Year 3 - My First Program

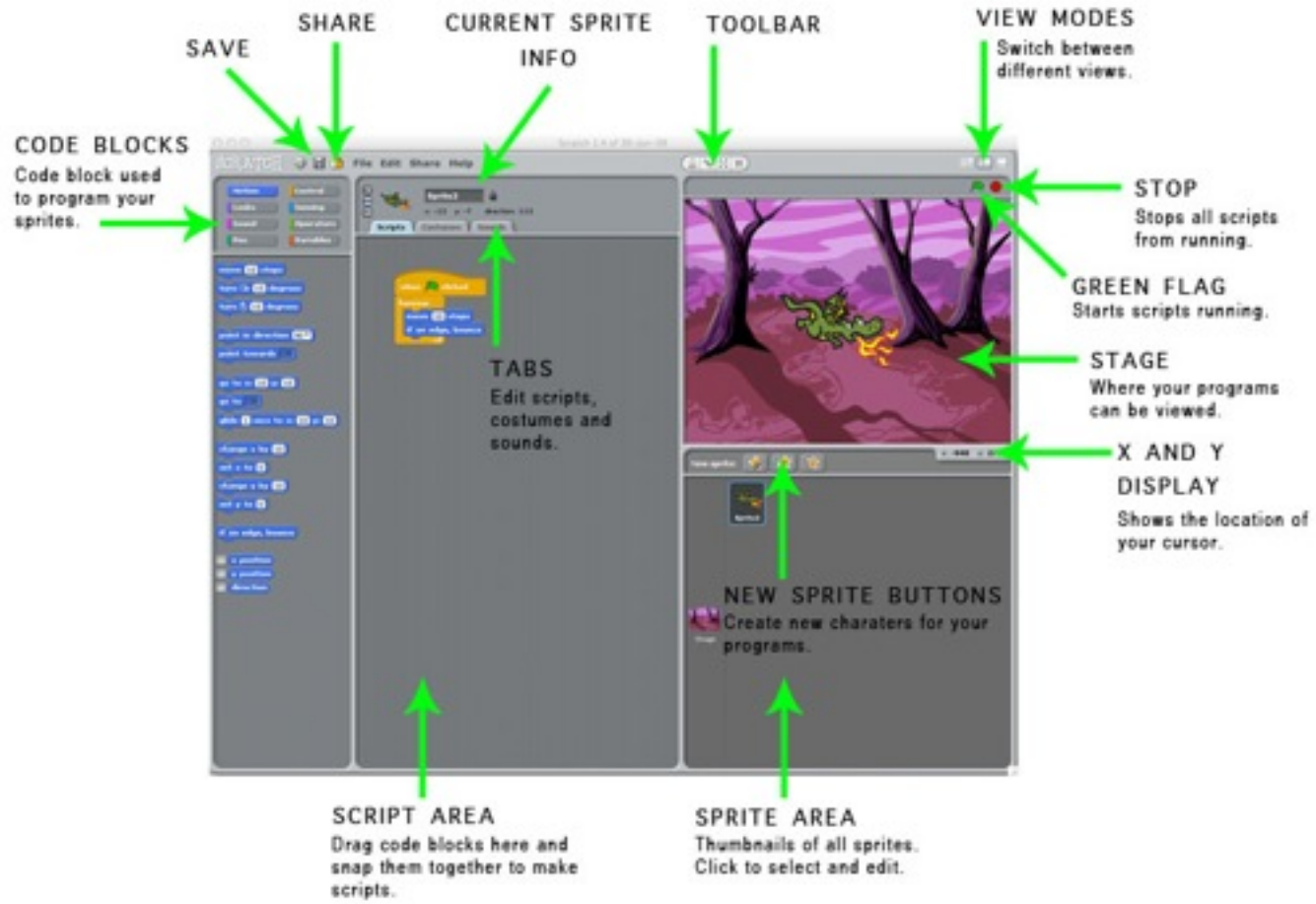
Knowsley
City Learning Centres

What is Scratch?

Scratch is a programming language and online community where you can create your own interactive stories, games and animations. It is freely available to download and install on either Mac OS X and Windows operating systems. A web based version will also run through your browser so it can be accessed without the need to install it on your computer.

Scratch Website: http://scratch.mit.edu
Download: http://scratch.mit.edu/scratch_1.4/
Web Version: http://scratch.mit.edu/projects/editor/?tip_bar=getStarted

The Scratch Interface



Sprites

What are sprites?

A sprite is a computer graphic which may be moved on-screen and manipulated as a single object independent of other objects.

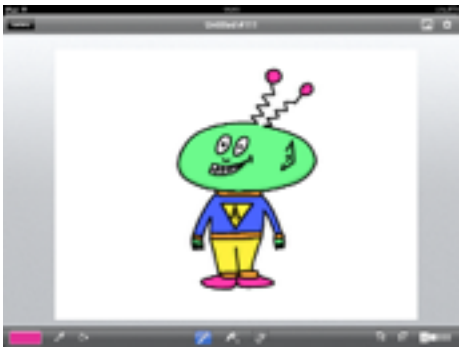
Sprites are objects that have scripts added to them to perform actions in a Scratch project. A Scratch project can have lots of sprites which can all perform different actions independently. A sprite can be created outside of Scratch and imported in or they can be created within Scratch using the costume editor. There is also the option to import sprites from the library that comes with Scratch.

Creating your own sprites without Scratch

It is possible to draw your own sprites using pen and paper or by using a art software application and then importing them into scratch to be used.

Hand drawn sprite

Hand drawn background for the stage.

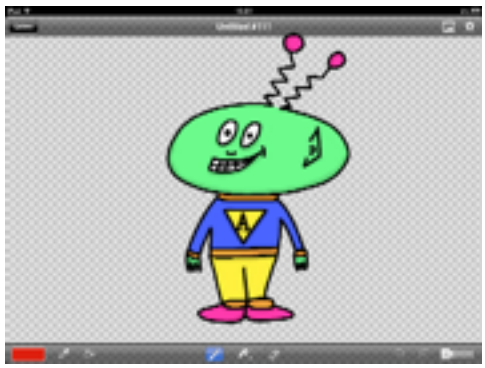


Sprite drawn using the Brushes App on the iPad.

Background drawn using the Brushes App on the iPad.

The images above show the two alternative ways that you can create sprites for use within Scratch. If you choose the hand drawn option you will then need to take a photograph of the drawn pictures using a digital camera and then use those photographs in Scratch. If

you choose to draw your sprites using an art package then you will need to save those drawing as JPGs and then import them into Scratch. The reason I have used Brushes on the iPad is because you can remove the white background of the sprite and it saves you having to remove that background later in Scratch which can be time consuming.



Once you have created your sprites you will need to import them into your Scratch by clicking the *New Sprite Button*, locate the sprites on your computer and click ok to import.

Video: Importing Sprites.mp4

If you chose to draw your sprite on paper and then take a digital photograph of it then you will need to edit your sprite using the paint editor within Scratch. To do this you need to select the *costume tab* and then the *edit* button.

Using the rubber icon in the paint editor you will then need to remove the white background.

Video: Editing Sprites.mp4

Creating your own graphics within Scratch

Using the paint editor in Scratch is the other way of creating your own sprites in Scratch.

Firstly you select the *Paint New Sprite* button to launch the paint editor.

Once the paint editor has launched you can then start to paint and draw your sprites and characters for your games.

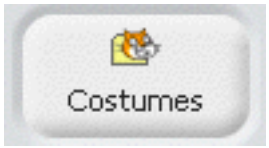
Click ok to save your sprite once you have finished

Video: Using The Scratch Paint Editor.mp4

Using sprites supplied with Scratch

The great thing about Scratch is that it comes with lots of sprites that are ready for you to use in your projects.

To use the sprites that come with Scratch you need to click the *Choose new sprite from file button.*



Once the new sprite window has loaded, click the *Costumes* icon down the left hand side.

Select a folder and search through to find a style of character you would like to use as your sprite. Once you have found the sprite you like click ok to import it.

Video: Importing Sprites From The Library.mp4

Activity 1

Using one of the previous methods ask the children to create their own sprite that will be the main character in their game and import it onto the stage in a new Scratch Project. Save the project so that it can be used in the following activities.

The Stage

Hand drawn stage

Stage drawn using Brushes on the iPad

Stage drawn using the paint editor within Scratch

Alternative stage using the library of images in Scratch

You can also import graphics to the stage which will act as the background image. It is possible for you to draw your own backgrounds on paper or draw them using an art software package like it is with the sprites or you can import background images from the library of images available in Scratch.

To change the background image of your stage you firstly need to click on the stage itself.

Next step is to select the backgrounds tab.

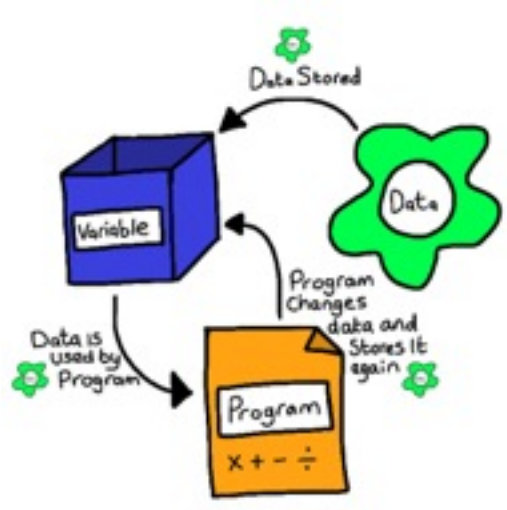
You will then have the choice of either painting your own background or using the import button to select one of the backgrounds you have created yourself or from the Scratch library.

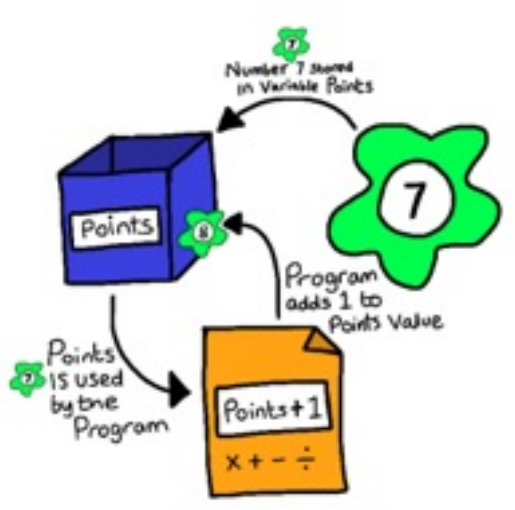
Video: Changing The Stage.mp4

Activity 2

Ask children to open their Scratch project from the previous activity and using one of the previous methods ask the children to create their background image for the character they created in activity 1 and save their project for the next activity.

Variables





What is a variable?

A variable may be thought of as a box within your program in which you can store data and information to be used by the program at different points. A variable may hold numbers, text, dates and lots of other types of information. For example a number can be used whenever needed and changed throughout the program. You may need to add to the number you have stored in a variable box or take away from it.

You can create your own variables in Scratch and it is always a good idea to give your variables a descriptive name that explains what the value stored in it represents. For example if you wanted to store the amount of points a player has got during the game you would create a variable called *Points* or *Score*.

Where might we use variables in a game?

We might use variables to store the number of points scored in a game by the player. You would have a variable called points or score which would be set to 0 and then you would add a value to that variable every time the play did a certain task.

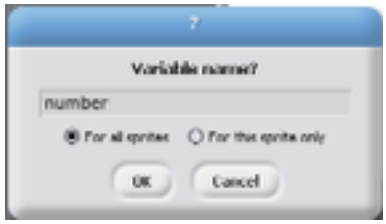
A variable will also be used to store the amount of lives that a player has within a game. This variable will be set at a certain value and then decreased each time the player loses a life.

Computer programs use lots of different variables and almost all computer programs will use at least one variable. Other examples of where variables might be used in a computer program are, players name, game time and game character speed.

Have a look at the different types of information about yourself that could be stored in a variable as part of a computer program.

Create a variable

To create a variable in Scratch you need to click on the **Orange Variables** tab and then click on the **Make a variable** button. This will open a new window where you can type the name of the of your variable. Then click **ok** to create your variable.



In the **Orange Variables** tab you now notice your variable now has its own code block. We need to remove the tick in the box next to it by clicking on it to remove it from the screen.

Video: Creating a variable.mp4

Changing a variable

Once you have created a variable you will notice more code blocks appear in the variables menu. One code block allows you to **set** your variable to a certain value and there is a code block that allows you to **change** the value of your variable.

Video: Changing a variable.mp4

Activity 3

Ask children to open their Scratch project from the previous activity, click on their sprite they have created and **make a variable** called *Number* for the sprite that they have drawn in activity 1. Once they have created the variable number ask them to build the code to set *Number* to 0 once the *Green Flag* has been clicked and save their project for the next activity.

Video: Activity 3.mp4

User Input

Why do we have user input?

A program or game means very little if there is no input need from the user. Inputs are ways in which the user or the player will interact with the game.

Examples of user input

The most common form of user input will come from the keyboard but you can also get inputs from the mouse or from a joypad on a games console.

We can get different types of user input, for example we might want the user to input their name, this would be a text input or we might want them to control the direction in which a sprite moves when a certain key is pressed.

Basic user input example

This example shows how a to ask a question to the user and wait for their input from the keyboard. If you start by using a **when green flag clicked** code block and attaching to that a **ask** *What's your name?* **and wait** code block under the **blue** sensing tab.

Next step is to connect to that the **say** *Hello* code block from the **purple** looks tab.

The final step would be to add the **answer** code block from the **blue** sensing tab to the inside of the **say** *Hello* code block.

Video: Basic User Input.mp4

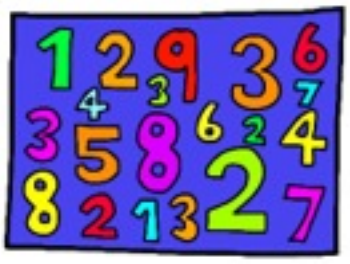
Activity 4

Ask children to open their Scratch project from the previous activity and add a **ask** *What's your name?* code block from the sensing tab. Then change it to ask *Guess which number I have picked?* and save their project for the next activity.

Video: Activity 4.mp4

Random Numbers

Why use random numbers?



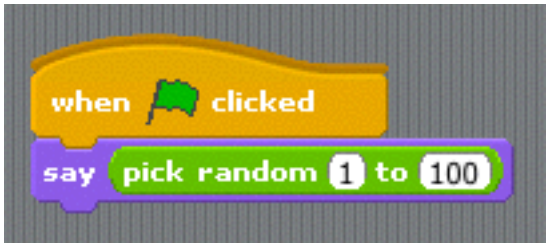
Random numbers are used to add a richer user experience for the user. By asking the computer to select a random number we cant predict what the computer is going to do. To use the random number code block you simply change the two given numbers and it will select a random number in between these, for example if you changed the numbers to 1 and 3 it would return a value of either 1, 2 or 3.

Random number example

This example uses the code block **pick random** *1* **to** *10* from the **green operators** tab to make a sprite say a random number from 1 to 100.

Start by adding a **when green flag clicked** code block and connecting to that a **say** *Hello* code block from the **purple** looks tab.

Then select the **pick random** *1* **to** *10* from the **green operators** tab and drop it inside where is says *Hello!*.



Change the value 10 to 100 so that it has a greater range of numbers to choose from when picking a random number. Keep pressing the green flag and see what happens.

Video: Random Numbers.mp4

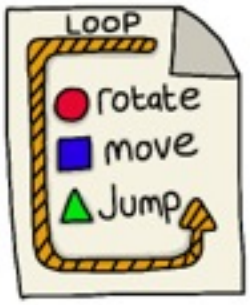
Activity 5

Ask children to open their Scratch project from the previous activity and add a **pick random** *1* **to** *10* code block from the operators tab to the **set** *Number* **to** code block in their project. Then change the values in the **random** code block *to 1 to 100* and save their project for the next activity.

Video: Activity 5.mp4

Loops

What is a loop?



A loop is a sequence of instructions that is continually repeated until a certain condition is reached. A loop is a important programming idea and it is commonly used in writing programs. A loop code block will repeat a group of other code blocks again and again until it is told to stop. Inside a loop the individual actions will still happen in the same sequence they appear. When the last code block finishes, the loop will start all over again with the first one.

How loops are used?

There are a few different types of loops available in Scratch:

Forever loops are used to repeat blocks of code continually throughout the program until the user stops the program.

Repeat loops are used to repeat a block of code for a certain number of times and then it will stop itself.

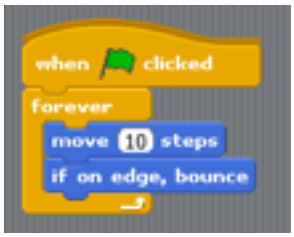
Forever If loops are used to repeat a code block within it if a condition is being met.

Repeat Until loops are used to continually execute the code block within it ***until*** something happens which will stop it from looping.

Loops example

In this loops example will be using a **forever loop** to make a character move backwards and forwards across the screen.

Start by adding a **when green flag clicked** code block and connecting to that a **forever** *loop* code block from the **orange** control tab.



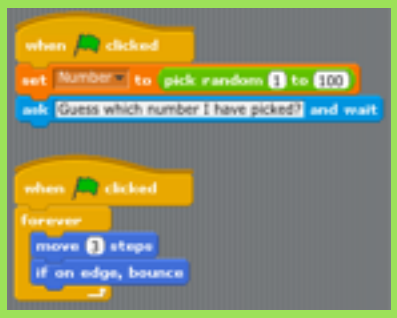
We can now add some code blocks inside our forever code block to create our loop. We need to connect two code blocks inside the forever code block and they are the **move** *10* **steps** code block and the **if on edge, bounce** code block which can be found in the **blue motion** tab.

The next step would be to click on the green flag and see what happens to your sprite.

Your sprite should begin to move across the screen and then bounce back the other way. You will notice when your sprite bounces back, it appears upside down. To turn your sprite the right way up at all time you will need to click on the *only face left right button.*

Video: Loops.mp4

Activity 6

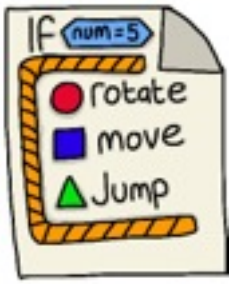


Ask children to open their Scratch project from the previous activity and add a loop to their project that will make their sprite move across the screen and then save their project for the next activity.

Video: Activity 6.mp4

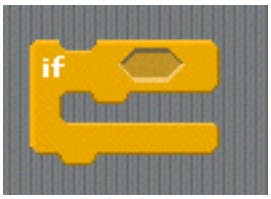
Conditions & Operators

What are conditional statements?



Conditional statements execute sequences of code blocks depending on whether or not a certain condition is True.

This is a **IF** code block which is used as a conditional statement within Scratch. The diamond shape next to the **if** is where the condition will go. The code blocks and actions that will happen if the condition becomes **TRUE** go inside the code block itself.



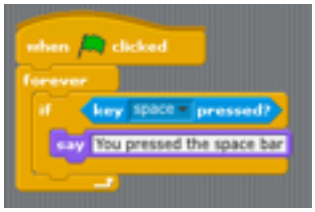
This is another type of conditional statement used in Scratch. It is a **repeat** loop that will loop for the life of the program until some condition becomes **TRUE** then it will stop looping.

Conditional statement example?

This example will show how a basic conditional statement works in Scratch. Start by adding a **when green flag clicked** code block and connecting to that a **forever** *loop* code block from the **orange** control tab.

Inside the forever loop add your **conditional statement** in the form of the **If** code block.

Inside the diamond (condition) part of the **If** code block add a **Key** *space* **pressed** code block form the **blue sensing** tab.



Then finally add a **say** *Hello!* code block from the **purple looks** tab inside the **If** code block and chafe it to say *You pressed the space bar*. Click the green flag and test your code.

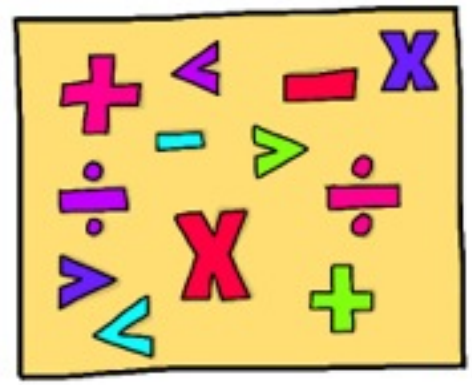
Video: Conditional Statements.mp4

Activity 7

Ask children to open their Scratch project from the previous activity and connect a **repeat until** code block to the **ask code block**. Then connect two **If** code blocks inside the **repeat until** loop. Save their projects for the following activity.

Video: Activity 7.mp4

Operators

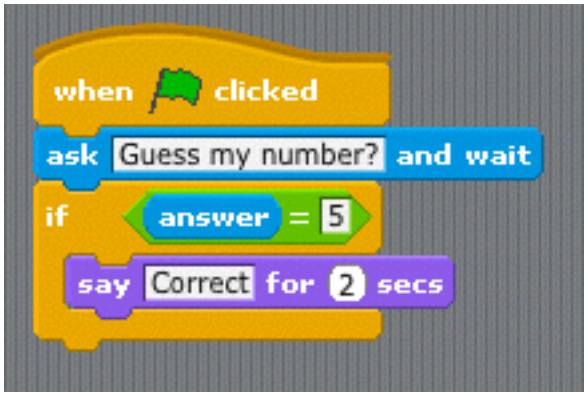


Operators are used to test the conditional statements. They will check if a value is equal to another value or if its greater than another value. They can also be used to add, multiply, takeaway or divide values.

For example here we can check if a number we have chosen at random is equal to the number given to us by the program user.

Operators and Conditional Statements will work together in a program.

Operators example



In this example we are going to look at how to use a basic **equals** operator.

Start by adding a **when green flag clicked** code block and connecting to that a **ask** *What's your name* code block from the **blue** sensing tab. Change the question inside to say *Guess my number?*

Then connect an **If** statement code block to that and place a **say** *Hello!* **for** *2* **secs** code block inside it, change it to say **say** *Correct.*

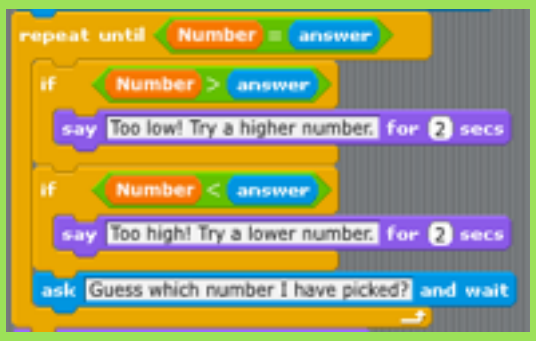
Next we will user are first **operator** from the green tab. Drag the diamond with the equals sign in and drop it in the **If** statement.

We then need to add the values to the operator for the **If** to have a condition to test against. If you drag the **answer** code block from the blue sensing tab and drop it into the first box of the green **equals** operator and in the second box a number of your choice. The answer in this example will be the response the user gives us to the question *Guess my number?* This is then tested against the other number we have provided (5 in our example). So **If** the **answer** (the number given by the user) is **=** the number the computer program has provided (**5**) **then** say **Correct** (the user has guessed correctly).

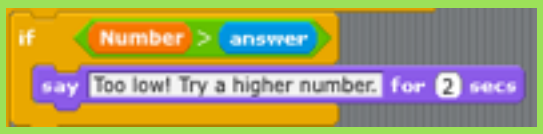
Video: Operators.mp4

Activity 8

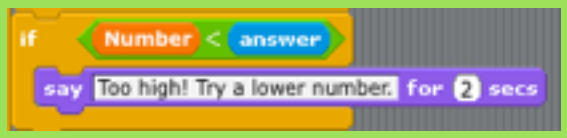
Ask children to open their Scratch project from the previous activity.



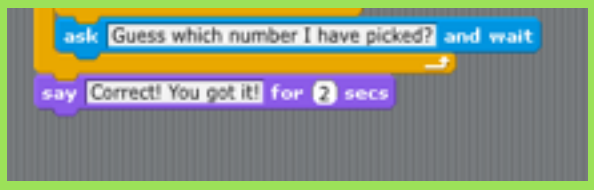
First step is to add to the **repeat until** loop a **equals to** operator. In the first box drag the **Number** variable we created in previous activity and drop that in. In the second box drag the **answer** code block form the sensing tab and drop that in. This section of code is going to **repeat until** the **Number** (our random number) is **equal to** the **answer** (the response given to us by the game player).



Next step is to add a **greater than** operator to our first **If** statement and again add **Number** to the first box and **answer** to the second. Here we are checking **If** the number we have chosen at **random** is greater than the **answer** given by the game player. If this is true we need to tell the game player to try again with a higher number. Add a **say** *Hello!* **for** *2* **secs** code block inside the **If** statement but change the text inside to say *Too low! Try a higher number.*



For the second **If** statement add a **less than** operator and again add the **Number** variable to the first box and **answer** to the second box. In this **If** statement we are checking to see if the **random** number is less than the **answer** given by the game player. If this is true then we need to tell the game player to try again but chose a lower number. Add a **say** *Hello!* **for** *2* **secs** code block inside the **If** statement but change the text inside to say *Too high! Try a lower number.*



To finish the game off add another **ask and wait** from the sensing tab under the second **If** statement and change the question inside to *Guess which number I have picked?*

Add a **say** *Hello!* **for** *2* **secs** code block to the bottom of the **repeat until** loop and change the text to say *Correct! You got it!*

Video: Activity 8a.mp4 **Video:** Activity 8b.mp4 **Video:** Activity 8c.mp4

Video: Activity 8d.mp4