least one large, clear stereogram

All work should be your own.

Teacher Notes

- The example produces a ring (a circle with a hole in the middle).

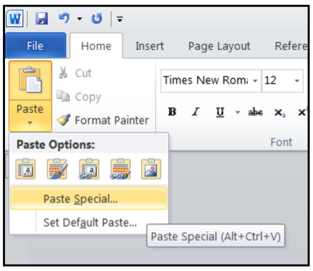
- There are two different ways to view a stereogram. The first is described in this document. The second is to imagine focusing on something further away until the black dots perfectly overlap and the eyes become focused.

  If we use this method then we should apply the horizontal offset to the left, and not the right.

- The RANDBETWEEN function only produces non-negative whole numbers. To create a random number to 1 decimal place we can generate a larger random number and then divide it by 10. For example.

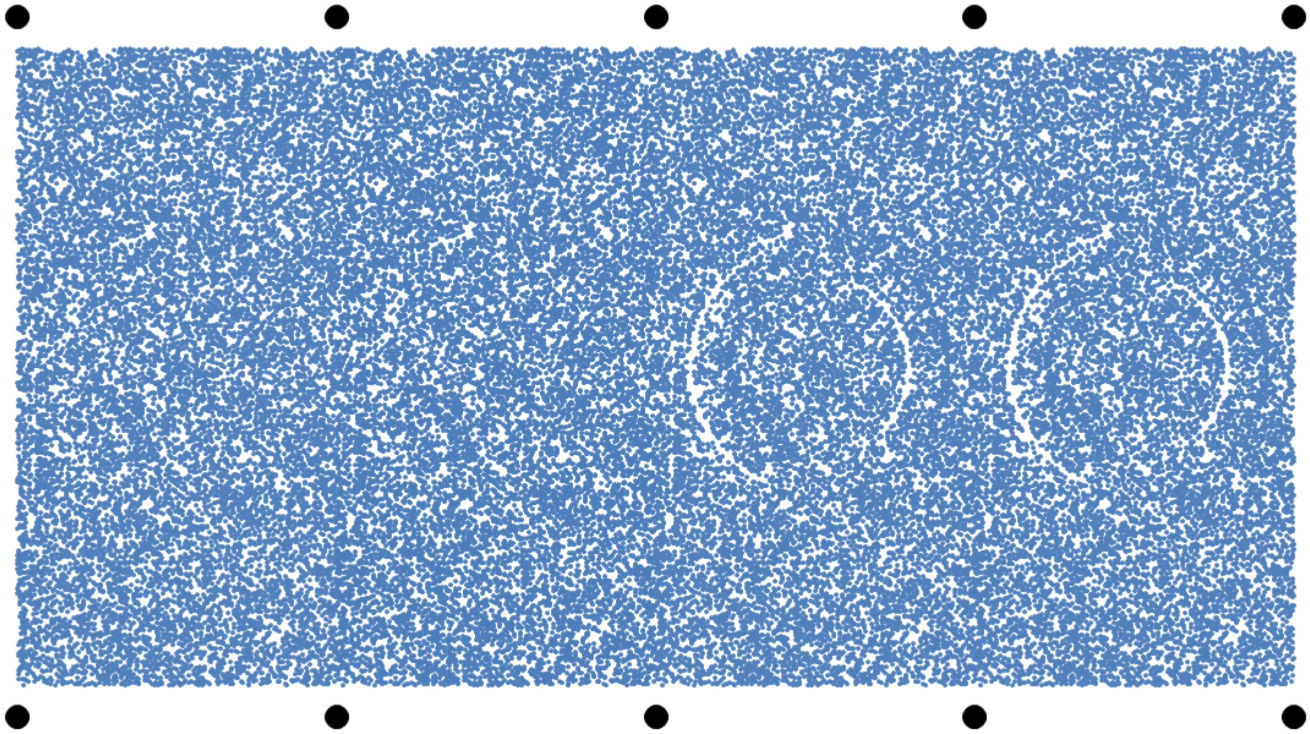
  =RANDBETWEEN(0,1000)/10

- When copying and pasting a complicated graph from Excel into Word it is often better to convert it into an image. To do this, in Word click *Home > Paste > Paste Special > Bitmap*



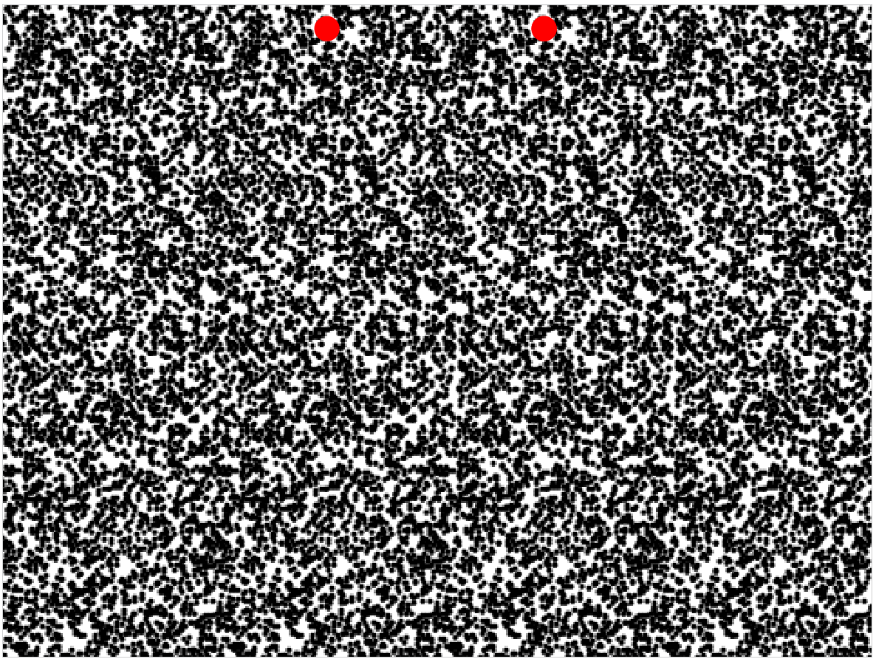
- Use conditional probability to determine if a point lies within the shape / object we wish to create, then position it accordingly (see the provided spreadsheet for an example). Depending on the shape the AND or OR functions, or nested IF functions may also need to be used.

- The student can experiment with different offset values but anything more than 0.5 or less than -0.5 results in stereogram hinting at what image it will produce. For example, using a horizontal offset of 1 gives the following. Part of a ring can easily be seen.

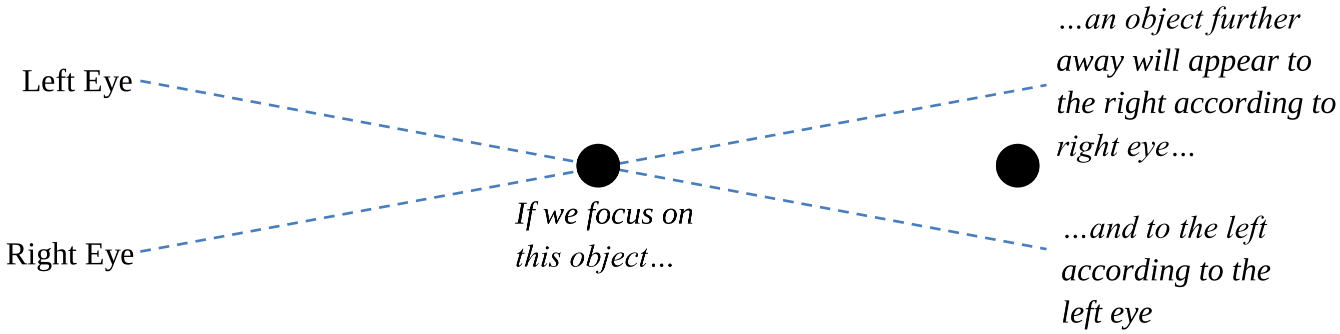


Creating Stereograms in Scratch

Cross your eyes so that two adjacent red dots perfectly overlap. Relax until your eyes become focused. Look at the black pattern and you should see something pop out of the page. What do you see?

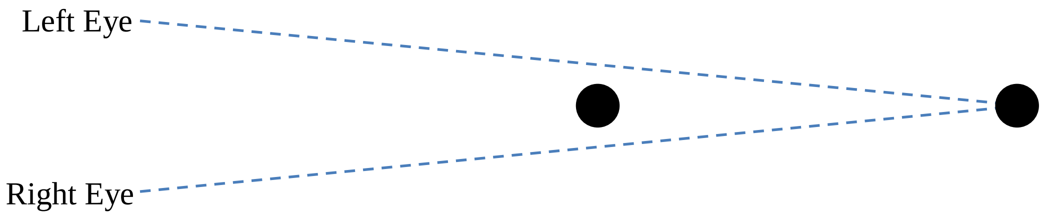


Stereograms work using the following principle:


Left Eye
Right Eye
If we focus on this object…
…an object further away will appear to the right according to right eye…
…and to the left according to the left eye

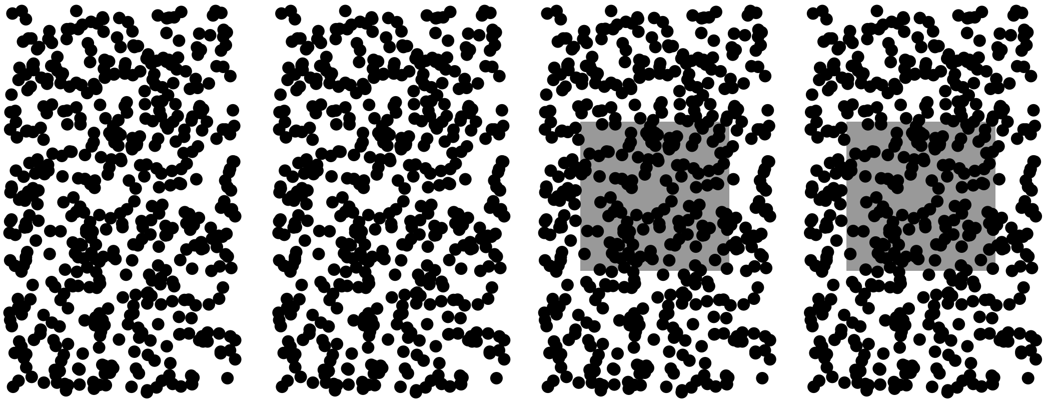
The object further away appears in a different position to each eye, so each eye sees a different version of it.

Alternatively, if we focus on the object which is further away the closer object will appear to the right according to the left eye and to the left according to the right eye.



Suppose we wish to create a stereogram which produces a square. We can do this using the following method:

Create four strips of randomly placed dots. The position of each dot in each strip is identical except for dots which lie inside the image we wish to produce in the two strips on the right. These dots are given a small horizontal offset.



Note that, unlike the diagram above, there should be no space in between each strip of dots.

Use Scratch to create a Stereogram of a shape or object of your choosing. Use the following steps to help you. Note that all of these steps should happen automatically when the user clicks the green flag.

- Each strip should be 120 high long and 360 pixels high.
- Create a sprite which represents the shape/object we wish to display. Place this sprite in the third strip.
- Create a sprite which represents one red dot. Use the *clone* function to place copies of it in the middle of the top of the second and third strips.
- Create one sprite which will represent all of the dots.
- Randomly place the dot in the first strip. Use the *stamp* function to leave a copy of the dot in the strip. Then, place a copy in the dot in the correct positions in all of the other three strips. If you need to give it a horizontal offset then give it a horizontal offset of 2 pixels.
- Repeat the previous bullet point 5000 times.
- Hide the sprite which represents the shape/object we wish to display.

Note that if the green flag is pressed after the program has already run then all previous dots will remain. Use the *clear* function to erase previous stamps.

Create a report of your work using Word (or similar). Make sure you include:

- an introduction
- the theory behind stereograms
- an explanation of the scripts you used in Scratch to make the stereogram
- an explanation of why the viewer sees a 3-dimensional image when he/she views the stereogram correctly
- at least one large, clear stereogram

All work should be your own.

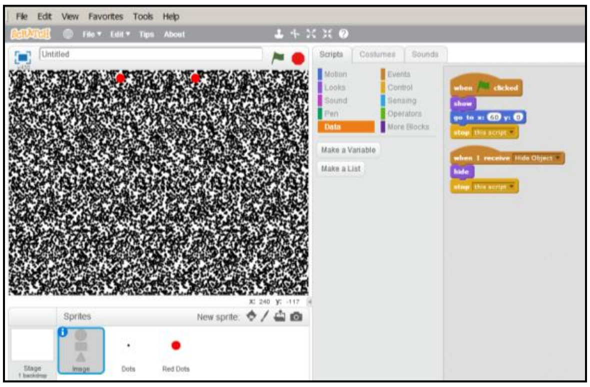
Teacher Notes

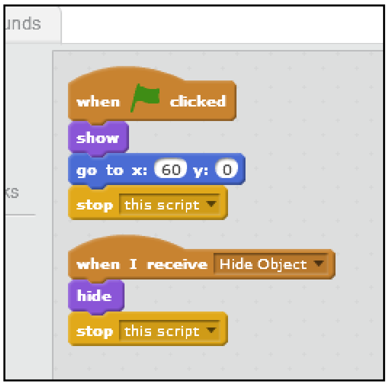
- The example produces a circle, rectangle and triangle (from top to bottom).

- There are two different ways to view a stereogram. The first is described in this document. The second is to imagine focusing on something further away until the black dots perfectly overlap and the eyes become focused.

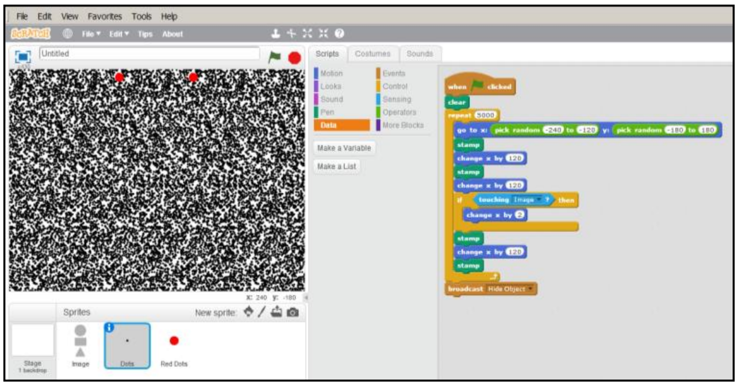
  If we use this method then we should apply the horizontal offset to the left, and not the right.

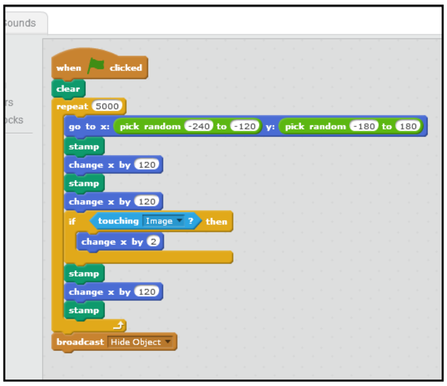
- The following scripts show the simplest way of creating a stereogram in Scratch:



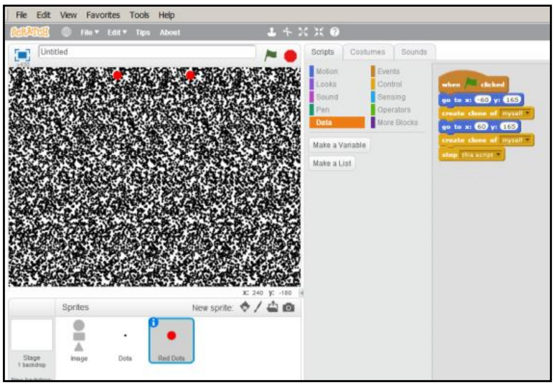


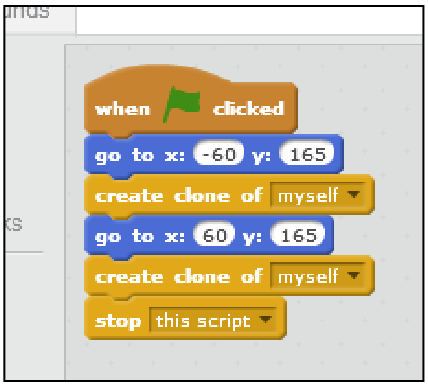
```
when [flag] clicked
show
go to x: 60 y: 0
stop this script

when I receive Hide Object
hide
stop this script
```





```
when [flag] clicked
clear
repeat 5000
    go to x: (pick random -240 to -120) y: (pick random -180 to 180)
    stamp
    change x by 120
    stamp
    change x by 120
    if touching Image ? then
        change x by 2
    stamp
    change x by 120
    stamp
broadcast Hide Object
```





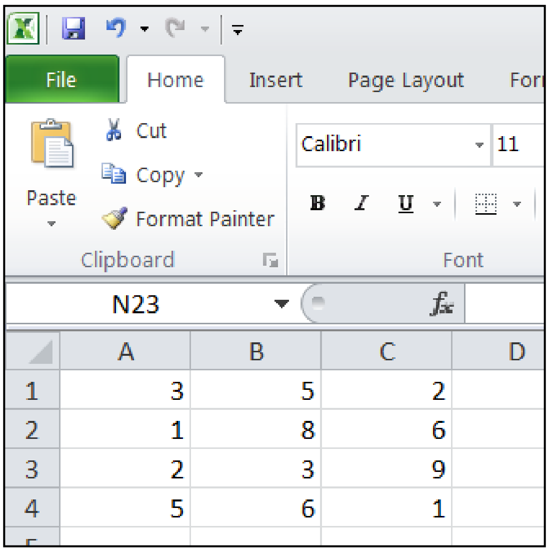
```
when [flag] clicked
go to x: -60 y: 165
create clone of myself
go to x: 60 y: 165
create clone of myself
stop this script
```

- Note that the *clone* function was used for the red dots because when the green flag is clicked often the *clear* function is called after the two red dots have been drawn. If we had used the *stamp* function for the red dots then the *clear* function would've erased them.

Excel Tutorial

Relative References

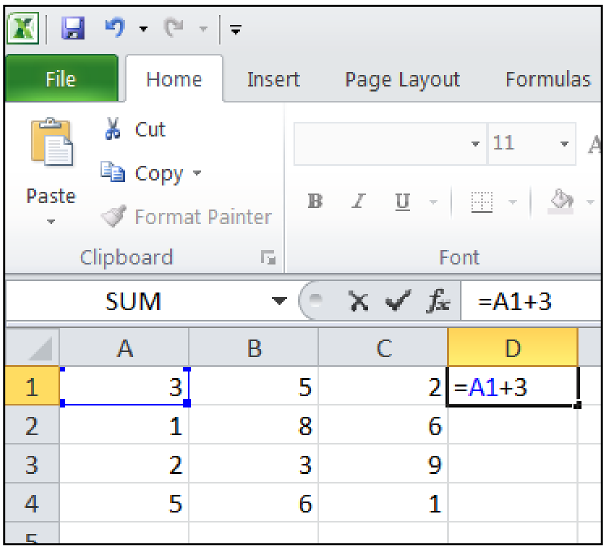
Type the following values into cells A1 to C4:



N23 fx

	A	B	C	D
1	3	5	2	
2	1	8	6	
3	2	3	9	
4	5	6	1	

Type the following equation into cell D1:

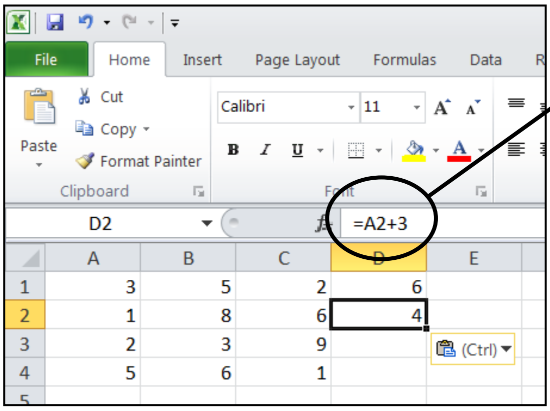


SUM fx =A1+3

	A	B	C	D
1	3	5	2	=A1+3
2	1	8	6	
3	2	3	9	
4	5	6	1	

Press *Enter* and the value displayed in cell D1 should be equal to 6.

Copy and paste cell D1 to cell D2:



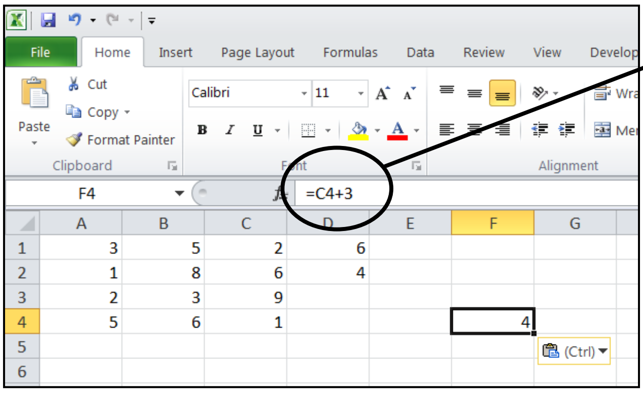
D2 fx =A2+3

	A	B	C	D	E
1	3	5	2	6	
2	1	8	6	4	
3	2	3	9		
4	5	6	1		

Notice that the equation in cell D2 becomes A2+3.

This is because we pasted one row below cell D1, so the reference A1 becomes A2 (the row number increases by one).

Copy and paste cell D1 into cell F4:



F4 fx =C4+3

	A	B	C	D	E	F	G
1	3	5	2	6			
2	1	8	6	4			
3	2	3	9				
4	5	6	1			4	
5							
6							

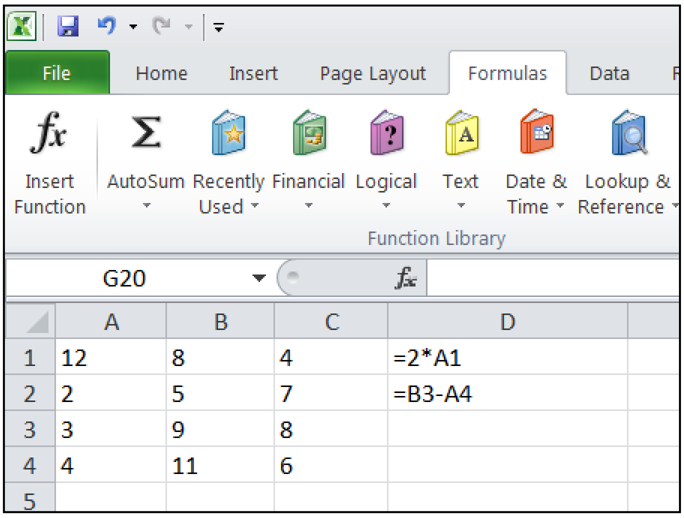
Notice that the equation in cell F4 becomes C4+3.

This is because we pasted two columns to the right and three rows below cell D1, so the reference A1 becomes C4 (the column letter increases by two and the row number increases by three).

These are examples of *relative references*.

Task 1

The following screenshot shows the contents of various cells (Note that you will not see the equations on your screen after you press *Enter*).



	A	B	C	D
1	12	8	4	=2*A1
2	2	5	7	=B3-A4
3	3	9	8	
4	4	11	6	
5				

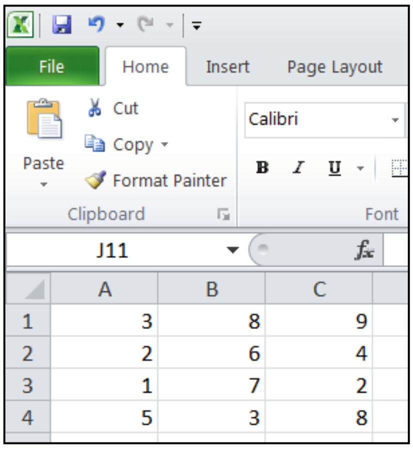
Complete the following table *without* using Excel. Then, check your answers using Excel.

For example, if we copy and paste cell D1 to cell E2, cell E2 will contain the equation 2*B2 and will display the value of 10.

Copied From	Pasted To	Displayed Equation	Displayed Value
D1	E2	2*B2	10
D1	D4		
D1	E3		
D2	E1		
D2	F2		
D2	E3		

Absolute References

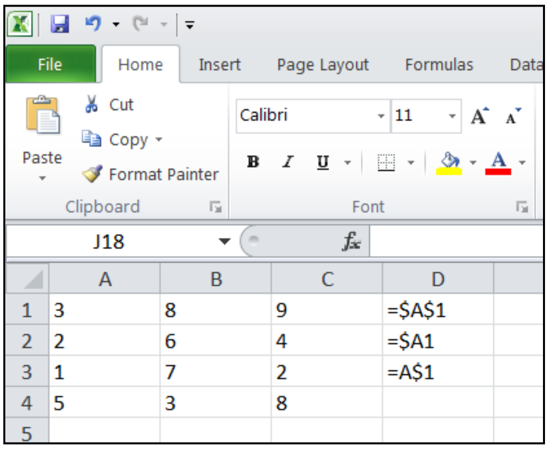
Type the following values in cells A1 to C4:



J11

	A	B	C
1	3	8	9
2	2	6	4
3	1	7	2
4	5	3	8

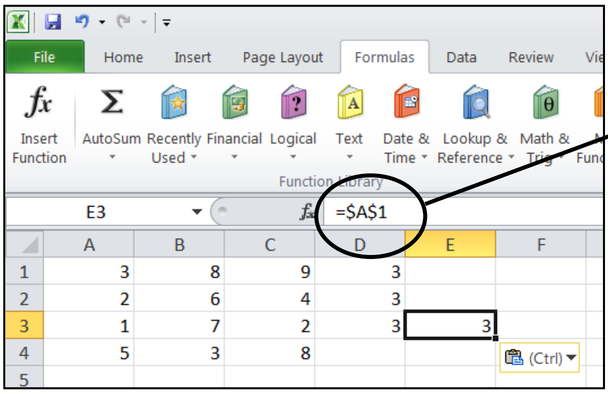
Type the following equations into cells D1 to D3:



J18

	A	B	C	D
1	3	8	9	=$A$1
2	2	6	4	=$A1
3	1	7	2	=A$1
4	5	3	8	
5				

Copy and paste cell D1 to E3:



E3 =$A$1

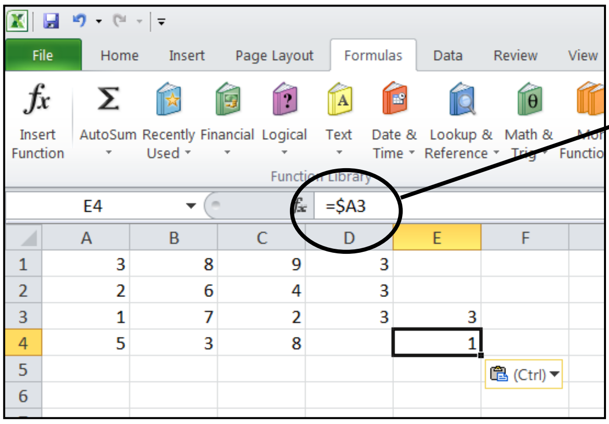
	A	B	C	D	E	F
1	3	8	9	3		
2	2	6	4	3		
3	1	7	2	3	3	
4	5	3	8			
5						

Notice that even though we pasted to a new column and row, the cell reference doesn't change.

Placing a $ sign in front of the column letter means the column letter will never change.

Placing a $ sign in front of the row number means the row number will never change.

Copy and paste cell D2 to E4:



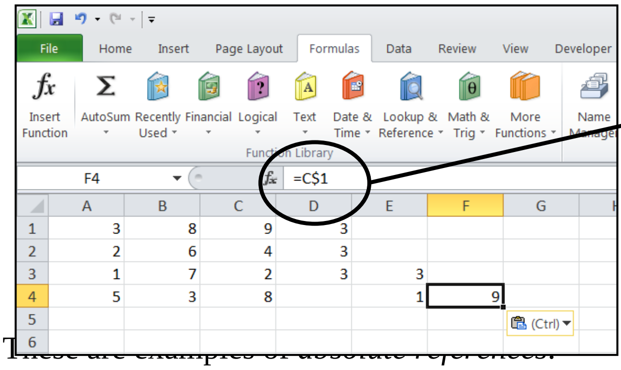
E4 =$A3

	A	B	C	D	E	F
1	3	8	9	3		
2	2	6	4	3		
3	1	7	2	3	3	
4	5	3	8		1	
5						
6						

Since there is a $ sign in front of the column letter, the column letter will never change.

Cell E4 is two rows below cell D2, so the row number increases by two.

Copy and paste cell D3 into cell F4:



F4 =C$1

	A	B	C	D	E	F	G
1	3	8	9	3			
2	2	6	4	3			
3	1	7	2	3	3		
4	5	3	8		1	9	
5							
6							

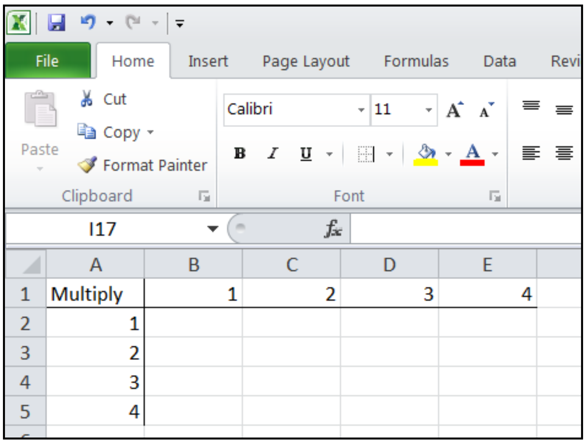
Since there is a $ sign in front of the row number, the row number will never change.

Cell F4 is two columns to the right of cell D3, so the column letter increases by two.

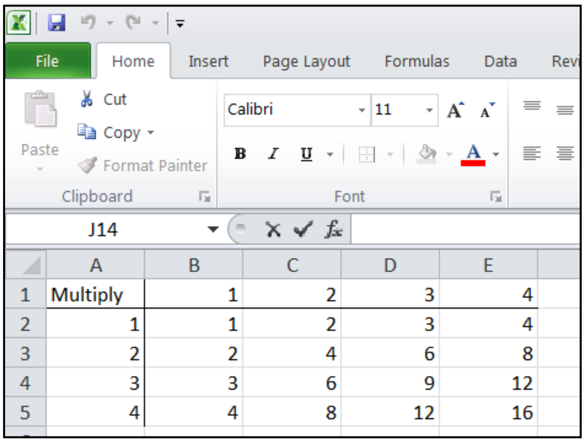
These are examples of *absolute references*.

Task 2

Enter one equation into cell B2, so that when it is copied and pasted into all of the other cells it correctly fills in the multiplication table:



	A	B	C	D	E
1	Multiply	1	2	3	4
2	1				
3	2				
4	3				
5	4				



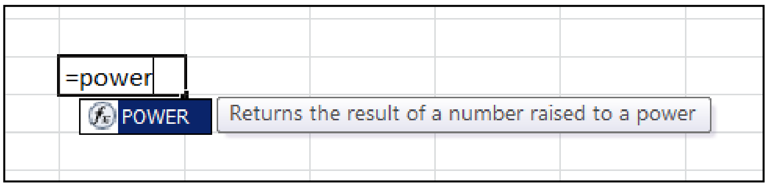
	A	B	C	D	E
1	Multiply	1	2	3	4
2	1	1	2	3	4
3	2	2	4	6	8
4	3	3	6	9	12
5	4	4	8	12	16

Mathematical Functions

Excel contains many mathematical functions. For example:

sin(), cos(), tan(), asin(), acos(), atan(), sqrt(), power(), int(), randbetween(), sum(), average(),countif()

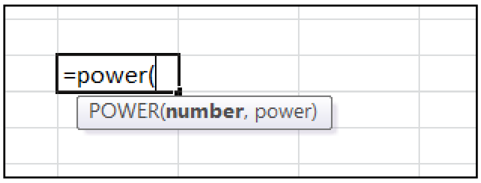
As you type the function into Excel, you will often be given a description of the function and what kind of values it accepts:



=power

POWER

Returns the result of a number raised to a power



=power(

POWER(**number**, power)

Examples

- Typing =power(3,4) will display the value of 81
- Type =sqrt(9) will display the value of 3
- Typing =sum(A1:A6) will calculate the sum of the contents of cells A1 to A6

Task 3

Use Excel to determine the following:

a) the square root of 20
b) the mean value of 20, 18, 23, 11, 25 and 19
c) a randomly generated number between 5 and 13
d) the inverse sine of 1
e) the sum of all positive integers less than 10

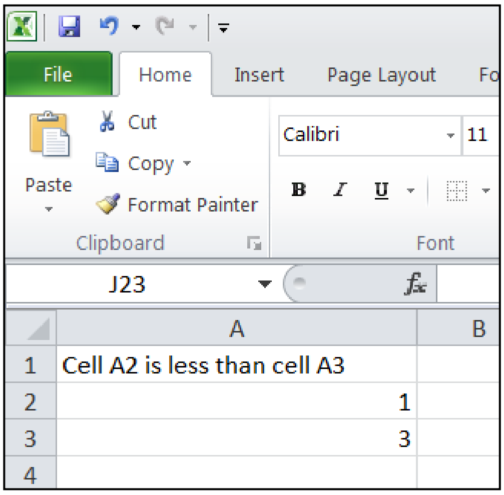
Conditional Logic

Excel contains an IF statement that works according to the following:

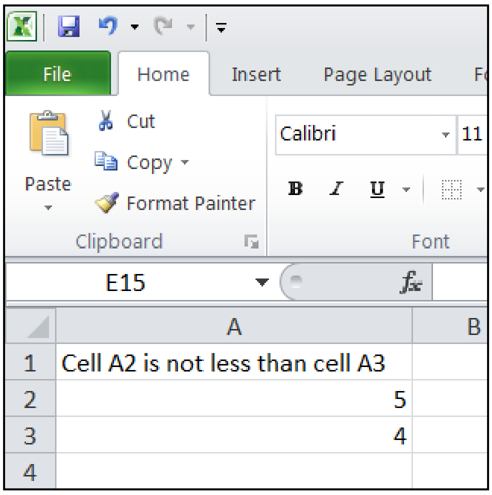
=IF(condition, value to display if true, value to display if false)

For example, in cell A1 type =IF(A2<A3,"Cell A2 is less than cell A3","Cell A2 is not less than cell A3").

Change the contents of cells A2 and A3 and notice what happens to cell A1, depending on the values of cells A2 and A3.



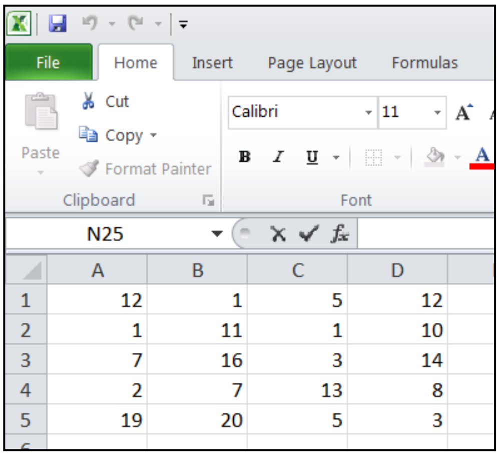
	A	B
1	Cell A2 is less than cell A3	
2	1	
3	3	
4		



	A	B
1	Cell A2 is not less than cell A3	
2	5	
3	4	
4		

Task 4

Without using Excel complete the table based on the screenshot. Then, use Excel to check your answers.



	A	B	C	D
1	12	1	5	12
2	1	11	1	10
3	7	16	3	14
4	2	7	13	8
5	19	20	5	3

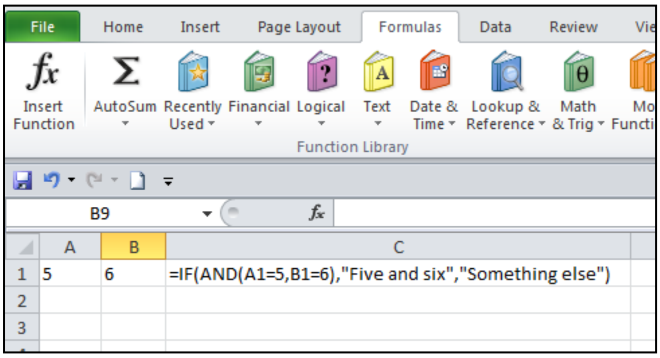
Conditional Statement	Displayed Value
=IF(B1=C1,A4,A3)	7
=IF(C2<B4,"Yes","No")	
=IF(A4=2,"Two","Not two")	
=IF(D5*C3>C5+1,B3,2*D1)	
=IF(SUM(A1:A5)<100,SQRT(4),SQRT(9))	
=IF(AVERAGE(A4:D4)>AVERAGE(B3:D4),0,1)	

Combining Multiple Conditions

We can combine two or more conditions together using AND or OR:

=IF(AND(condition 1, condition 2, …), value if all are true, value if not all are true)

=IF(OR(condition 1, condition 2, …), value if at least one is true, value if none are true)



	A	B	C
1	5	6	=IF(AND(A1=5,B1=6),"Five and six","Something else")
2			
3			

In cell A1 type 5. In cell B1 type 6. In cell A3 type

=IF(AND(A1=5,A2=6),”Five and six”,”Something else”)

Change the value of cells A1 and A2 and notice the text displayed in cell A3. The words “Five and six” will only appear if A1=5 and A2=6.

Replace cell A3 with

=IF(OR(A1=5,A2=6),”Five or six”,”Something else”)

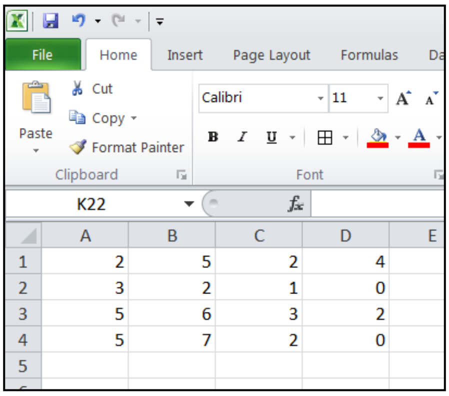
Change the value of cells A1 and A2 and notice the text displayed in cell A3. The words “Five or six” will only appear if A1=5 or A2=6.

Task 5

The screenshot on the right shows the contents of various cells.

Complete the table below without using Excel. Then, use Excel to check your answers.

The first row has been completed as an example.



	A	B	C	D	E
1	2	5	2	4	
2	3	2	1	0	
3	5	6	3	2	
4	5	7	2	0	
5					

Formula	Displayed Value
=IF(AND(A1=2,B3=6),3,5)	3
=IF(OR(C1=2,B4=7),”Yes”,”No”)	
=IF(OR(SQRT(A2)>2,D4*B1=0),”Red”,”Yellow”)	
=IF(AND(AVERAGE(C2,B4)=3,SUM(C1:C4)=8),9,2)	
=IF(OR(B2>C3,A4<B4),23,87)	
=IF(AND((5-D4)^2+(3-B3)^2>4,B4*C3<D2*A3),”Black”,”White”)	

Nested IF Functions

It is possible to put one IF function inside another IF function. For example:

=IF(A1<0,0,IF(A1>10,10,A1))

To understand what will happen, go through the function step by step.

1. If the value of A1 is less than 0, then the value 0 will be displayed.
2. Otherwise (i.e. A1 is not less than 0) the function IF(A1>10,10,A1) will be applied.
3. This means if A1 is greater than 10 the value 10 will be displayed.
4. Otherwise the value of A1 will be displayed.

Task 6

For the function above, determine what value will be displayed *without* using Excel if the value of cell A1 is equal to the following:

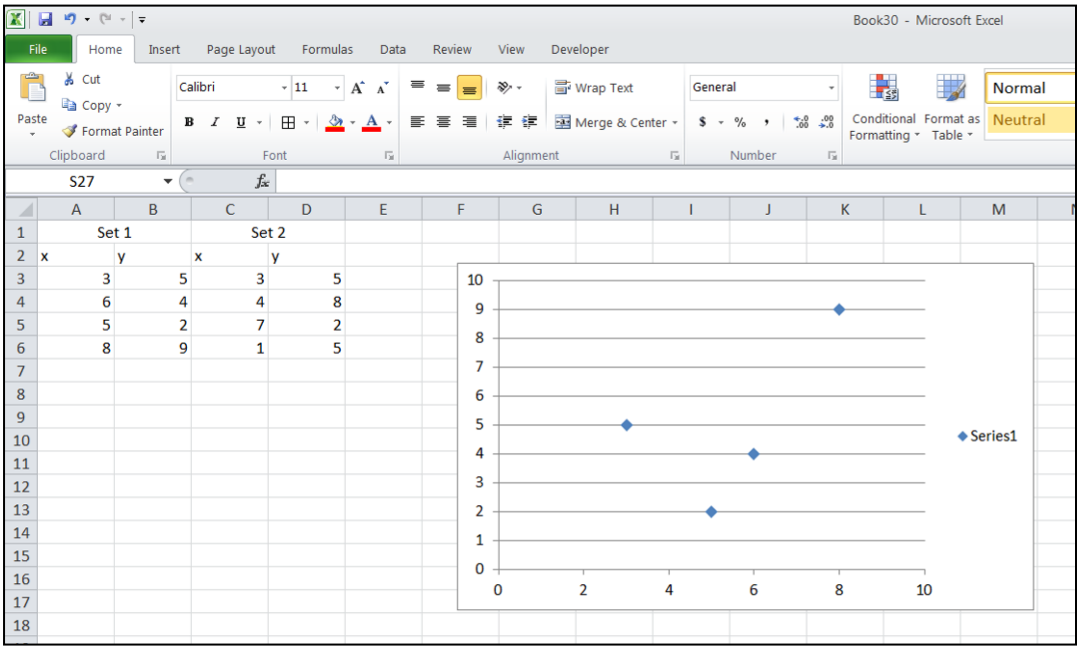
a) -3
b) 0
c) 13
d) 5
e) -10
f) 10
g) 11
h) 9
i) 3.4

Check your answers using Excel

Creating Scatterplots

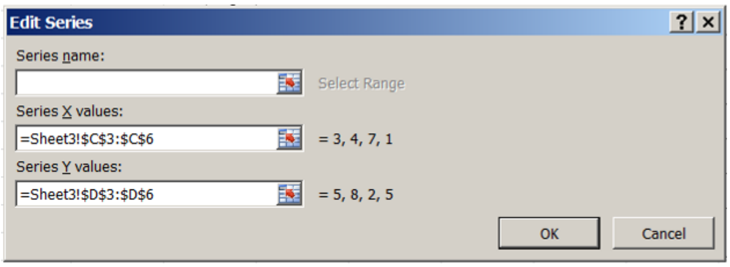
Suppose we wish to enter two sets of data into a scatterplot.

Enter the following into Excel. Highlight cells A3 to B6 and click *Insert* > *Scatter* > *Scatter with only markers*.



	A	B	C	D
1	Set 1		Set 2	
2	x	y	x	y
3	3	5	3	5
4	6	4	4	8
5	5	2	7	2
6	8	9	1	5

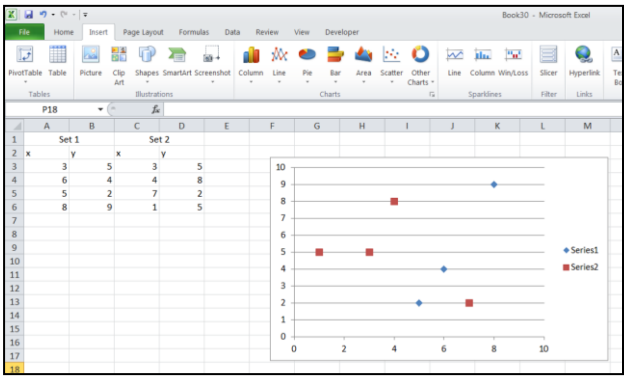
Right-click on the scatterplot and click *Select Data* > *Add*. Place the cursor in the *Series X Values* box and then highlight cells C3 to C6. Place the cursor in the *Series Y Values* box, delete the current contents, then highlight cells D3 to D6. Click *Okay* twice.



Series name:

Series X values: =Sheet3!$C$3:$C$6 = 3, 4, 7, 1

Series Y values: =Sheet3!$D$3:$D$6 = 5, 8, 2, 5



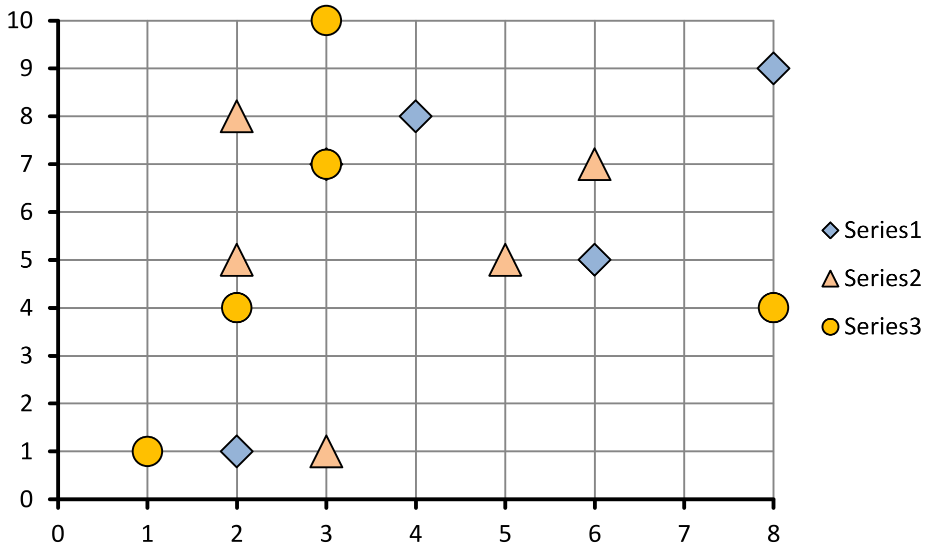
	A	B	C	D
1	Set 1		Set 2	
2	x	y	x	y
3	3	5	3	5
4	6	4	4	8
5	5	2	7	2
6	8	9	1	5

Note that you can change the appearance of the data points by right-clicking on them and selecting *Format Data Series*.

These steps can be repeated to enter more sets of data, should it be necessary.

Task 7

Recreate the following graph as accurately as possible in Excel, including the gridlines, the scale on each axis, the size, shape and colour of the points etc.

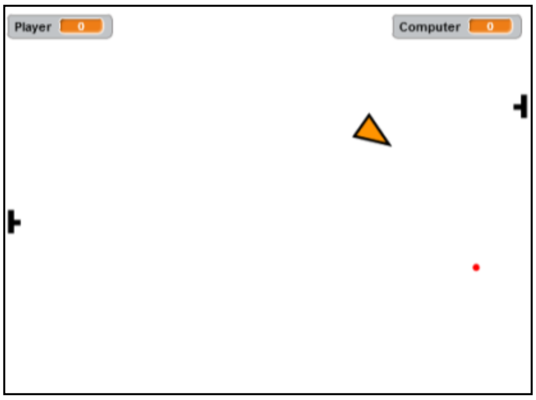


Tip: To change the appearance of a graph, click on it and select relevant options from the Layout tab.

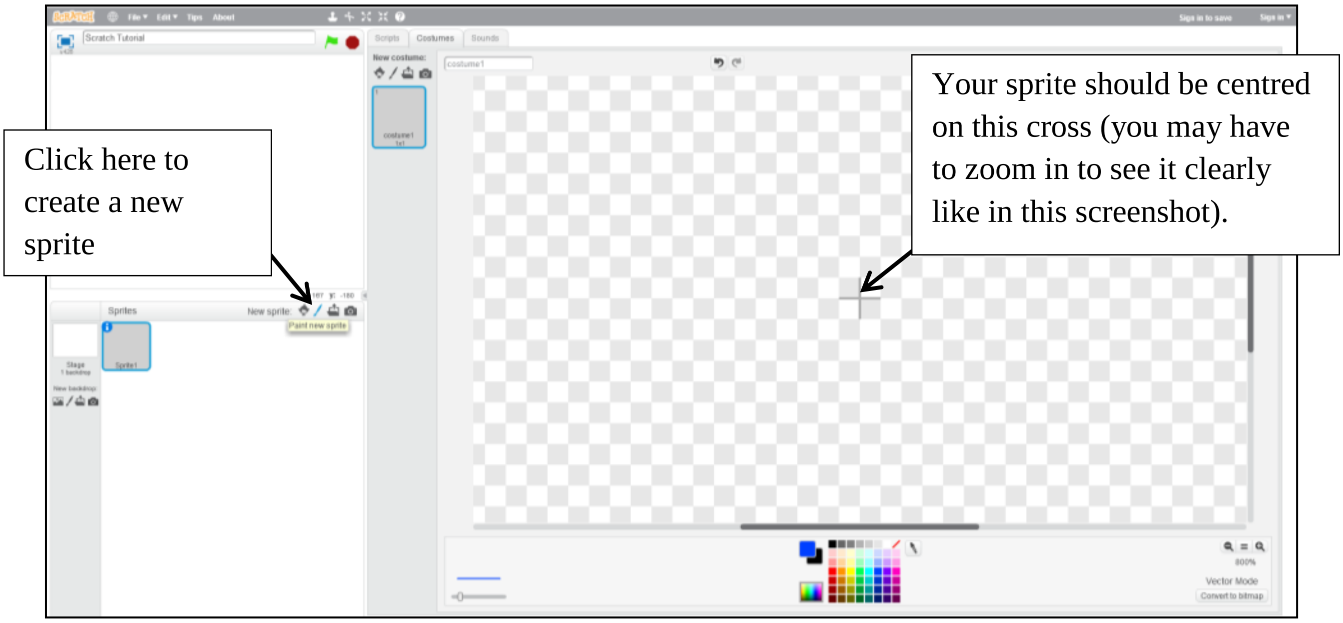
Scratch Tutorial

We will use Scratch to create a game with the following rules:

- The player controls a spaceship. Pressing *up* makes the spaceship accelerate, pressing *down* makes the spaceship decelerate, press *left* makes the spaceship turn left, pressing *right* makes the spaceship turn right, and pressing *space* makes the spaceship fire a rocket.
- On each side of the screen the computer will control enemy spaceships. They will move vertically and randomly fire rockets across the screen.
- If the play gets hit by a rocket the computer's score will increase by 1. If the player shoots an enemy spaceship the player's score will increase by 1.



Creating Sprites



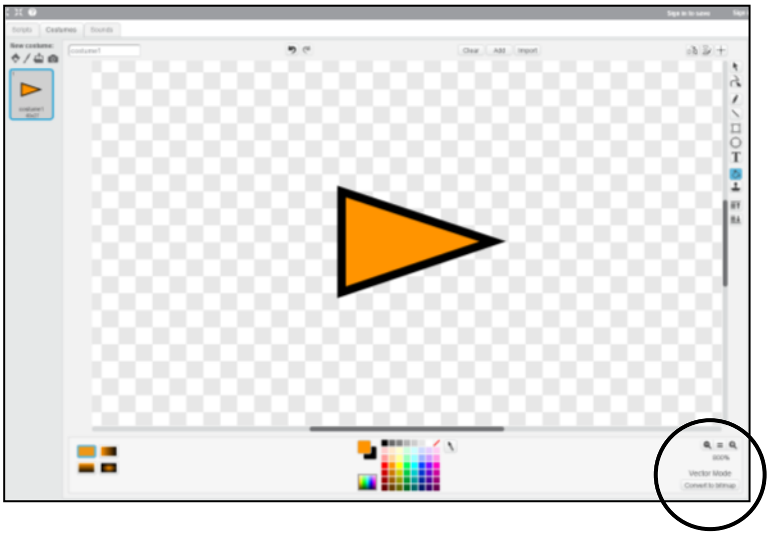
Click here to create a new sprite

Your sprite should be centred on this cross (you may have to zoom in to see it clearly like in this screenshot).

Draw a triangle to represent your spaceship. The spaceship should be pointing to the right.

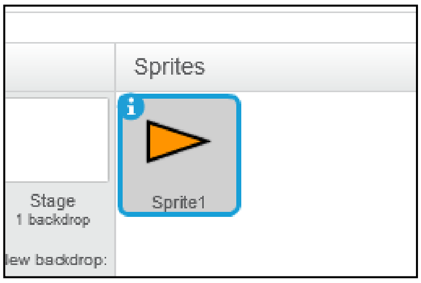
You may need to zoom in to get a better view.

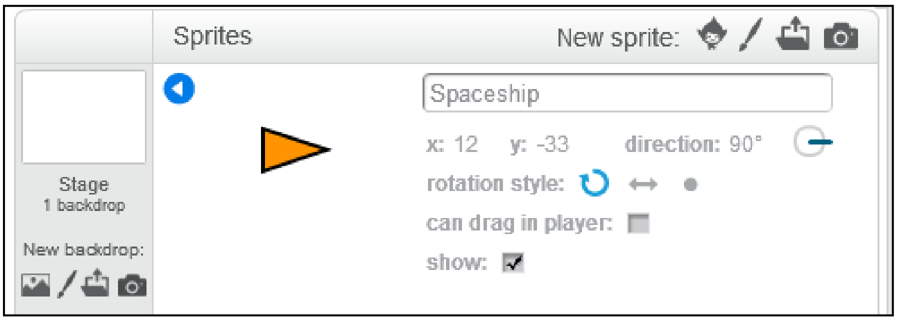
Note that the quality of the sprite looks better if you create it in *vector mode* (circled in the screenshot on the right).



The default name of the sprite will be *Sprite*.

1. To change this into something more descriptive click on the blue and white *i*.
2. Change the name of the sprite to *Spaceship*.
3. Then click on the blue and white arrow to go back.





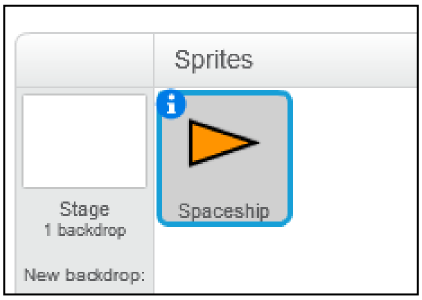
Spaceship

x: 12 y: -33 direction: 90°

rotation style:

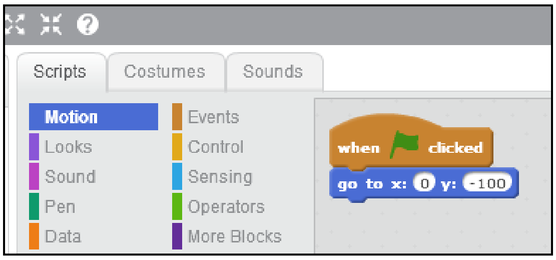
can drag in player: □

show: ☑



Controlling Sprites - Positioning

When the game starts we will position the spaceship just below the centre of the screen. Click on the *Scripts* tab and enter the following code:



```
when 🏳 clicked
go to x: 0 y: -100
```

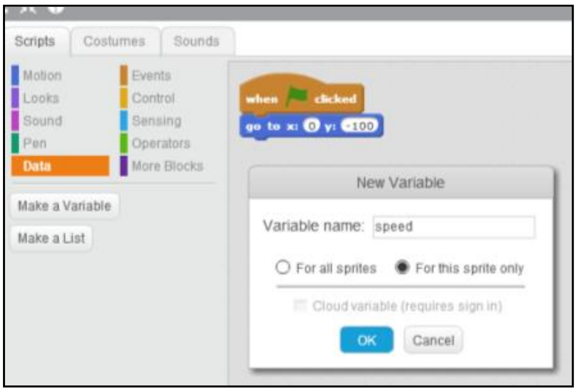
This tells Scratch to position the spaceship at coordinates (0,-100) when the green flag is clicked to start the game.

Note that the scripts are colour coded to make finding them easier.

We need a way to control the speed of the spaceship, and the direction in which it is facing.

Variables

Click *Make a Variable* from the *Data* menu and create a variable called *speed* for this sprite only (this means only the spaceship will be able to access this variable). This will be used to control the speed of the spaceship.



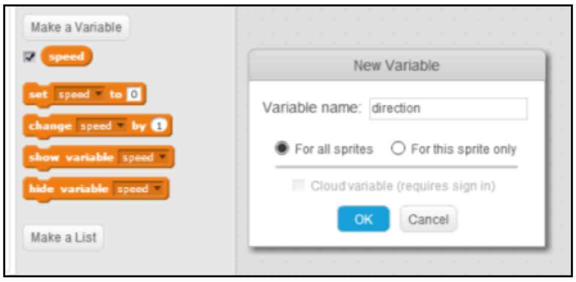
```
when 🏳 clicked
go to x: 0 y: -100
```

Variable name: speed

○ For all sprites ● For this sprite only

□ Cloud variable (requires sign in)

Create another variable called *direction* for all sprites. This means that other sprites will be able to access it and read and change its value. This will be used to control the direction in which the ship is facing, and also the direction at which it will fire its rockets.



```
☑ speed
set speed ▾ to 0
change speed ▾ by 1
show variable speed ▾
hide variable speed ▾
```

Variable name: direction

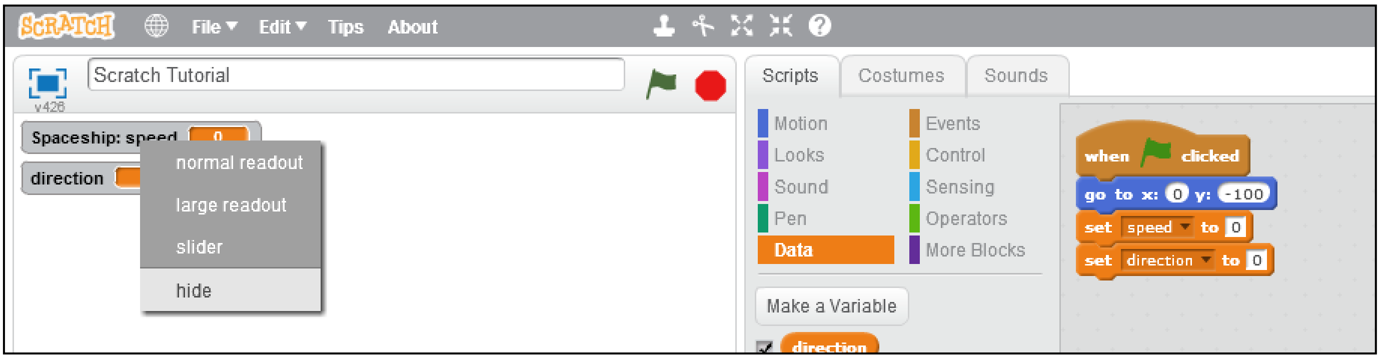
● For all sprites ○ For this sprite only

□ Cloud variable (requires sign in)

When the game first starts the value of *speed* should be 0 and the value of *direction* should be 0.

Modify your existing script to make this happen, as in the screenshot below.

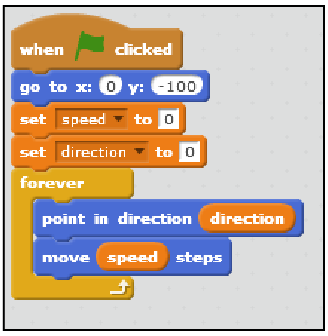
Notice that the values of the variables are displayed on the screen. To hide these values right-click on them and select *hide*.



```
when clicked
go to x: 0 y: -100
set speed to 0
set direction to 0
```

Controlling Sprites – Moving and Rotating

To animate the motion of a sprite we can use a loop which will repeat what is inside it indefinitely.



```
when clicked
go to x: 0 y: -100
set speed to 0
set direction to 0
forever
    point in direction direction
    move speed steps
```

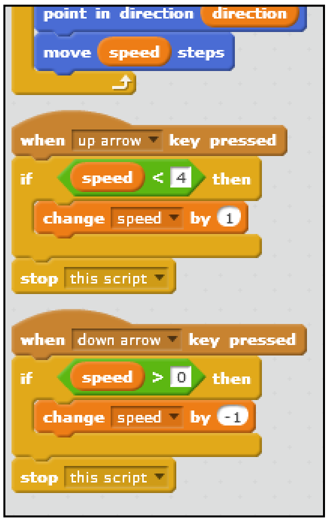
Modify your code as in the screenshot on the right. This will continuously point the spaceship in the direction of the variable *direction*, and move it forward by *speed* pixels.

If you click the green flag the spaceship should now be facing upwards. However, it will not move because the value of speed is still equal to zero.

Reacting to User Input

When the up arrow is pressed we need the spaceship to accelerate i.e. we need the value of *speed* to increase.

When the down arrow is pressed we need the spaceship to decelerate i.e. we need the value of *speed* to decrease.



```
    point in direction direction
    move speed steps

when up arrow key pressed
if speed < 4 then
    change speed by 1
stop this script

when down arrow key pressed
if speed > 0 then
    change speed by -1
stop this script
```

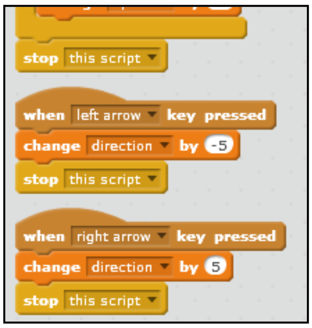
We need to be careful that we do not increase the speed by too much otherwise the spaceship will be uncontrollable. We also need to make sure that when the speed is zero, the ship stops decelerating.

Modify your code as in the screenshot on the right. Notice that this sets the maximum value of *speed* to 4. The *stop* function is not mandatory. However, it makes the code look neater as it marks the end of this particular script. If it were not there and we went back to look at the code in the future, we may think that we have accidentally deleted some code since the script doesn't end in a neat way. Make sure you only stop *this script* and not *all* or *other scripts in sprite*.

To test to see if everything is working, click the green flag to start the game and hold down the up arrow. The spaceship should move forward. Holding the down arrow should make it slow down.

When the left and right arrows are pressed we need the spaceship to turn left and right.

Modify your code as in the screenshot on the right. You should now be able to steer the spaceship. Test to see if it works by clicking the green flag and controlling the spaceship.

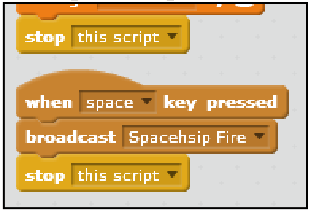


```
stop this script

when left arrow key pressed
change direction by -5
stop this script

when right arrow key pressed
change direction by 5
stop this script
```

Broadcasting

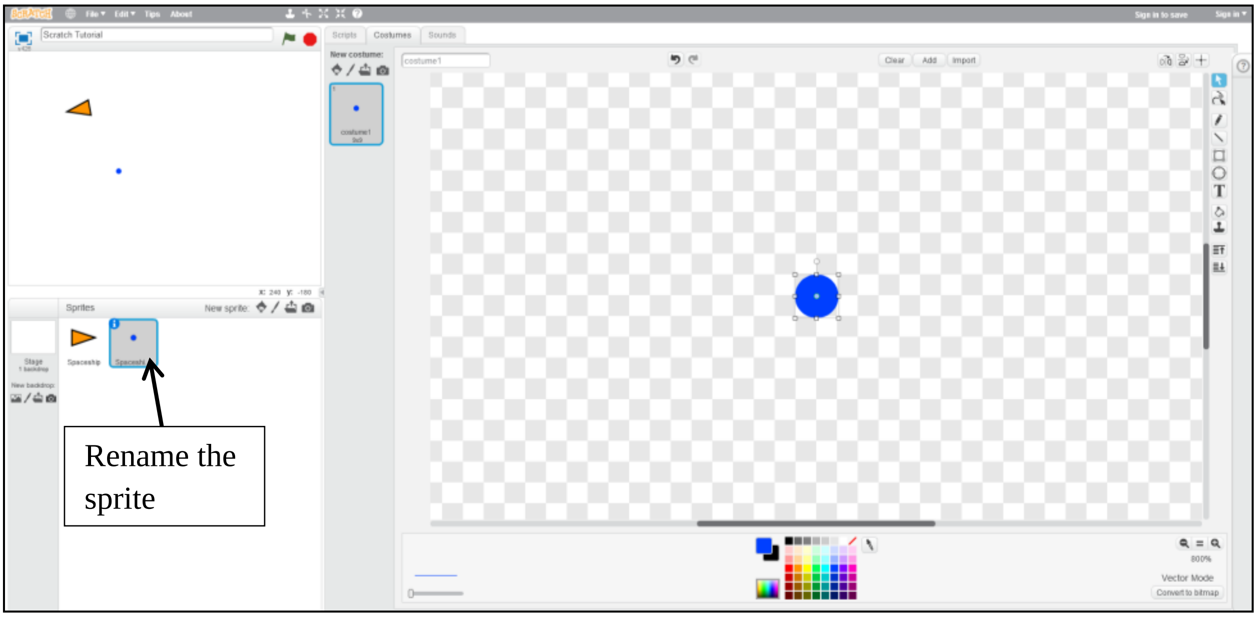
There are often many ways to make something happen in Scratch. To make our spaceship fire when we press *space* we will use the *broadcast* function. We do not have to use this to make the spaceship fire, but now is a good time to see how this function works.

Modify your code as in the screenshot on the right. Note that you should create a new message (by clicking the black downwards arrow) and call it something descriptive such as *Spaceship Fire*.



```
stop this script

when space key pressed
broadcast Spacehsip Fire
stop this script
```

Create a new sprite and draw something to represent your rocket. It can be as simple as a circle. Remember you should draw your sprite in the centre of the window, on top of the gray cross. It is a good idea to rename your sprite into something more descriptive. Rename it to *Spaceship Rocket*



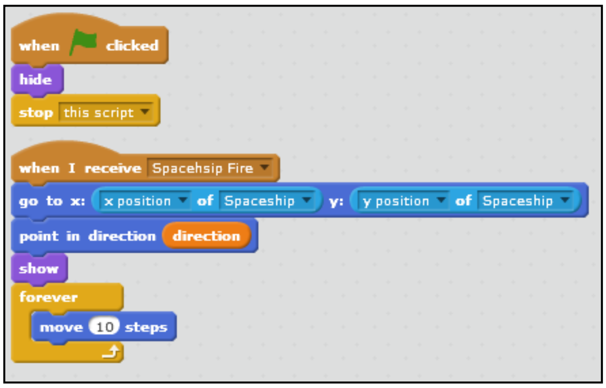
Rename the sprite

When the game first starts the rocket should be hidden. It should only appear when *space* is pressed. When the message *Spaceship Fire* is broadcast the rocket should appear at the same position as the spaceship, and move forward in the same direction as the spaceship is facing.

Click on the *Scripts* tab of this new sprite and add code as in the screenshot on the right.

Now test the script by clicking the green flag to start the game, moving and rotating the spaceship, and firing the rocket.

Notice that you can change the speed of the rocket by changing how much it moves inside the forever loop.



```
when ⚑ clicked
hide
stop this script

when I receive Spacehsip Fire
go to x: x position of Spaceship y: y position of Spaceship
point in direction direction
show
forever
    move 10 steps
```

Troubleshooting

When testing you should notice two things:

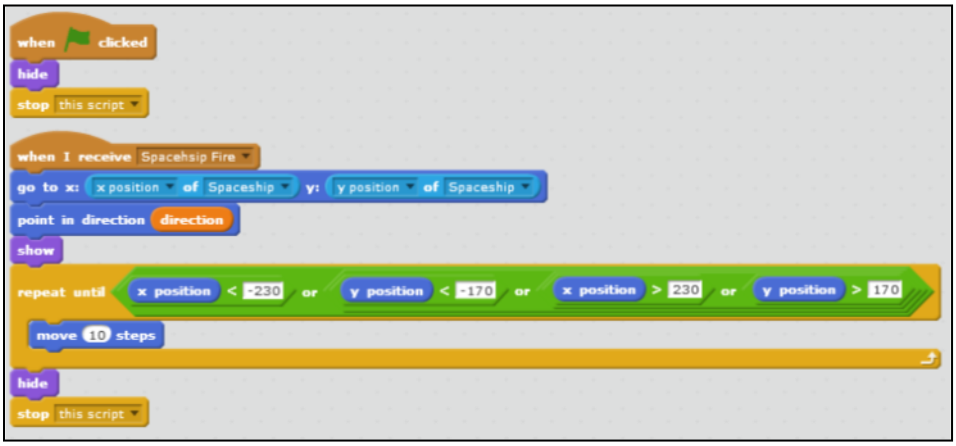
1. When the rocket reaches the edge of the screen it will just stay there and still be partially visible.

2. If you press space at any time the rocket will move back to where the spaceship is and fire again. This does not look good if the rocket has not completed its previous journey as it will just disappear mid-journey to be fired again.

To fix problem 1, *change* the code according the screenshot on the right.

This only moves the rocket if it hasn't reached the edge of the screen.

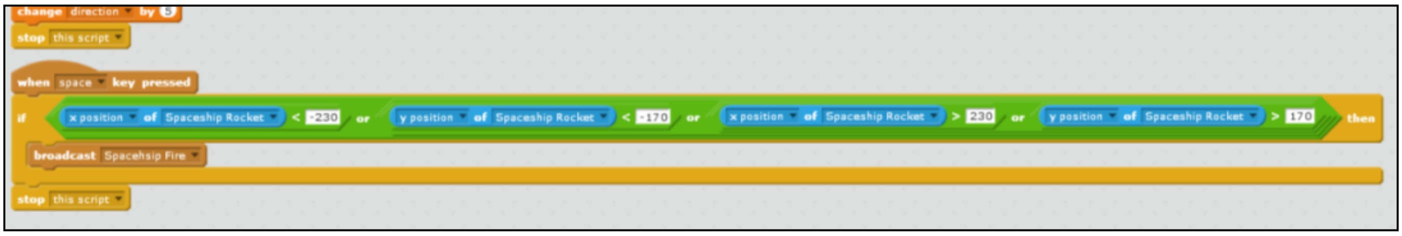
When it does reach the edge of the screen it is hidden.



```
when ⚑ clicked
hide
stop this script

when I receive Spacehsip Fire
go to x: x position of Spaceship y: y position of Spaceship
point in direction direction
show
repeat until x position < -230 or y position < -170 or x position > 230 or y position > 170
    move 10 steps
hide
stop this script
```

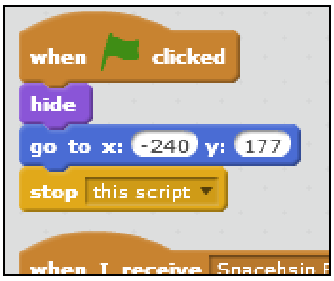
Test the code to see if it works.

To fix problem 2 we need to go back to the scripts for the spaceship. Change the code for when space is pressed to the following:



```
change direction by 5
stop this script

when space key pressed
if x position of Spaceship Rocket < -230 or y position of Spaceship Rocket < -170 or x position of Spaceship Rocket > 230 or y position of Spaceship Rocket > 170 then
    broadcast Spacehsip Fire
stop this script
```

This means that *Spaceship Fire* will only be broadcast if the rocket is currently at the edge of the screen. However, this will cause another problem. When the game first starts the rocket will not necessarily be at the edge of the screen. To fix this, go back to the scripts for the rocket and change to the code on the right.

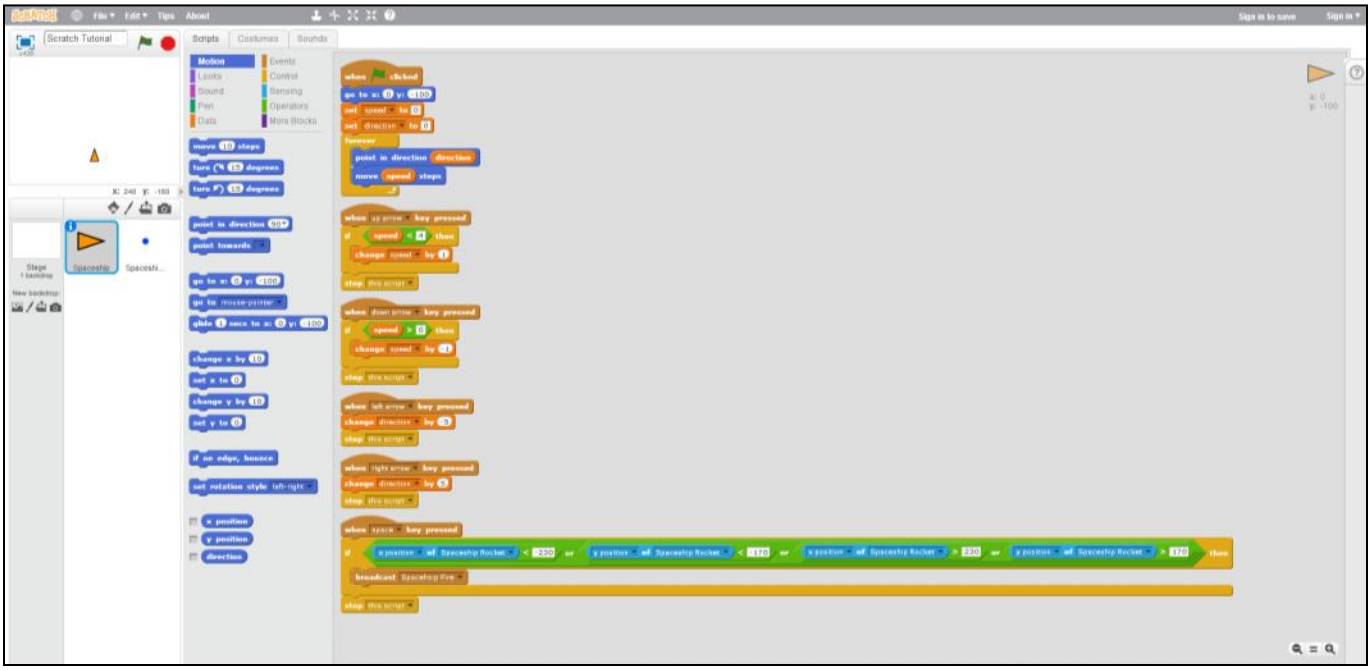


```
when [flag] clicked
hide
go to x: -240 y: 177
stop this script

when I receive Spacehsip F
```

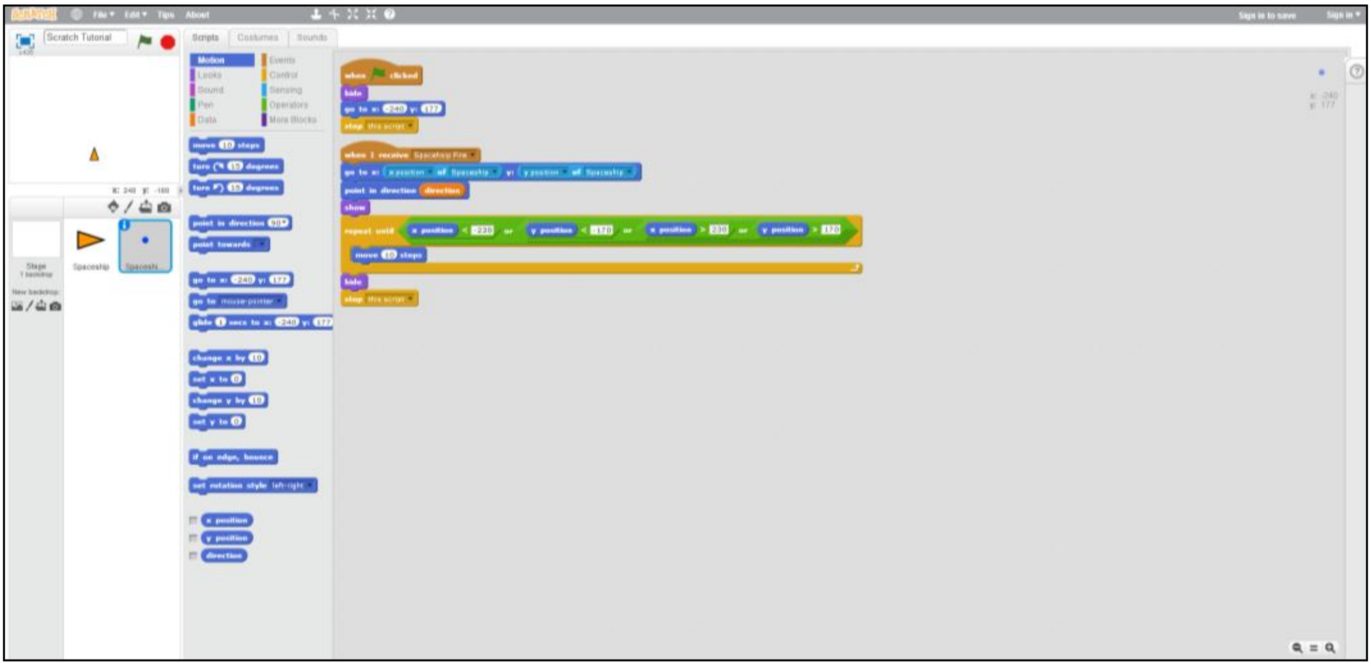
This will place the rocket at the edge of the screen when the game begins. Note that the value of the *y*-coordinate is not important.

Test the game to check that it has fixed the problems we had.

The code for your two sprites should now look similar to the following:



```
when [flag] clicked
go to x: 0 y: -100
set speed to 0
set direction to 0
forever
    point in direction (direction)
    move (speed) steps

when up arrow key pressed
if <(speed) < 4> then
    change speed by 1
stop this script

when down arrow key pressed
if <(speed) > 0> then
    change speed by -1
stop this script

when left arrow key pressed
change direction by -5
stop this script

when right arrow key pressed
change direction by 5
stop this script

when space key pressed
if <<<<(x position of Spaceship Rocket) < -230> or <(y position of Spaceship Rocket) < -170>> or <(x position of Spaceship Rocket) > 230>> or <(y position of Spaceship Rocket) > 170>> then
    broadcast Spaceship Fire
stop this script
```



```
when [flag] clicked
hide
go to x: -240 y: 177
stop this script

when I receive Spaceship Fire
go to x: (x position of Spaceship) y: (y position of Spaceship)
point in direction (direction)
show
repeat until <<<<(x position) < -230> or <(y position) < -170>> or <(x position) > 230>> or <(y position) > 170>>
    move 10 steps
hide
stop this script
```

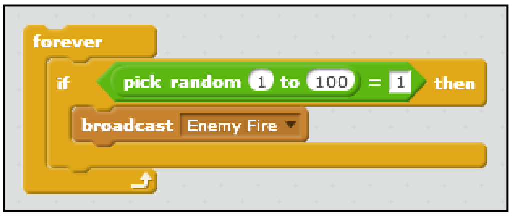
Putting it all Together

Use the skills you have learned to add an enemy to your game on the left side of the screen that obeys the following rules:

- The enemy can only move vertically. If the player is above the enemy it will move up; if it is below it will move down
- The enemy will randomly fire a rocket horizontally across the screen.
- If the enemy's rocket hits the player the computer's score will increase by one; if the player's rocket hits the enemy the player's score will increase by one (use variables for the scores).
- Add another enemy to the right side of the screen that obeys the same rules. To save time you can duplicate the sprites you created for the first enemy and alter the code slightly. Give this enemy a different speed to the other enemy to make the game more interesting (and challenging).

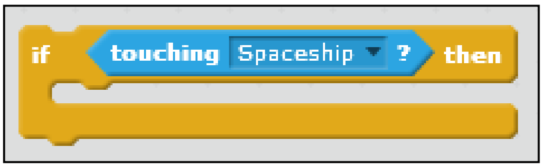
Hints:

To randomly fire a rocket you can use code similar to the screenshot on the right. To fire more or less frequently, adjust the range of numbers from which the random number is chosen. You will also need to make sure that you don't encounter the same problems we encountered when firing our rocket (you should only fire if the previous shot has reached the edge of the screen).



```
forever
    if pick random 1 to 100 = 1 then
        broadcast Enemy Fire
```

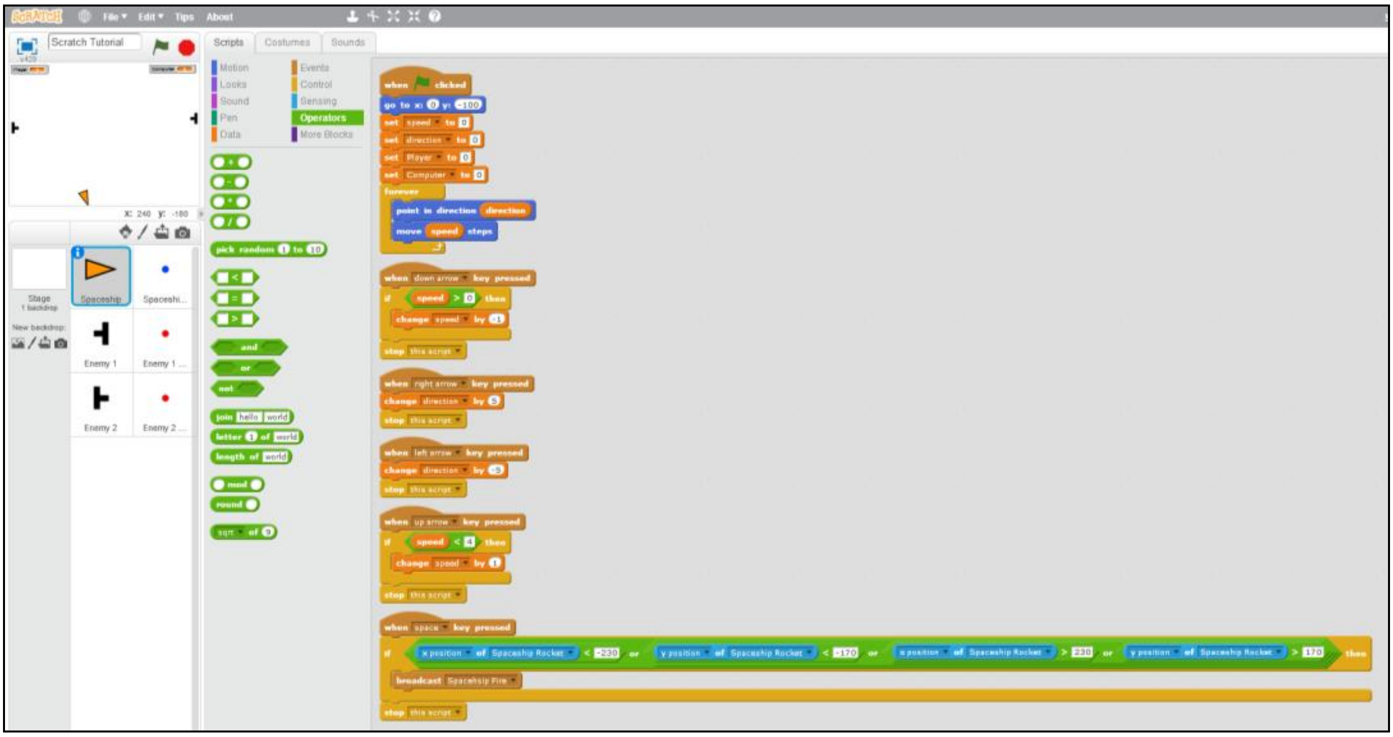
To determine whether a rocket has hit a spaceship use the function below. You will need to check this every time you move the rocket to a new position.



```
if touching Spaceship ? then
```

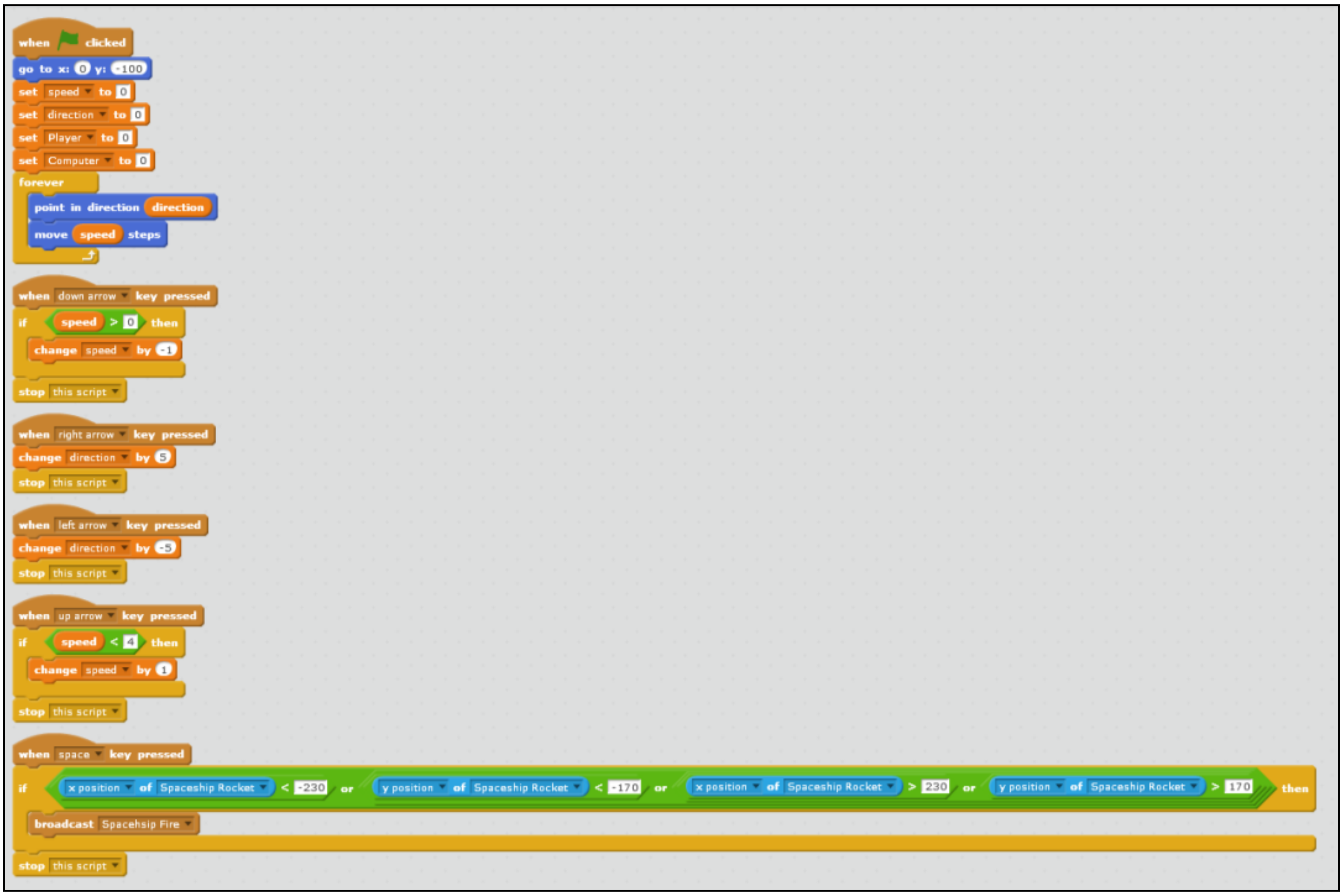
Teacher Notes

There are many different ways to complete this tutorial. The screenshots below show one way.

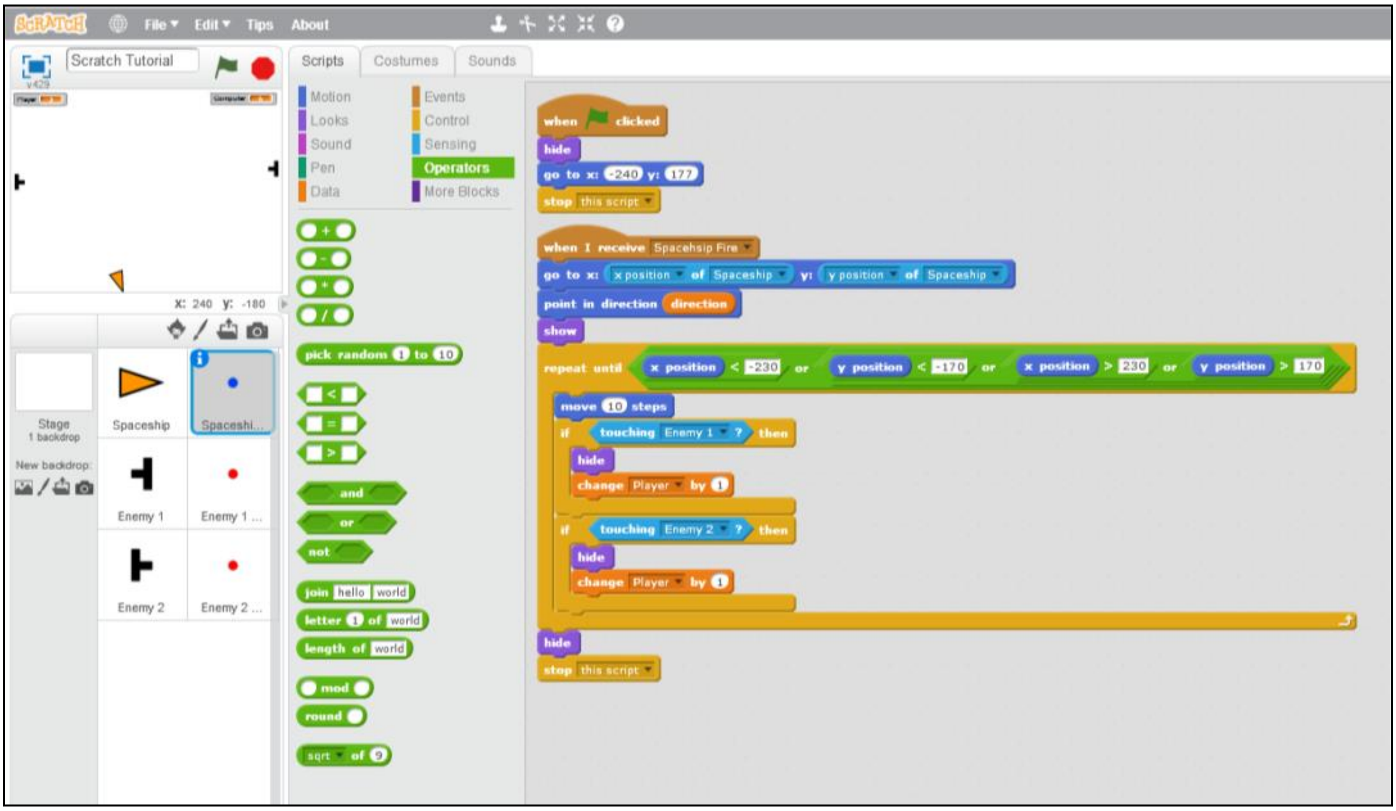


```
when [flag] clicked
go to x: 0 y: -100
set speed to 0
set direction to 0
set Player to 0
set Computer to 0
forever
    point in direction (direction)
    move (speed) steps

when down arrow key pressed
if <(speed) > 0> then
    change speed by -1
stop this script

when right arrow key pressed
change direction by 5
stop this script

when left arrow key pressed
change direction by -5
stop this script

when up arrow key pressed
if <(speed) < 4> then
    change speed by 1
stop this script

when space key pressed
if <(x position of Spaceship Rocket) < -230> or <(y position of Spaceship Rocket) < -170> or <(x position of Spaceship Rocket) > 230> or <(y position of Spaceship Rocket) > 170> then
    broadcast Spacehsip Fire
stop this script
```

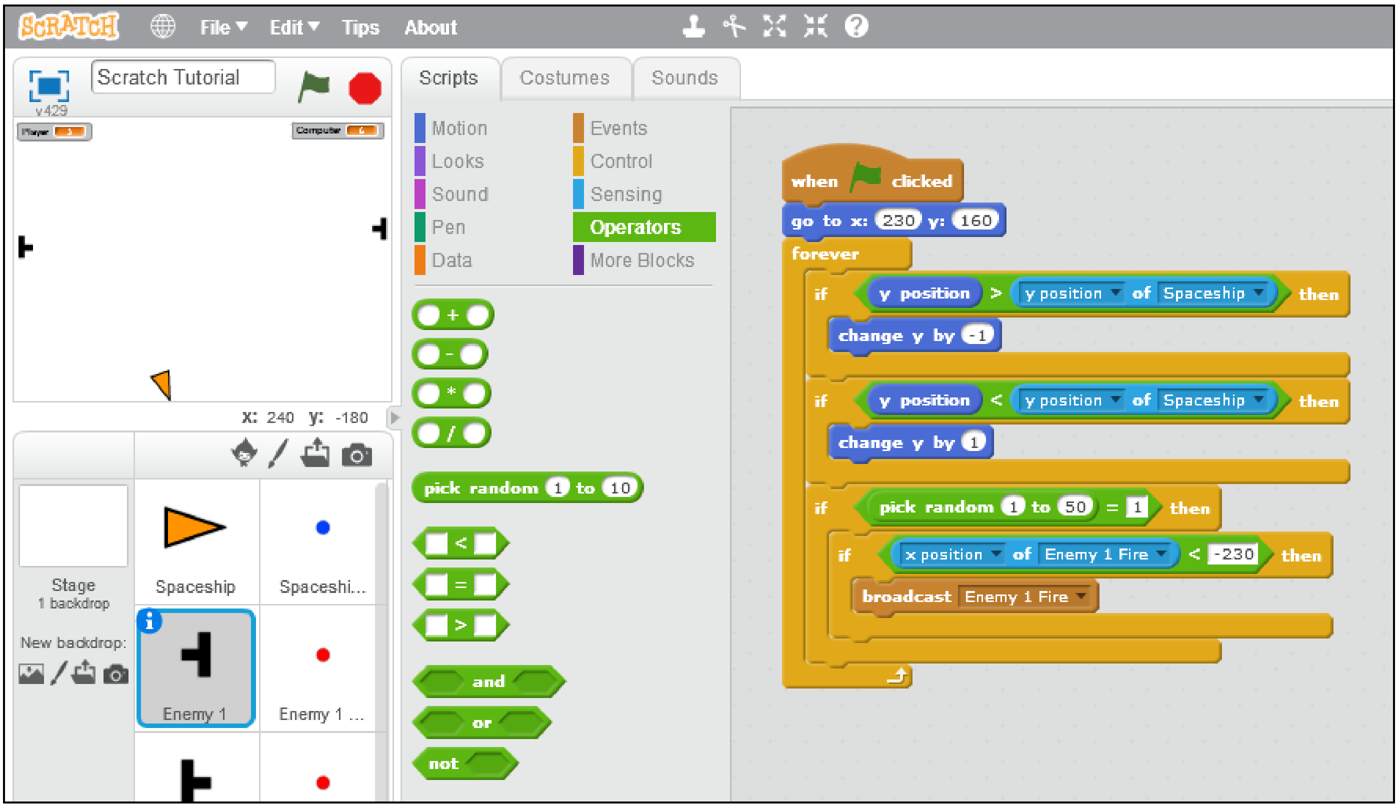
The next screenshot is an enlarged view of the code from above.



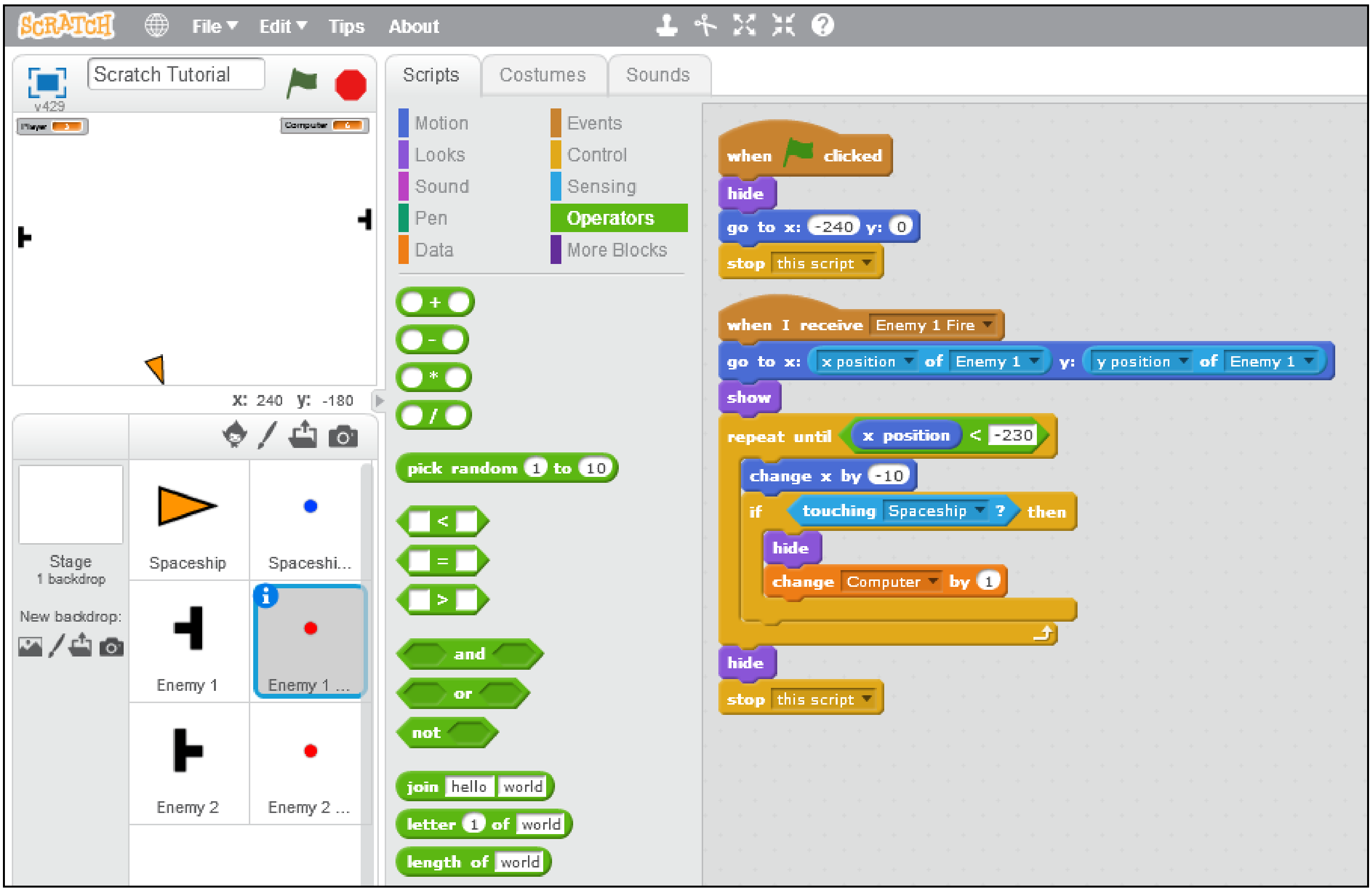
```
when [flag] clicked
go to x: 0 y: -100
set speed to 0
set direction to 0
set Player to 0
set Computer to 0
forever
    point in direction (direction)
    move (speed) steps

when down arrow key pressed
if <(speed) > 0> then
    change speed by -1
stop this script

when right arrow key pressed
change direction by 5
stop this script

when left arrow key pressed
change direction by -5
stop this script

when up arrow key pressed
if <(speed) < 4> then
    change speed by 1
stop this script

when space key pressed
if <(x position of Spaceship Rocket) < -230> or <(y position of Spaceship Rocket) < -170> or <(x position of Spaceship Rocket) > 230> or <(y position of Spaceship Rocket) > 170> then
    broadcast Spacehsip Fire
stop this script
```



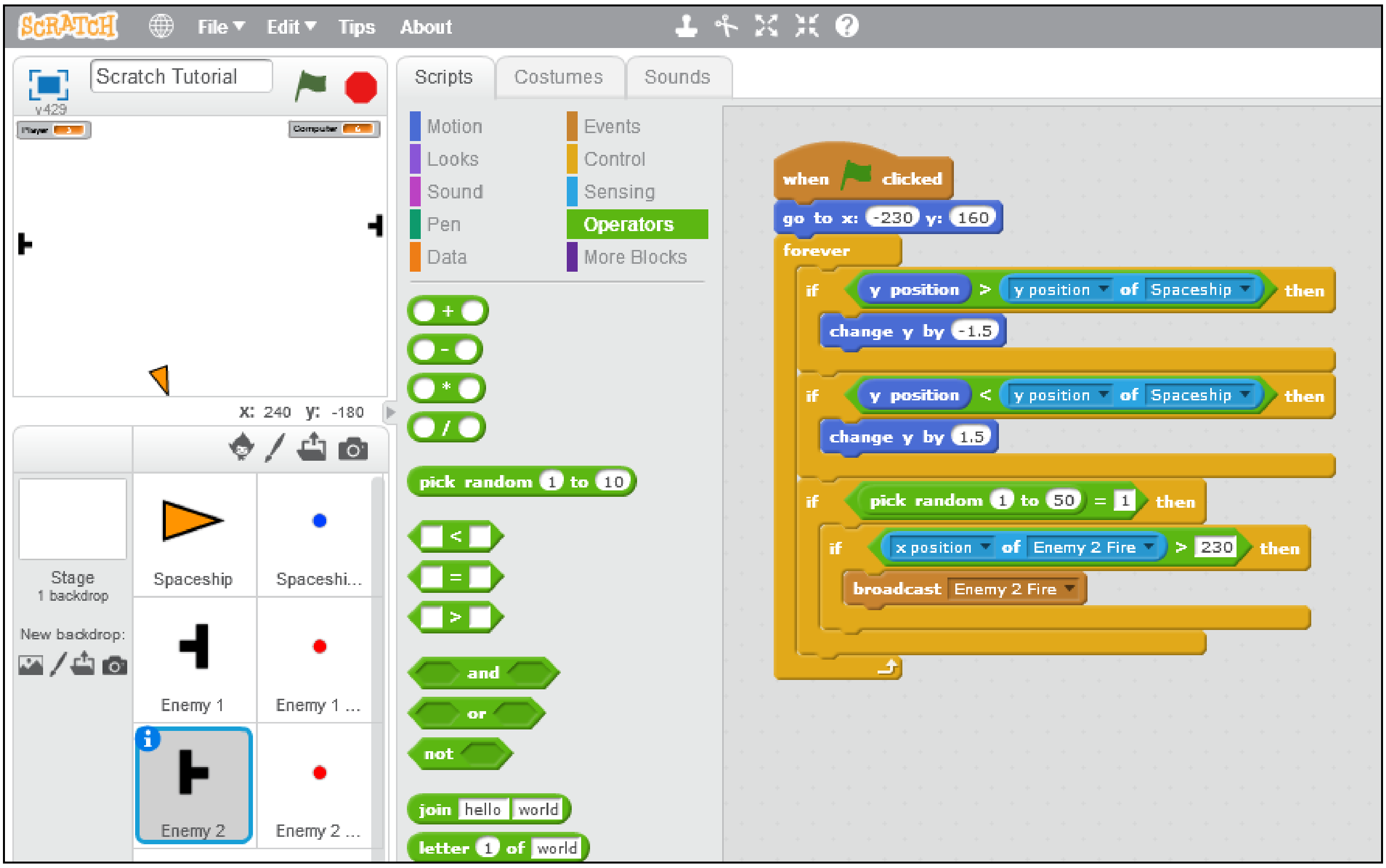
```
when [flag] clicked
hide
go to x: -240 y: 177
stop this script

when I receive Spacehsip Fire
go to x: x position of Spaceship y: y position of Spaceship
point in direction direction
show
repeat until x position < -230 or y position < -170 or x position > 230 or y position > 170
    move 10 steps
    if touching Enemy 1 ? then
        hide
        change Player by 1
    if touching Enemy 2 ? then
        hide
        change Player by 1
hide
stop this script
```



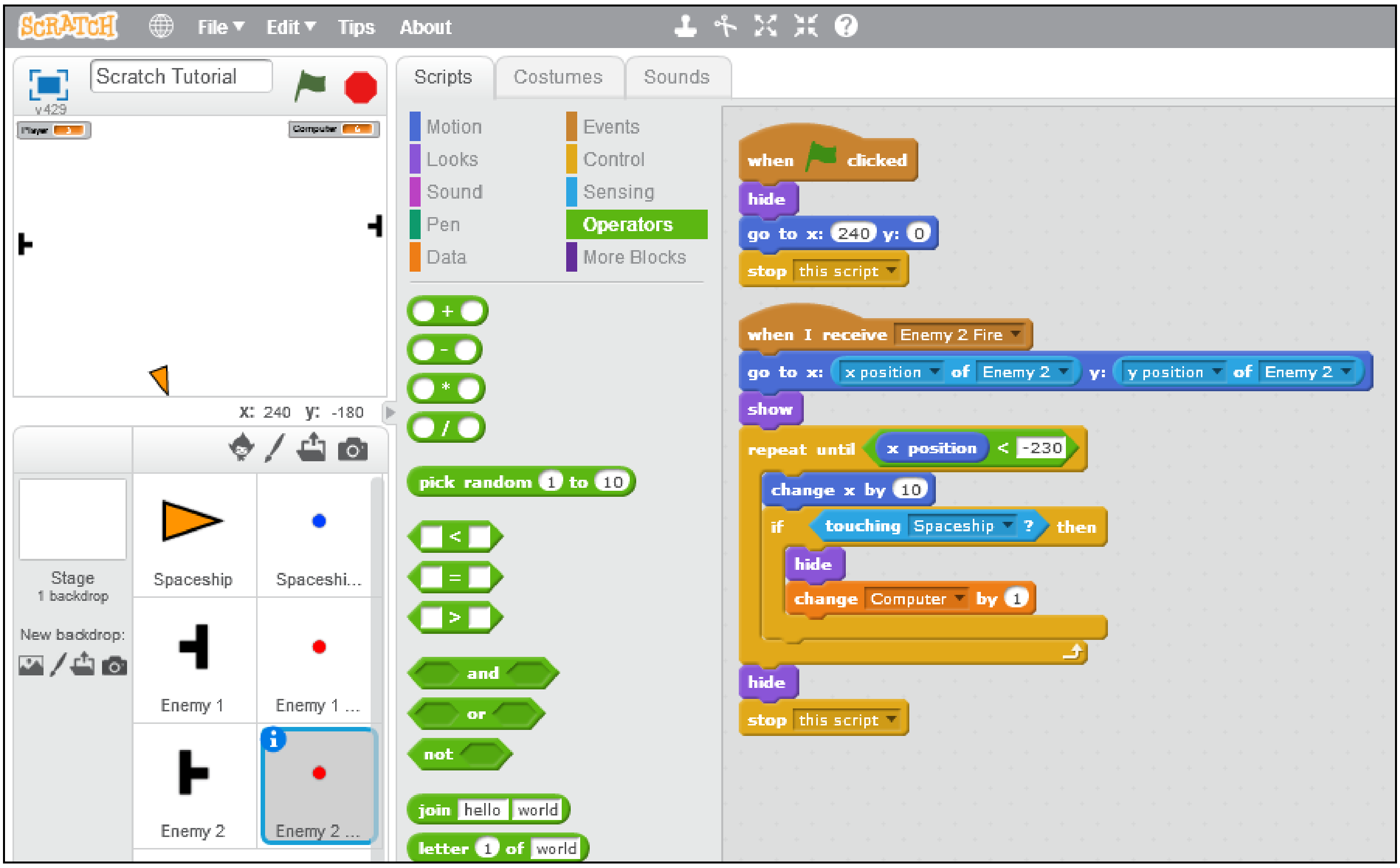
```
when [flag] clicked
go to x: 230 y: 160
forever
    if y position > y position of Spaceship then
        change y by -1
    if y position < y position of Spaceship then
        change y by 1
    if pick random 1 to 50 = 1 then
        if x position of Enemy 1 Fire < -230 then
            broadcast Enemy 1 Fire
```



```
when ⚑ clicked
hide
go to x: -240 y: 0
stop this script

when I receive Enemy 1 Fire
go to x: x position of Enemy 1 y: y position of Enemy 1
show
repeat until x position < -230
    change x by -10
    if touching Spaceship ? then
        hide
        change Computer by 1
hide
stop this script
```



```
when ⚑ clicked
go to x: -230 y: 160
forever
    if y position > y position of Spaceship then
        change y by -1.5
    if y position < y position of Spaceship then
        change y by 1.5
    if pick random 1 to 50 = 1 then
        if x position of Enemy 2 Fire > 230 then
            broadcast Enemy 2 Fire
```



```
when ⚑ clicked
hide
go to x: 240 y: 0
stop this script

when I receive Enemy 2 Fire
go to x: x position of Enemy 2 y: y position of Enemy 2
show
repeat until x position < -230
    change x by 10
    if touching Spaceship ? then
        hide
        change Computer by 1
hide
stop this script
```