

PROJECT BOOKLET FOR CODING CURRICULUM

GRADE VII

Volume 1.0



TABLE OF CONTENTS

Distributing Birthday Sweets.....	1
Arranging The Books	3
Algorithm For A Perfect Square.....	5
Sorting The List	7
Variable Initialization In Programming	11
Drawing A Rectangle Using Minecraft	14
Better Way To Create A Square.....	20
Fun Activity: Chase the apple.....	24
Adding two integers	34
Finding the square of a number	37
Can a function return value	38
Calculating Area Of A Circle.....	46
Create An Array And Calculate Its Length.....	55
Building A Zoo.....	59
Examples of Simple functions in Arcade.....	67
Examples of arrays in Arcade	71
Building a calculator	73
References	79



DISTRIBUTING BIRTHDAY SWEETS

Chapter: Sequencing with Block Coding

Problem Statement:

Consider there is a birthday of one of the students in your class today, named Arun. As a normal birthday ritual, he is planning to distribute the birthday sweets amongst the students. Can you draw a flowchart of the activities that he performs while distributing the birthday sweets within the class?

Learning Outcomes:

At the end of this exercise, you will learn:

- Learn the concept of sequencing and how sequencing is applied in the activities that we perform on a daily basis.

Solution:

It is Arun's birthday today. The entire class wishes birthday to Arun. Later, Arun takes out the sweets from his bag and starts distributing it to the students of the class.



Take a look at flowchart in *Fig 1.0* understand how Arun uses concept of iteration while distributing birthday sweets with the class.

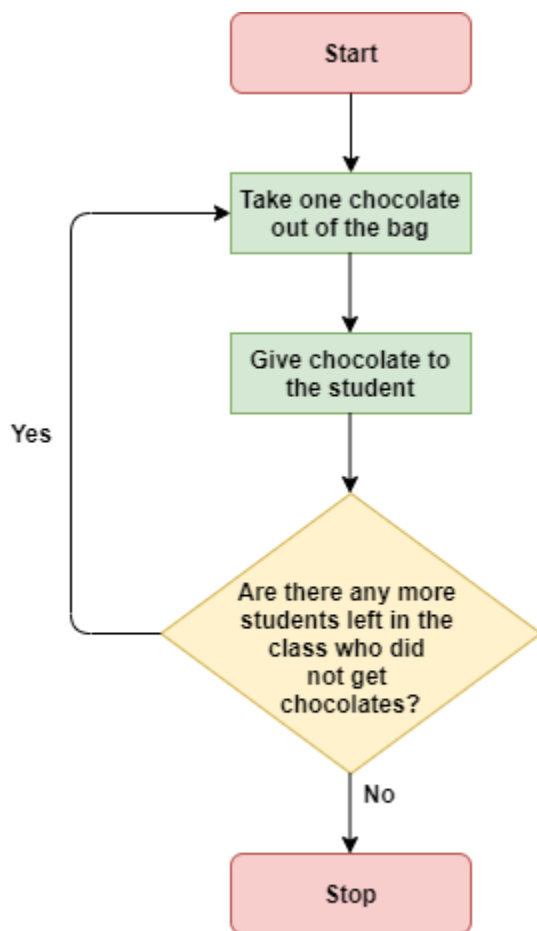


Fig 1.0 Distributing Birthday Sweets

If you notice, there is a pattern which Arun follows while distributing sweets. He takes the chocolates out from his bag, gives one chocolate to a student, moves to the next student and repeats the same steps again till the sweets are distributed within all the students.

This is an example of an iterative process. Repetition of a set of steps with a defined end – in this case the repetition ended when all the students in the class were given chocolates.



ARRANGING THE BOOKS

Chapter: Fun with Functions

Problem Statement: Suppose you have a lot of books with you. However, all these books are mixed up. Draw a flowchart to explain the activities that you need to perform in order to sort these mixed up books from shortest to tallest.

Learning Outcomes:

At the end of this exercise, you will learn:

- Learn the concept of sorting how sorting is applied in the activities that we perform of daily basis.

Solution:

In this activity, you will learn about the concept of “Sorting” in programming.

For the given problem statement, if you want to arrange these books, you can do it in many ways.

1. You can arrange them from tallest height to smallest.
2. You can arrange them alphabetically.
3. You can arrange them by the frequency in which you use them.

The ways of arranging that we just listed down, is called as “Sorting” in programming.

In Sorting, we take a jumbled set of objects and arrange them in some kind of order.

For Sorting, we need to know how to compare two items. So, we will ask questions like:

Which book is taller/shorter?

Let us understand sorting of books from shortest to tallest with help of flowchart in *Fig 2.0*.

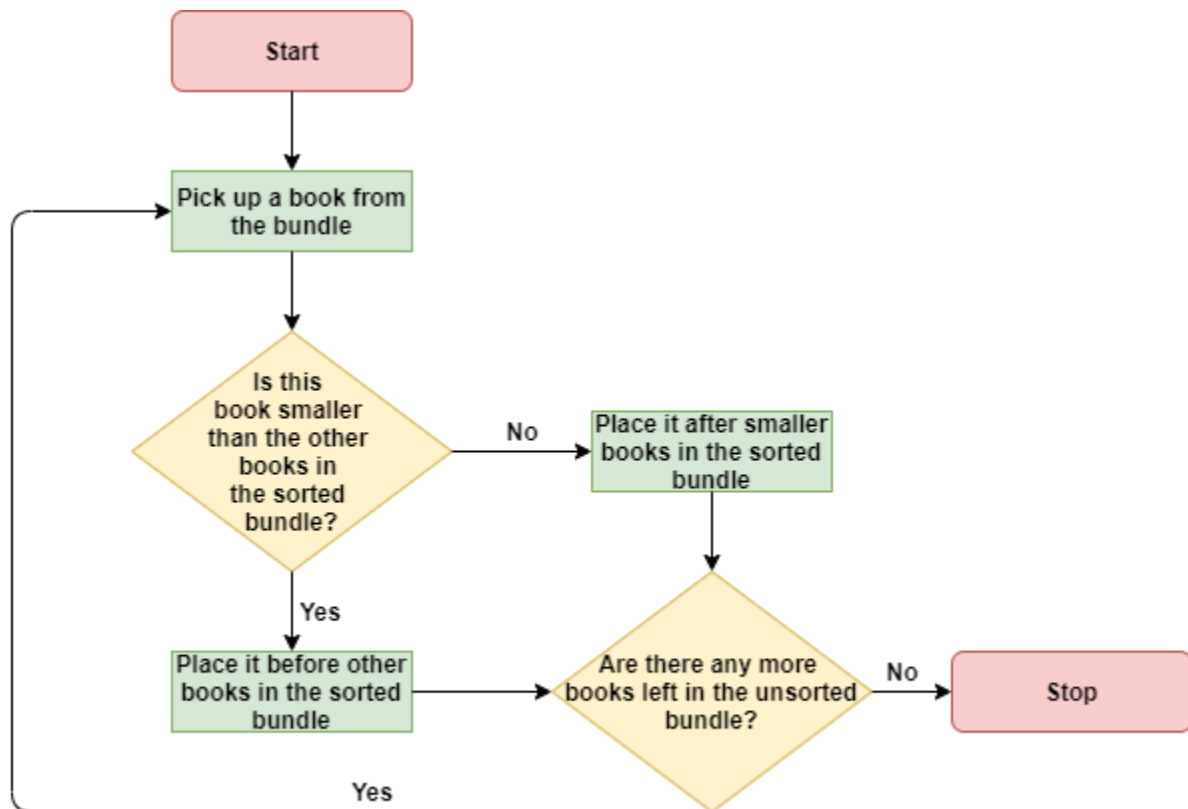


Fig 2.0 Arranging Books

There are many clever sorting algorithms that computers use. They help you sort different kinds of items very quickly.

Sorting is helpful because:

1. Sorting a collection helps you answer questions like “which is the tallest/smallest/fastest item in this collection?”

Arranging things in order can make other algorithms work better.



ALGORITHM FOR A PERFECT SQUARE

Chapter: Understanding Arrays and Collections

Problem Statement: You have learnt the concept of perfect square in mathematics. Similar to how you find out if a number is a perfect square in mathematics, can you write an algorithm to find out if the given number is a perfect square that can be applied in programming?

Learning Outcomes:

At the end of this exercise, you will learn:

- Learn to write an algorithm for finding a perfect square.
- Learn how to use this algorithm as a reference to write algorithms for other mathematical concepts that they have learnt so far.

Solution:

In this activity, we will learn to write an algorithm for finding if a number is a perfect square.

To start with let us take a look at following questions:

- How quickly can you determine whether a number is a perfect square?
- And, if a number is a perfect square, how can you find out its square root?
- Can you write the steps for finding the same for any given large number?
- Let us try with a few examples and identify the repeatable steps in the process.

Example 1: Is 36 a perfect square?

We can use the below algorithm to solve this problem:

In the below algorithm, “n” is the input number (36 in this case)

Output is true if the number is a perfect square, and false if it is not a perfect square.



```
begin
    for num :=1, num <= n, increase num
    by 1:
        if n is divisible by num, and n / num
        == num, then
            return true
    done
    return false
```

Can you think of
algorithm?

ways to improve the

Have a look at below optimized algorithm.

1. Say number is n
2. Start a loop from 1 to $n/2$
3. During iteration, for every integer 'i', calculate $x = i*i$
4. Now with this 'x' there are 3 possibilities:
 - a. If $x == n$ then n is a perfect square, return true.
 - b. If $x > n$ then x has crossed the n, is not perfect square. Return false
 - c. If above step a or b are not true, then continue.

As you must have noticed, the number of times we multiply, and compare is reduced significantly in the above algorithm. This is how we can use optimized algorithm to find the square of numbers.



SORTING THE LIST

Chapter: Hello World with Code

Problem Statement: You have learnt the concept of sorting. There are different techniques using which you can perform sorting. Consider the **Array** = {3, 2, 11, 7}. Can you write an algorithm to sort this array in ascending order using selection sort technique?

Learning Outcomes:

At the end of this exercise, you will learn:

- Learn how to write algorithm for selection sort.
- Learn how different sorting techniques affects how the code runs.

Solution:

We will learn the concept of Selection Sort in this activity.

Suppose we have a list of unsorted numbers. Over here we want to sort the list in a way that smallest number is on top of the list and largest number is at the bottom.

We start with finding the smallest number from the list. Once we find this number, we want to put it at the top of the list. However, if you put this number on the top of the list, where should the number who currently is on the top of the list be shifted to? You do not have any additional space here.

Hence, once you spot the smallest number from the list, you swap it with the number which was present at the top position in the unsorted list.

Now, the first row from our list is sorted. We will sort the second row now. We will repeat the similar technique here. We will find the smallest number from the remaining unsorted list and swap it with the number in second row.

We will repeat this process till we reach the last row from the list. When you are done with the last element from the list you can look back and check that all the numbers from the list are now sorted and our list is now ordered in ascending order.

This method of sorting in programming is called as “Selection Sort”.

The algorithm for Selection Sort is as stated below:

- Step 1:** Set **MIN** to location 0

Step 2: Search the minimum element in the list

Step 3: Swap with value at location **MIN**

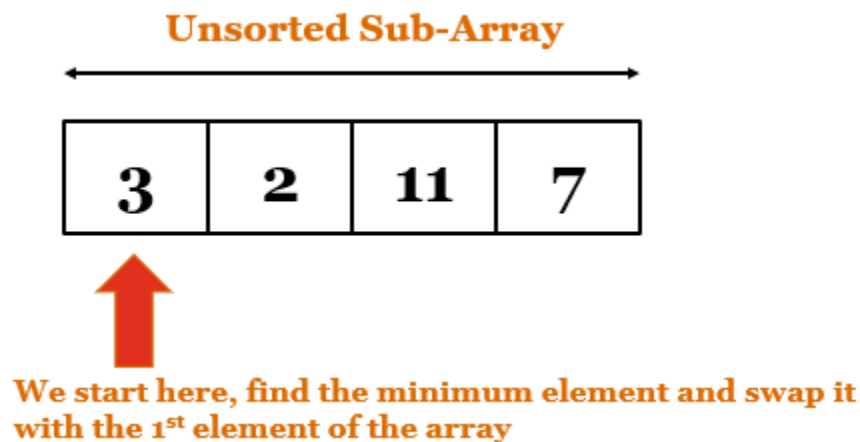
Step 4: Increment **MIN** to point to next element

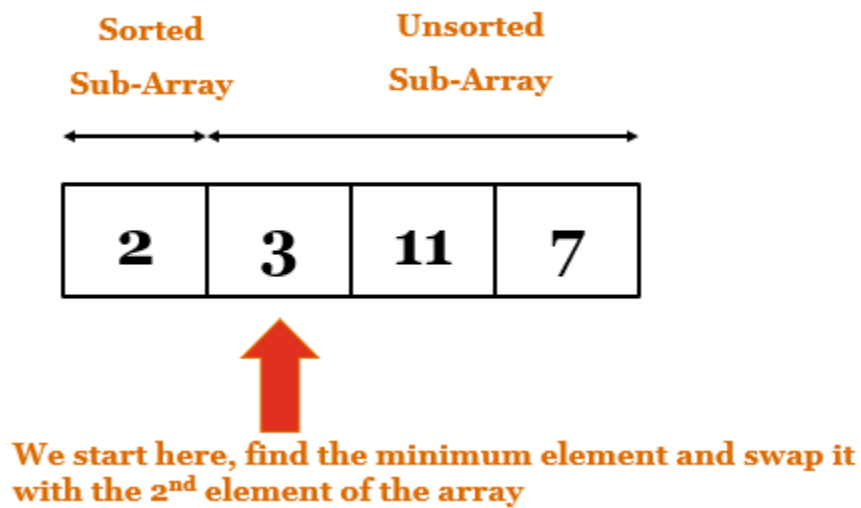
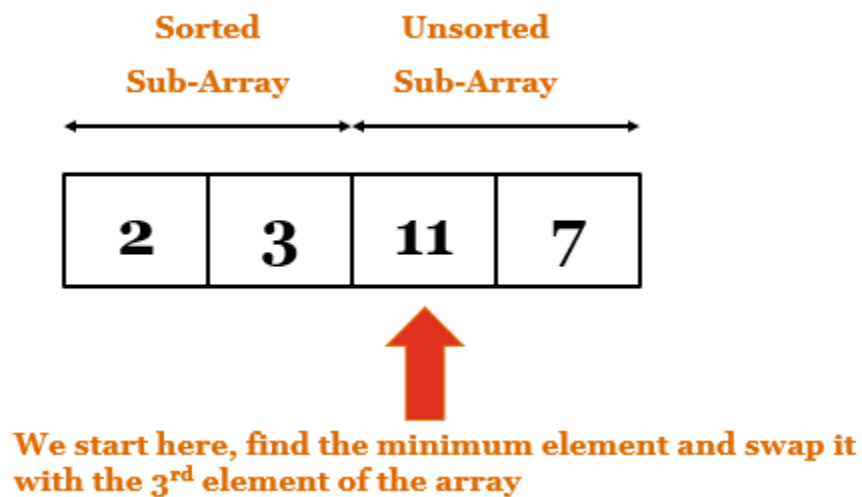
Step 5: Repeat until list is sorted

Let us now sort the below array in ascending order using selection sort algorithm:

Array = {3, 2, 11, 7}

Step 1: For i = 0



Step 2: For i = 1**Step 3: For i = 2****Step 4: For i = 3**

The loop gets terminated as “I” becomes 2.

VARIABLE INITIALIZATION IN PROGRAMMING

Chapter: Variables in Real Life

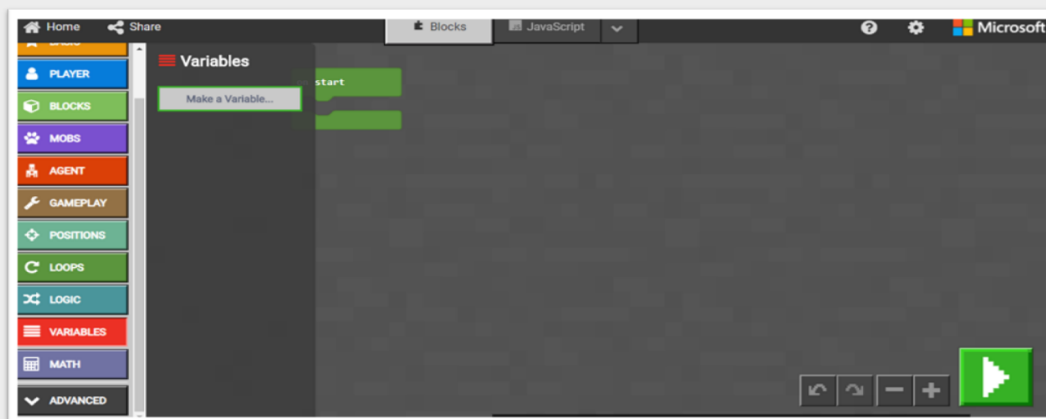
Problem Statement: You have learnt about variables in programming and its usage. However, in order to use the variables, it is important to initialize them first. Create a project in Minecraft to demonstrate variable initialization in programming.

Learning Outcomes:

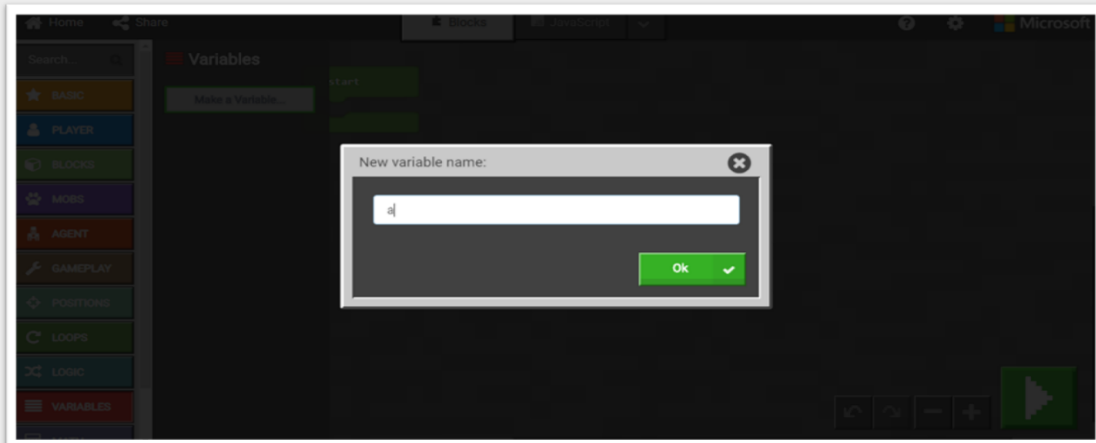
At the end of this exercise, you will learn:

- Learn to create variables that represent different types of data and manipulate their values.
- Learn how to create clearly named variables that represent different data types and perform operations on their values.

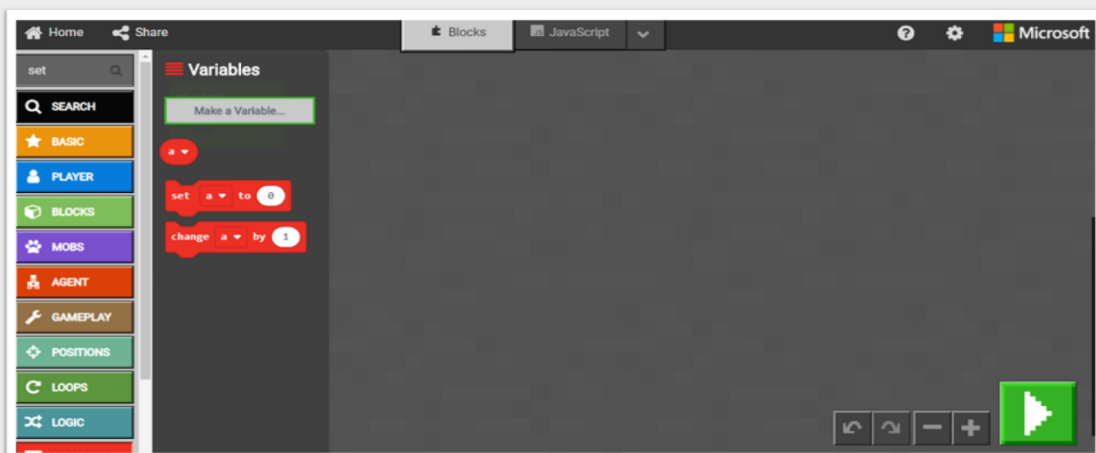
Solution: Create a new a project in Minecraft.



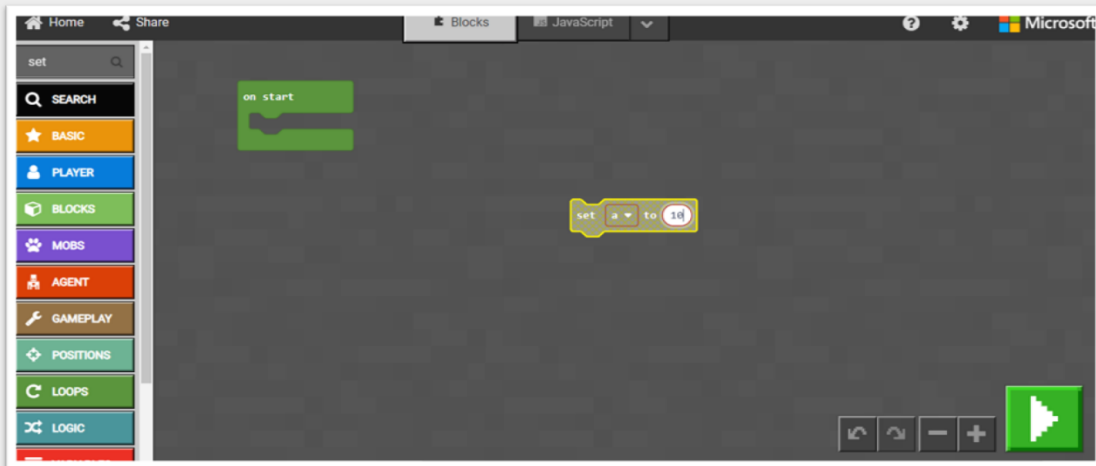
Step 1: Open Minecraft Editor, click on the “**Variables**” link from the toolbox and the click on “**Make a Variable**”



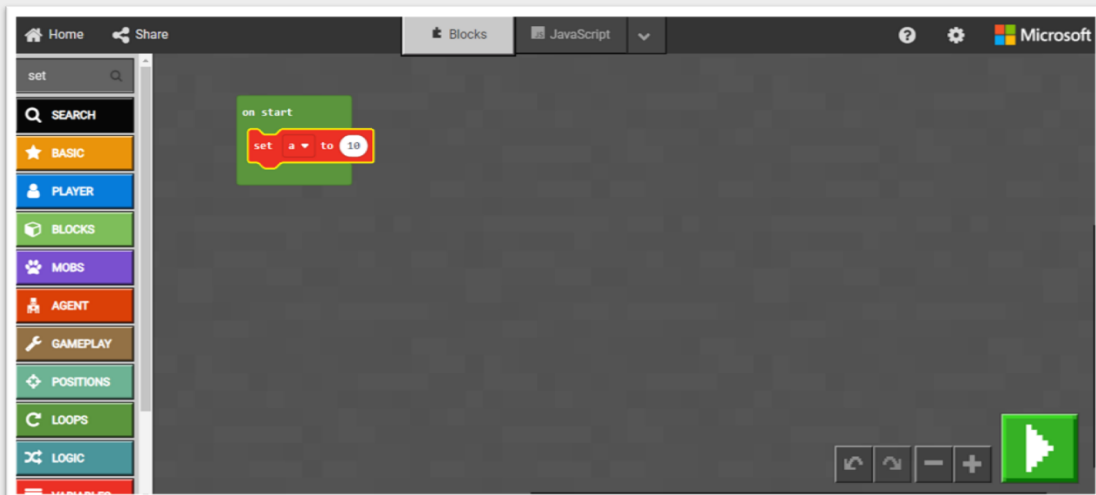
Step 2: Write “a” in the “**New Variable Name**” text box that appears on screen and click on green “**Ok**” button on the bottom of the page.



Step 3: Now a variable is created. This variable does not hold any value yet. So next step is to assign a value to this variable, i.e initializing the variable. To do this, click on “Variables” link again in the Toolbox and then drag and drop “set a to” block in the play area



Step 4: In the “**set**” block, assign a value to variable “a” as shown in the image.



Step 5: Drag and drop set block into “On start” block

We have now completed this exercise and learnt how can create and initialize variables in programming

DRAWING A RECTANGLE USING MINECRAFT

This activity will help you understand about sequencing with the help of Minecraft.

What would be the steps to draw a rectangle of length 5 cm and height 3 cm on your screen?

First, draw a line 5 cm in length

Then turn the cursor right by 90 degrees

Then draw a second line 3 cm in length

Then turn right by 90 degrees

Then draw a third line 5 cm in length

Then turn right by 90 degrees

Then draw a fourth line 3 cm in length

Chapter: Sequencing with Block Coding

Problem Statement: Suppose that you are sitting in a mathematical class and the teacher asks you to draw a rectangle. You know how to draw in using pen and paper. Can you think how you can draw a rectangle of length 5 cm and height 3 cm on screen?

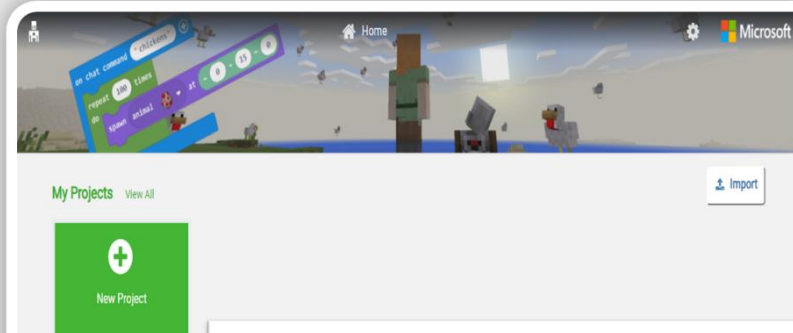
Learning Outcomes:

At the end of this exercise, you will learn:

- Learn how the sequence of the code affects how the code runs.
- Learn how to write algorithm for drawing various geometrical shapes using sequencing.

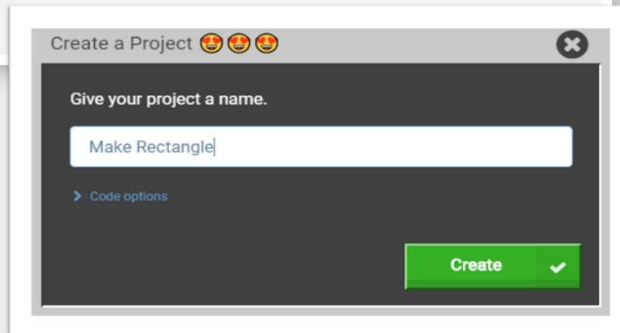
Solution: Let us now see how we can make a rectangle in Minecraft using the above steps. You should try this exercise on Minecraft using the MakeCode editor for Minecraft which can be found here <https://minecraft.makecode.com/>

First, create a new project as shown below.



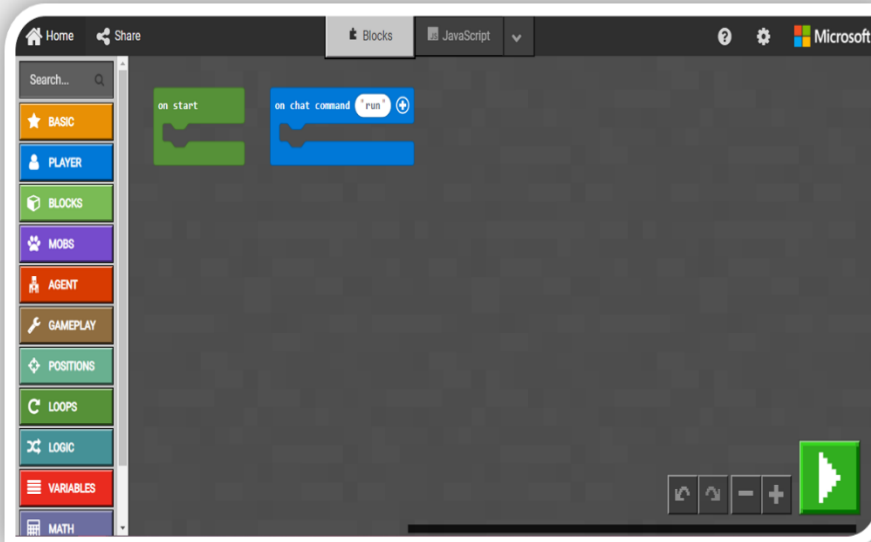
Creating New Project

You can create a new project by clicking on green box labeled as 'New Project'. A dialog box will appear prompting you to give a project name.



Giving Your Project A Name

You need to type down a name in the text and click on 'Create' button



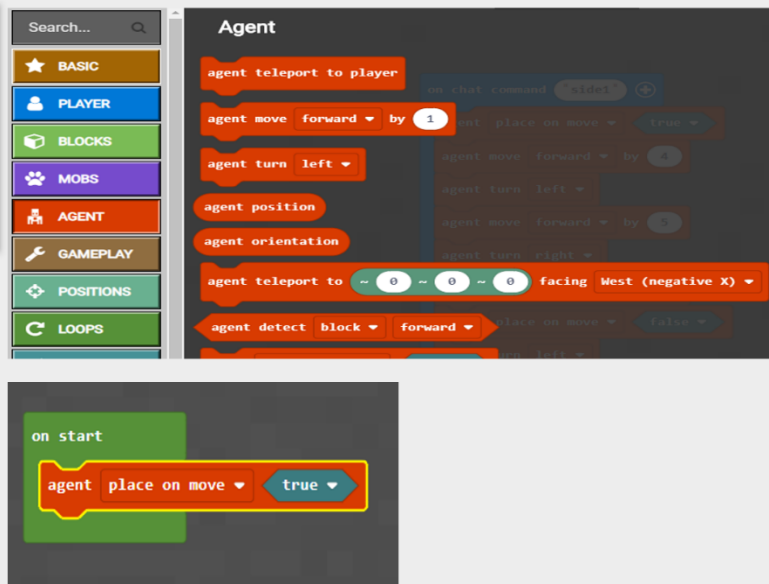
Minecraft Code Editor

Once you create your project, you should see the editor like this.

Now follow the following steps

Step 1

From the Agent section, select the “place on move” block as shown below. Set the value to True and add it to the Start block.

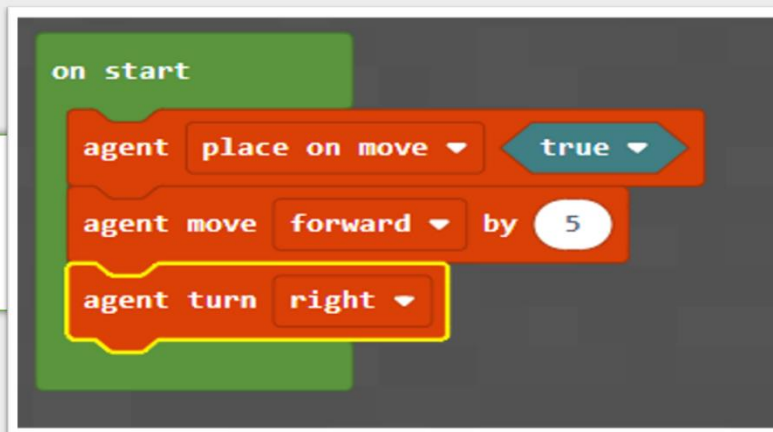


Step 2

From the Agent section, select the “move forward” block, add it to the Start block and set the value to 5 as shown.

Step 3

From the Agent section, select the “turn” block, add it to the Start block and set the value to right



on start



Step 4

From the Agent section, select the “move forward” block, add it to the Start block and set the value to 3 as shown

on start

```
agent place on move ▾ true ▾  
agent move forward ▾ by 5  
agent turn right ▾  
agent move forward ▾ by 3  
agent turn right ▾
```

Step 5

From the Agent section, select the “turn” block, add it to the Start block and set the value to right as shown

Step 6

You need to repeat Step 2. Placing a block which moves the agent forward by 5

on start

```
agent place on move ▾ true ▾  
agent move forward ▾ by 5  
agent turn right ▾  
agent move forward ▾ by 3  
agent turn right ▾  
agent move forward ▾ by 5
```




Step 7

You need to repeat Step 3 placing a block which turns the agent to right

Step 8

You need to repeat Step 4 placing a block which moves the agents forward by 3



If you complete the steps above, you should be able to create a rectangle.

BETTER WAY TO CREATE A SQUARE

With an example, let us now understand sequencing with loops and conditionals.

Chapter: Sequencing with Block Coding

Problem Statement: Suppose you are told to draw a square of side 5cm. You can either do it using pen and paper or you can draw it on screen using algorithm for programming. Now that you have learnt the concept of sequencing with loops and conditionals, can you think of steps to draw a square of side 5 cm on your screen?

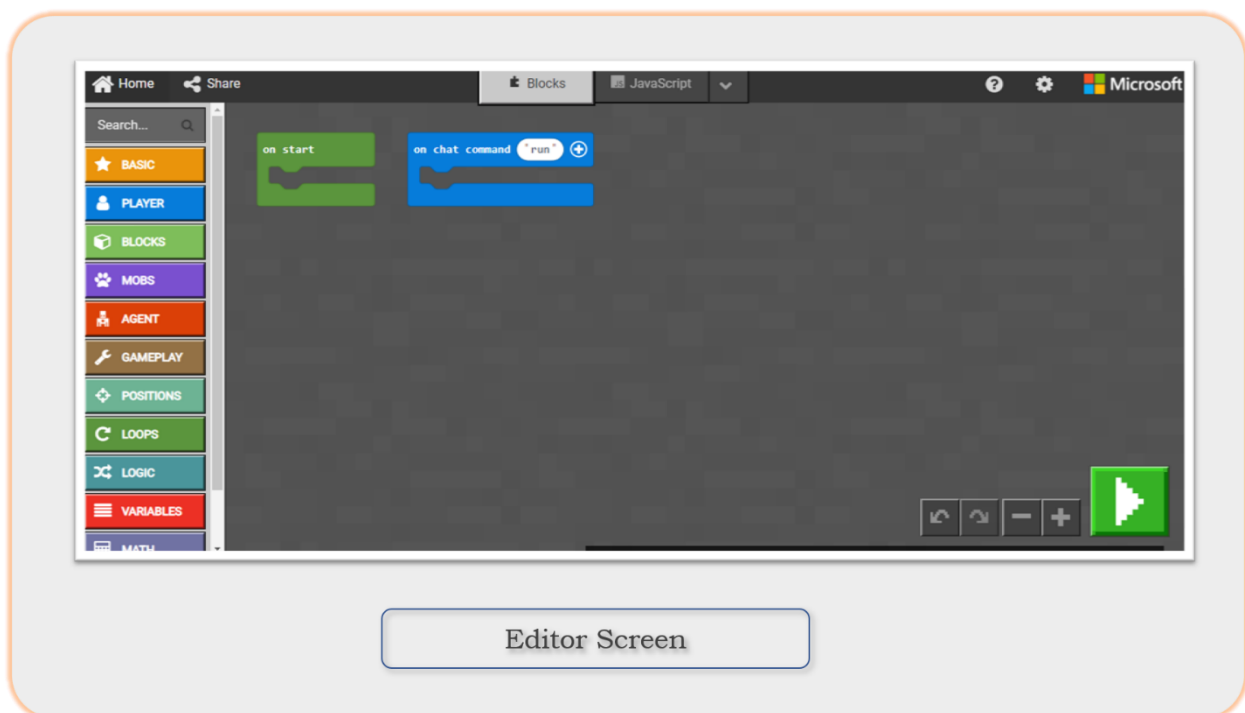
Learning Outcomes:

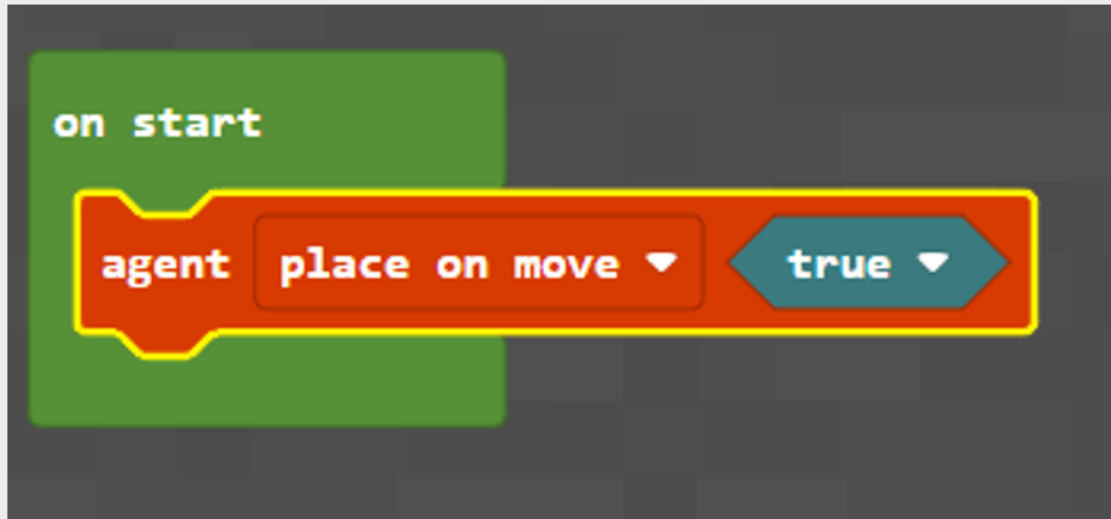
At the end of this exercise, you will learn:

- Learn to use loops and conditionals to allow the program to perform repeated tasks.
- Learn the importance of using “do while” with conditional loops.
- Learn to use conditional loops to complete challenges.

Solution: Do you notice a pattern getting repeated in the steps? Let us see how we can use a loop to reduce the steps by using the Minecraft platform.

First, create a new project called “Make Square”. Once you create your project, you should see the editor below

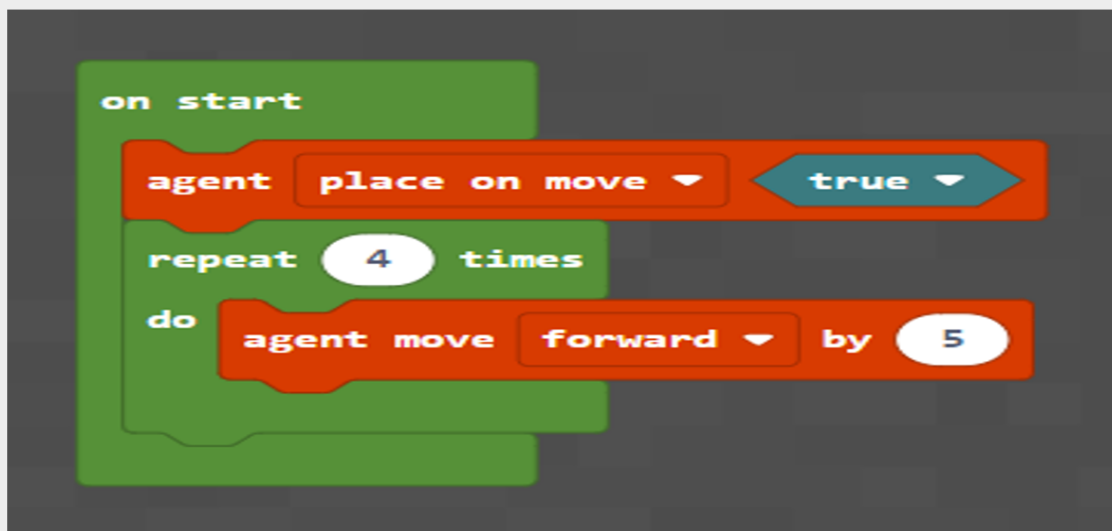




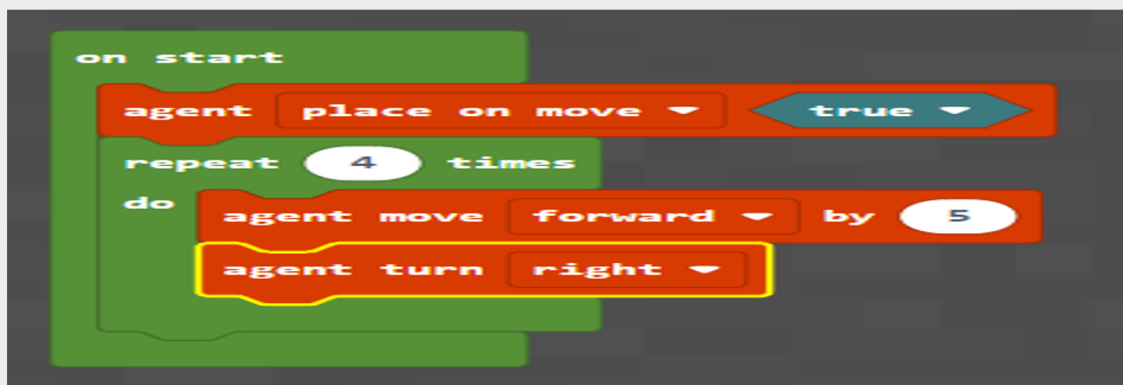
Step 1: From the Agent section, select the “**place on move**” block as shown below. Set the value to True and add it to the **start** block.



Step 2: From the loops section, select the **repeat** block and add it to the **Start** block as shown below.



Step 3: From the Agent section, select the “**move forward**” block, add it to the repeat block and set the value to 5 as shown below.



Step 4: From the agent section, select the “**turn**” block, add it to the repeat block and set the value to right as shown



Step 5: Add agent teleport, agent set active, and agent set block or item commands before agent place on move block as shown in the image and click on **“Play”** button to see the final output



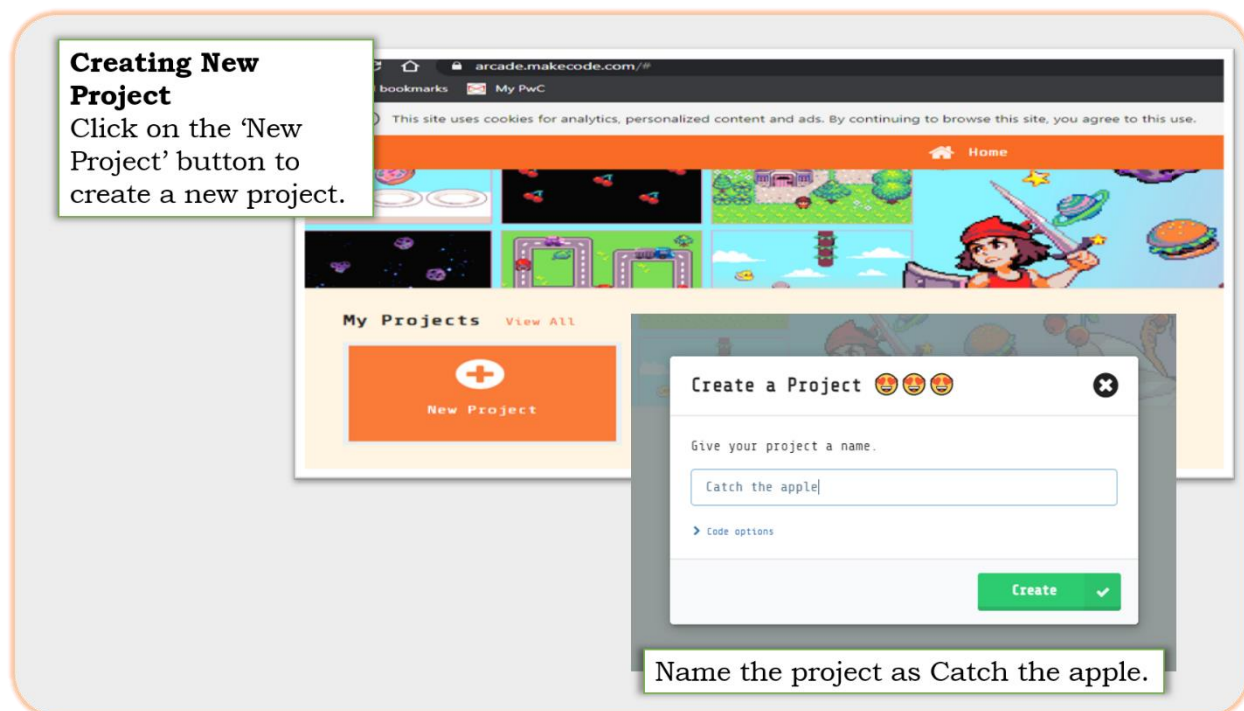
Final output for activity better way to create a square

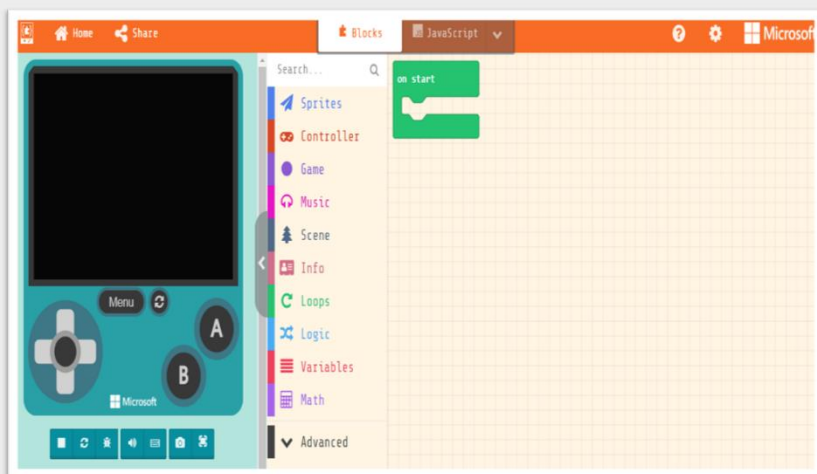
FUN ACTIVITY: CHASE THE APPLE

Chapter: Sequencing with block coding

Problem Statement: In this activity, we will create a game with 2 sprites, a Player sprite and an Apple sprite. The objective of this game is to chase and catch the wandering apple and collect as many points as possible before the time runs out. Every time the apple is caught, points are added, and the timer is restored.

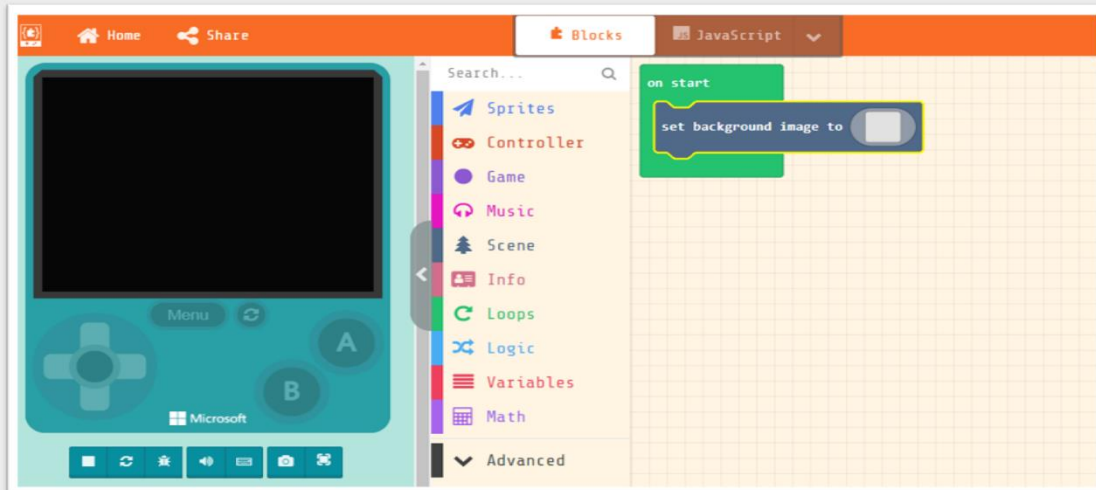
Solution: First, create a new project in Minecraft as shown below.



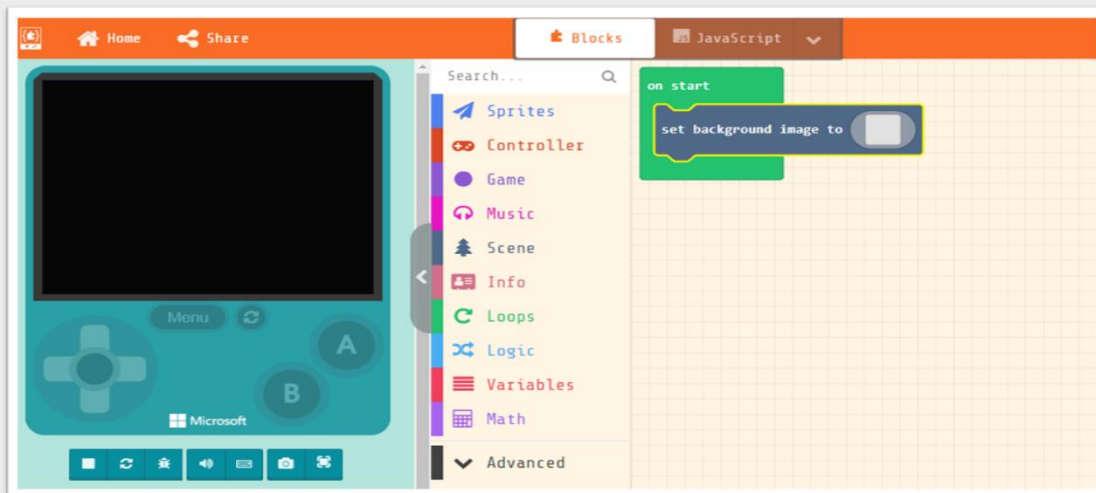


Once the project is created, you should see the editor.

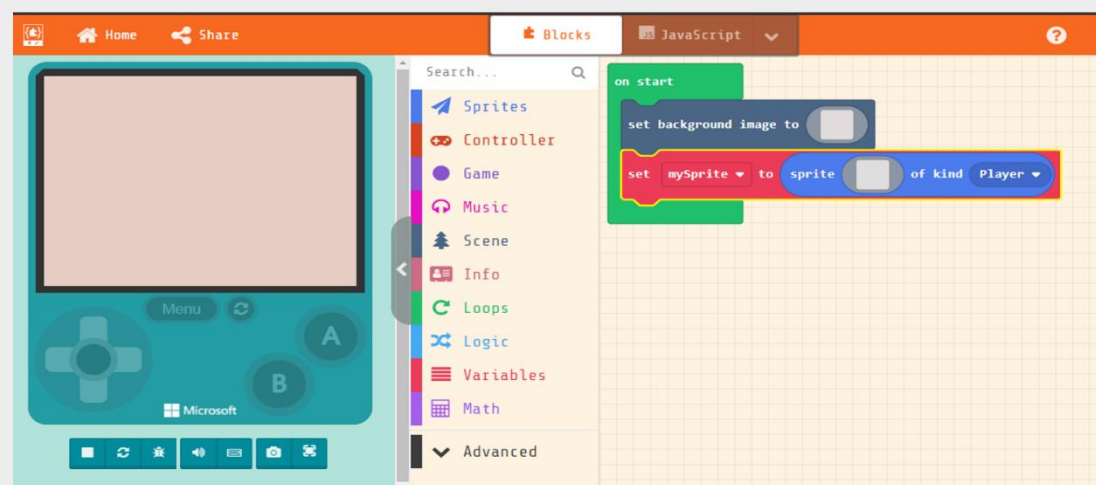
Now follow the steps mentioned below to create the game.



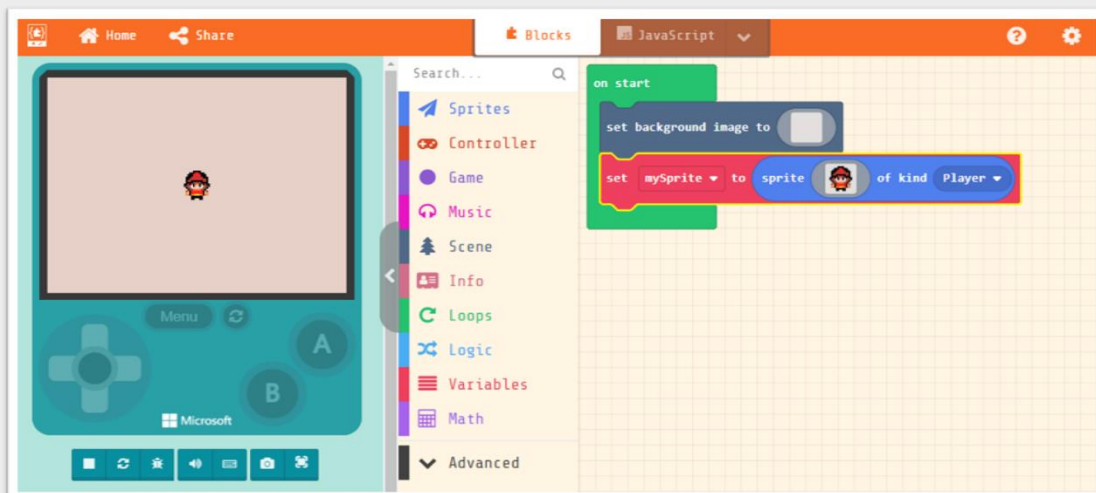
Step1: Open the Scene toolbox drawer and drag the “set background image” block into the “on start” block on your workspace.



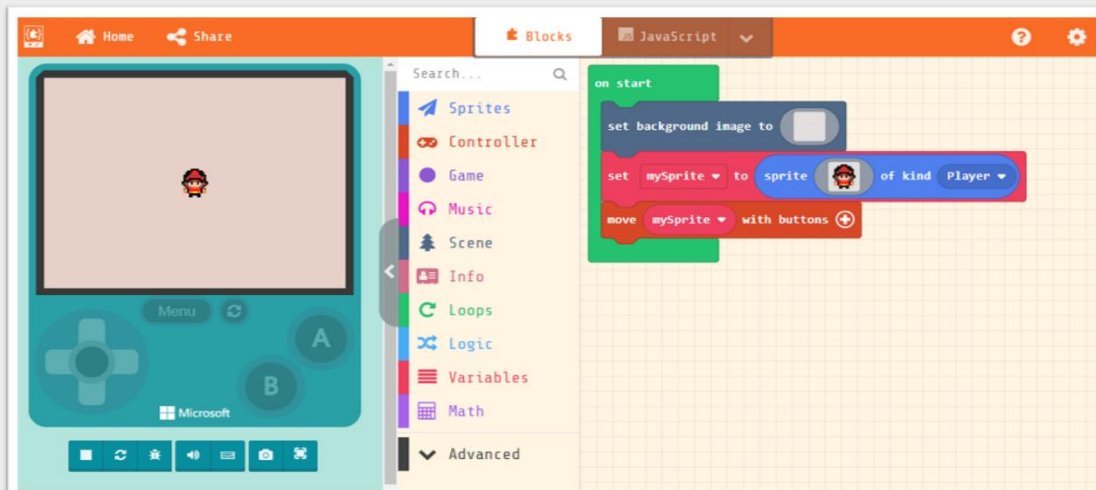
Step 2: In the “set background image” block, click on the gray square to open the image editor, select a color to fill in the background color.



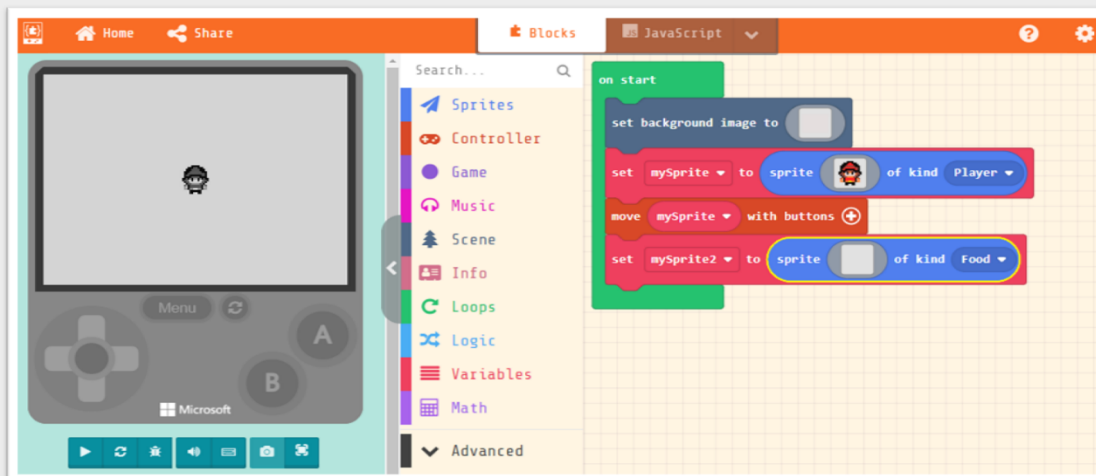
Step 3: Open the Sprites toolbox drawer and drag the first block you see, “set mySprite” into the “on start” block on your Workspace This will create a new Player character for the game.



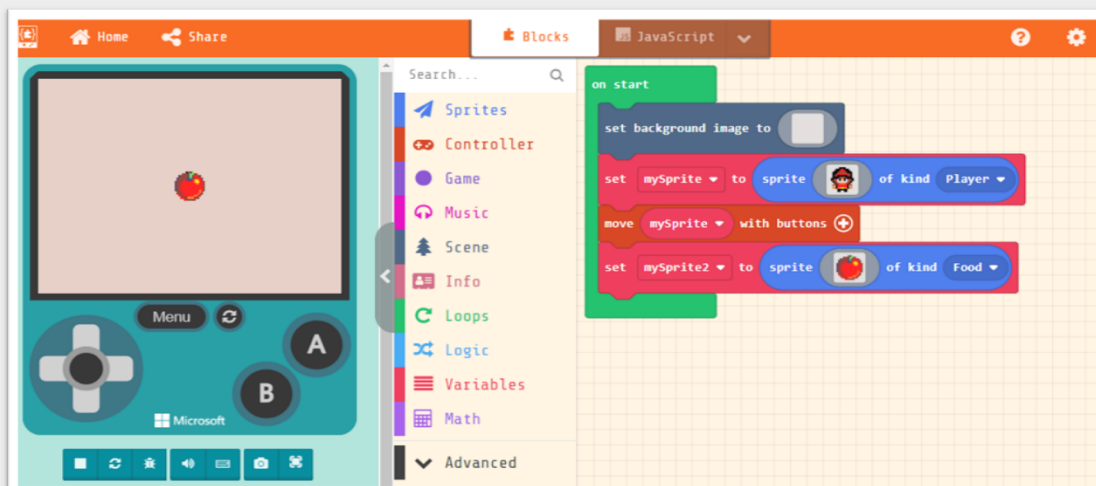
Step 4: In the **Player** block, click on the gray square to open the image editor and select the Gallery view. Find and select a character of your choice.



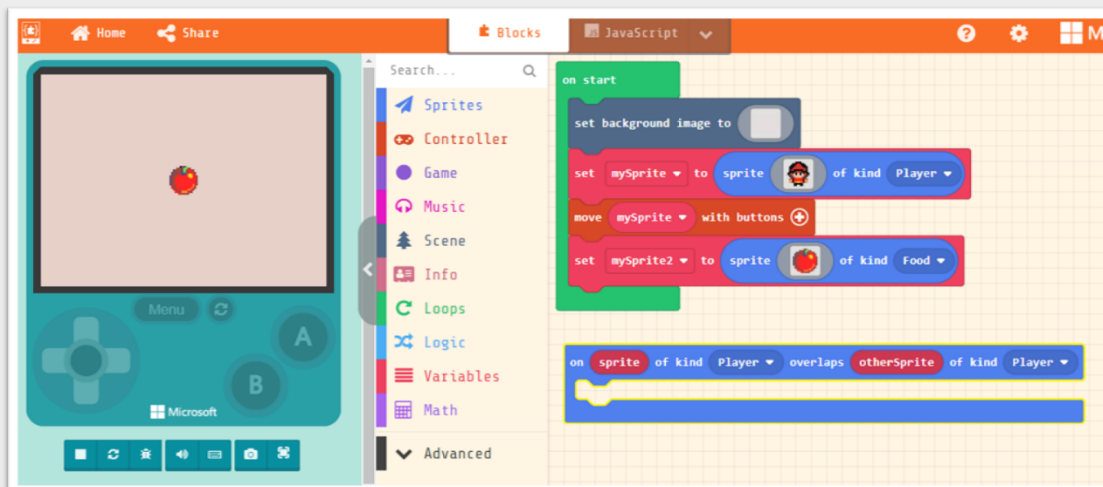
Step 5: Open the Controller toolbox drawer and drag the “move mySprite with buttons” block after the “set mySprite” block. This allows you to move your **Player** sprite around the screen with arrow keys



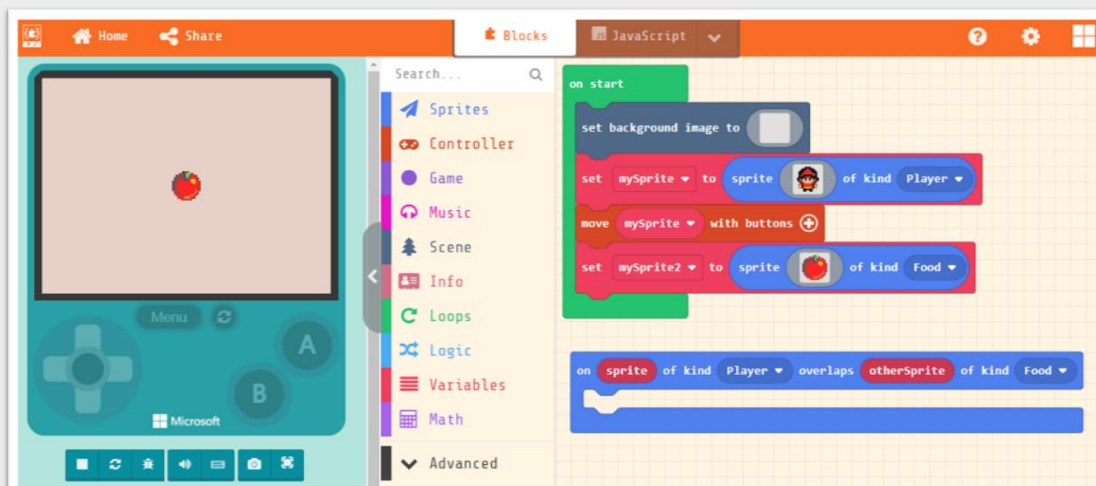
Step 6: Open the Sprites toolbox drawer and drag another “set mySprite2” block into the “on start” block on your workspace. In the kind of Sprite select “Food”.



Step 7: In the Food block, click on the gray square to open the image editor and select the Gallery view. Find and select the image of an apple.



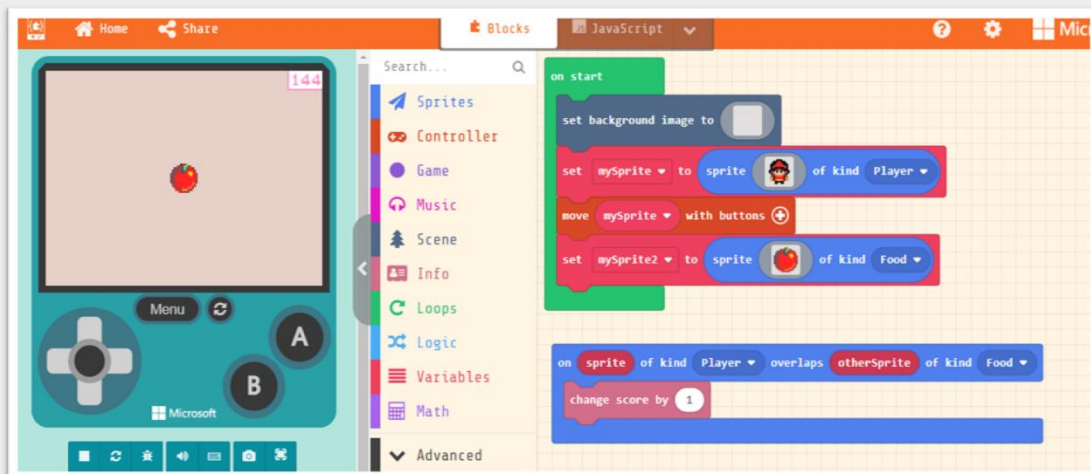
Step 8: Open the Sprites toolbox drawer and drag the “on sprite overlaps otherSprite” block onto your workspace (this can be placed anywhere)



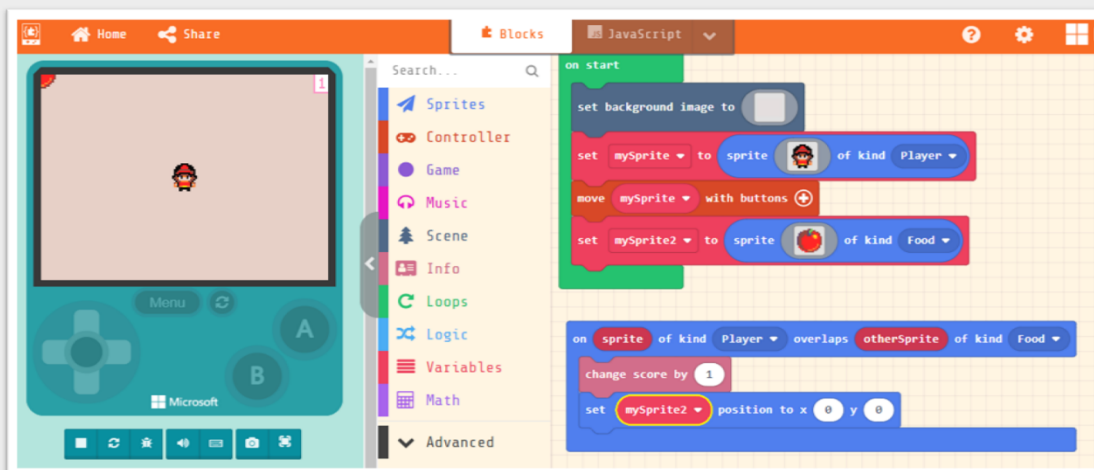
Step 9: In the “on sprite overlaps otherSprite” block, click on the second Player kind after otherSprite to open the menu. Select Food as the kind.



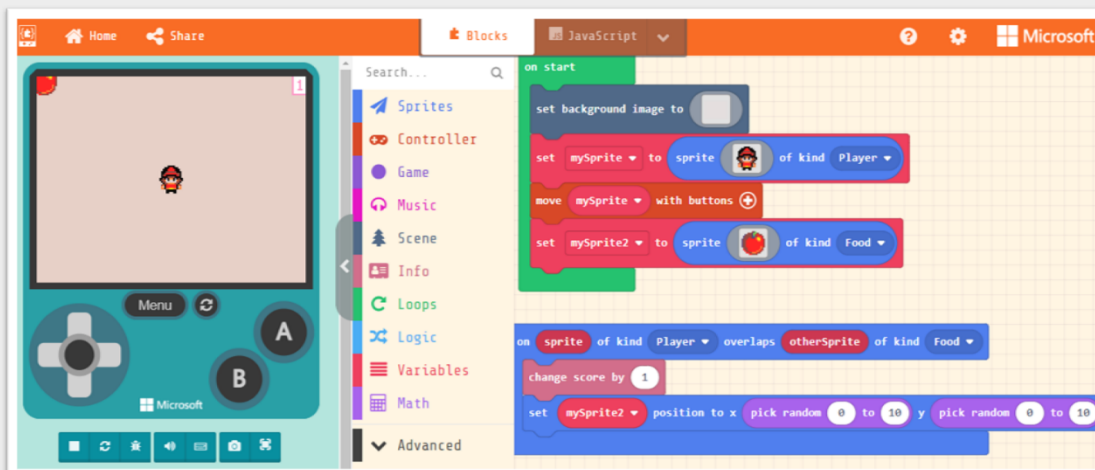
Step 10: When our player overlaps with the apple, let's add a point to our game score. To do so open the Info toolbox drawer and drag the "change score" block into the "on sprite overlaps otherSprite" block.



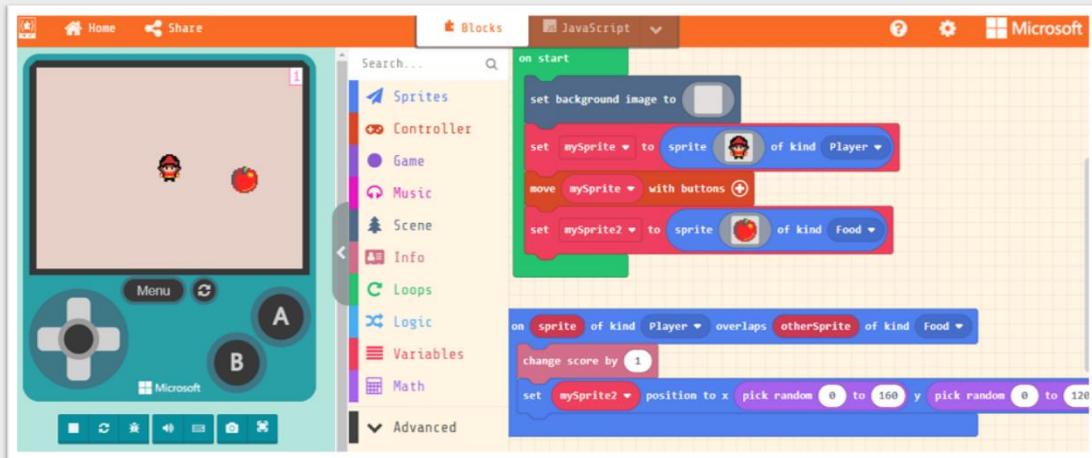
Step 11: Once the sprites overlap, let's set the position for the apple to random locations around the screen. To do so, open the **Sprites** toolbox drawer and drag the "set mySprite position" block into the "**on sprite overlaps otherSprite**" block.



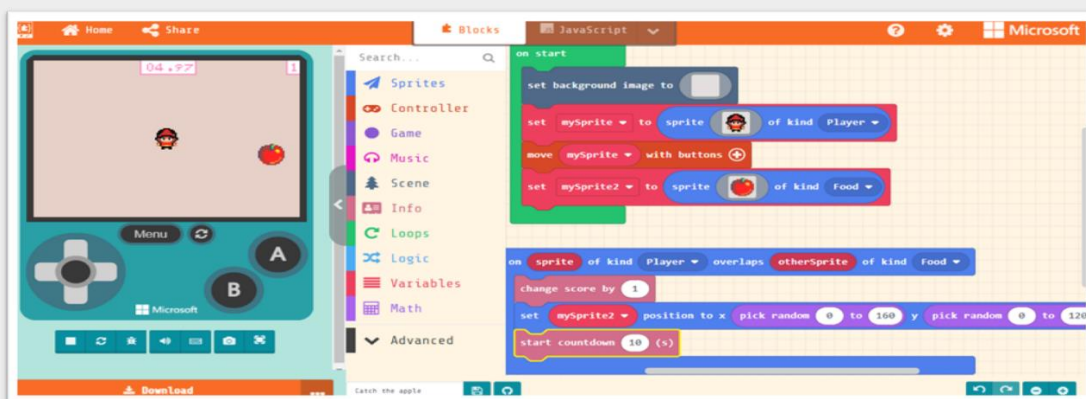
Step 12: In the the “set mySprite position” block into the “**on sprite overlaps otherSprite**” block, click on **mySprite** variable to open the menu, and select your **mySprite2** sprite.



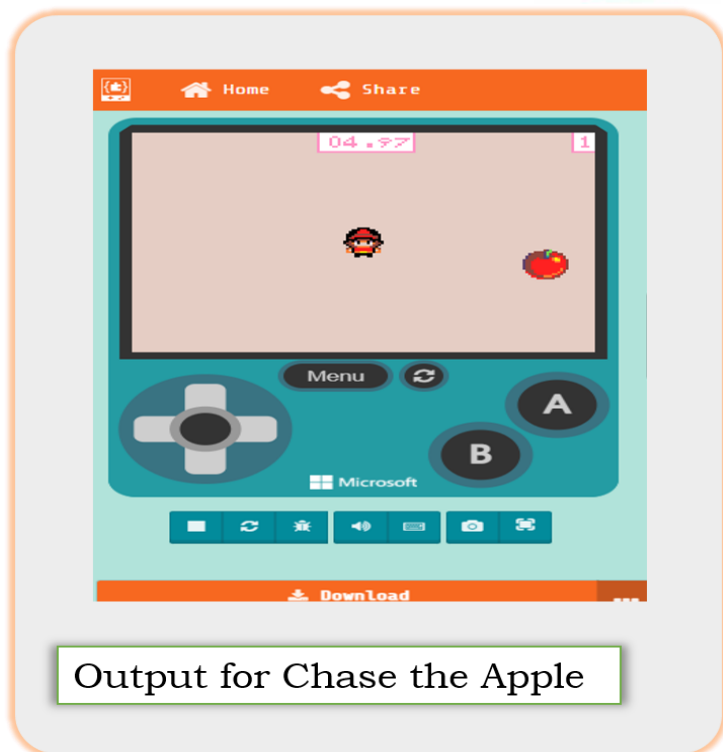
Step 13: Open the Math toolbox drawer and drag two **pick random blocks** on to the workspace. Drop one into the x coordinate of the “set mySprite2 position” block, and the other into the y coordinate replacing the 0 values.



Step 14: The arcade game screen is 160 pixels wide, and 120 pixels high. In the first **pick random** block in the x coordinate of the **set mySprite2** position block, change the maximum value to 160. In the second **pick random** block in the y coordinate, change the maximum value to 120.



Step 15: In order to restart the countdown of the timer each time, open the Info toolbox drawer and drag a **start countdown** block into the “**on sprite overlaps otherSprite**” block on your workspace.



Note: Arcade is just one of the platforms to achieve this output. You can use many similar platforms available online to achieve similar output like – Scratch (<https://scratch.mit.edu/>) and Code.org

ADDING TWO INTEGERS

Chapter: Fun with Functions


Problem Statement: You have learnt about Addition in Mathematics. You can find the addition of numbers through programs too. Consider you have to create a function which returns the value of addition of any given number and returns the result as a return value from the function. Can you write a block code for this function in Minecraft?

Learning Outcomes:

At the end of this exercise, you will learn:

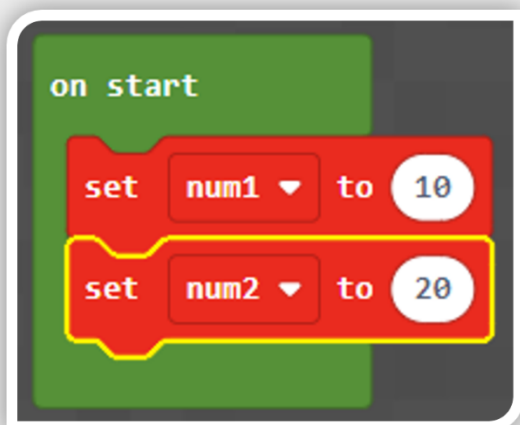
- Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- Create procedures with parameters to organize code and make it easier to reuse.
- Create prototypes that use algorithms to solve computational problems by leveraging previous student knowledge.

Solution: In Minecraft Editor, click on “Make a Function” button from “Functions” link

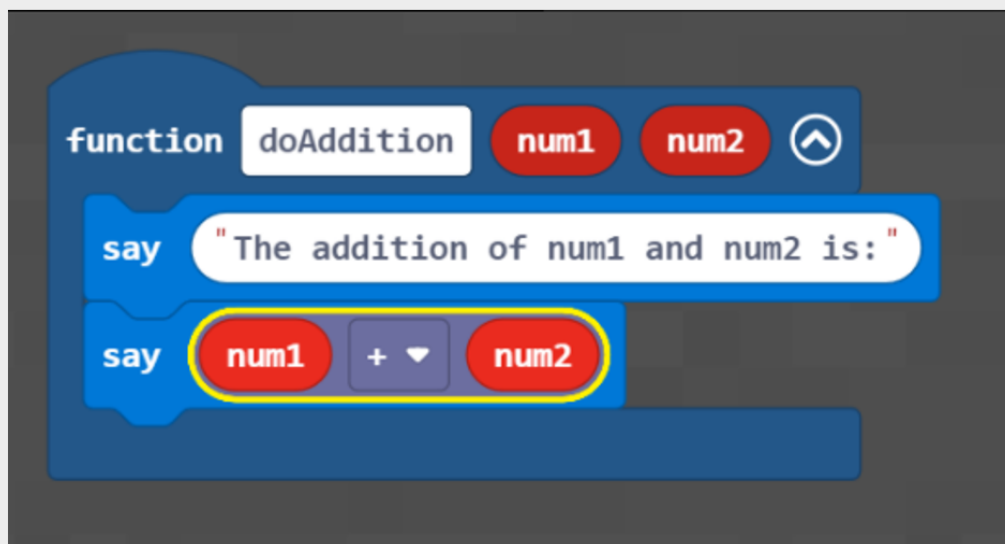


Step 1:
Click on “**Make a Function**” button from “**Functions**” link in toolbox. Name the Function as “*doAddition*”. Click on “**Number**” button and add “*num1*” and “*num2*” as input parameters. Once done, click on “**Done**” button

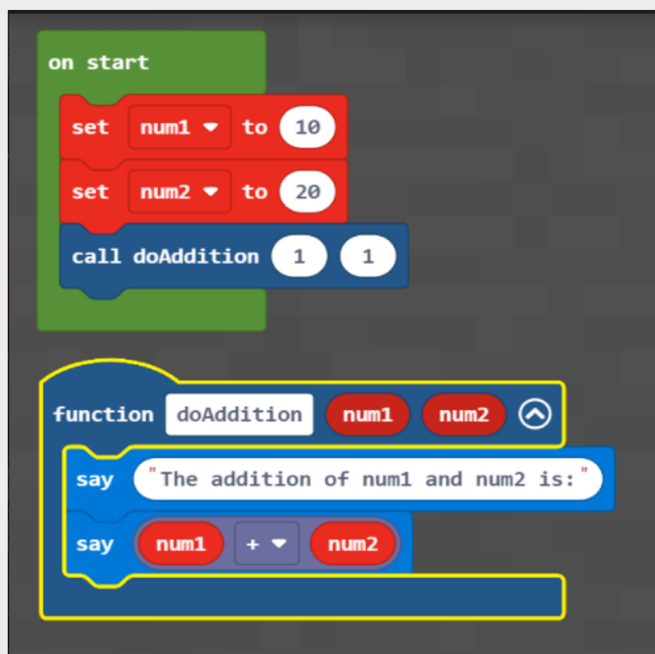
in toolbox.

**Step 2:**

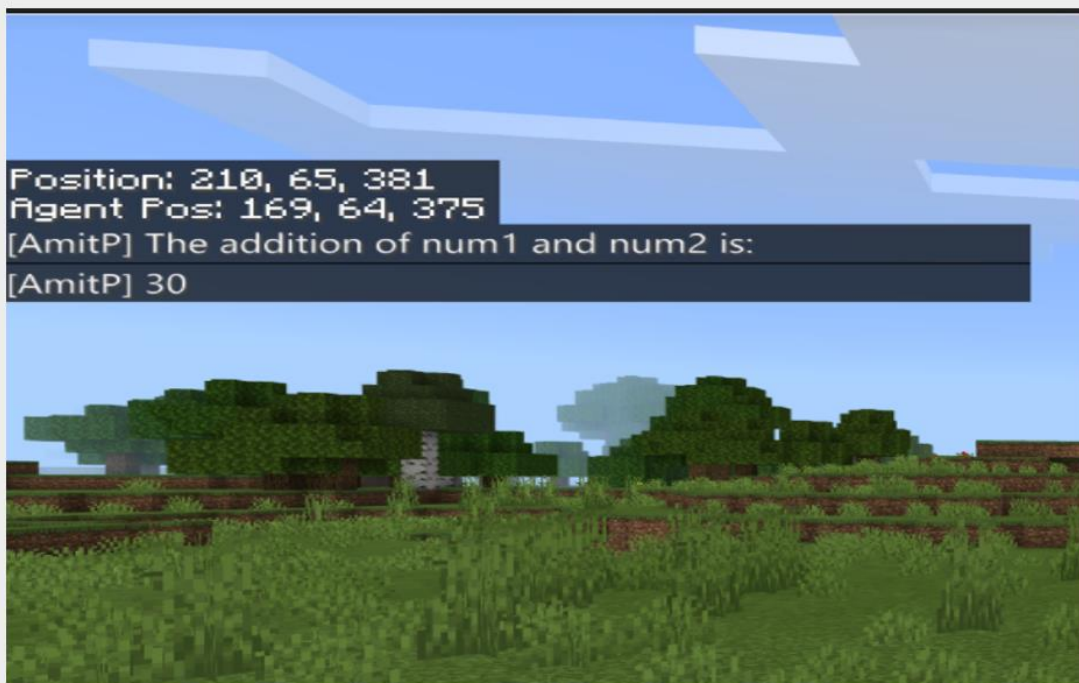
Create two variables "num1", "num2" and set its value to "10" and "20" as shown in the image. Attach these blocks to the "on start" block.



Step 3: Place the "say" block inside the "doAddition" function that we had created earlier.



Step 4: Drag and drop “**call doAddition**” block from “Functions” link in the toolbox and place it inside the “on start” block. Place variables “num1” and “num2” inside “**call doAddition**” block. Now click on play button at the bottom of the screen to see the final output which should be “30”



Final output for activity adding two integers

FINDING THE SQUARE OF A NUMBER

Chapter: Fun with Functions

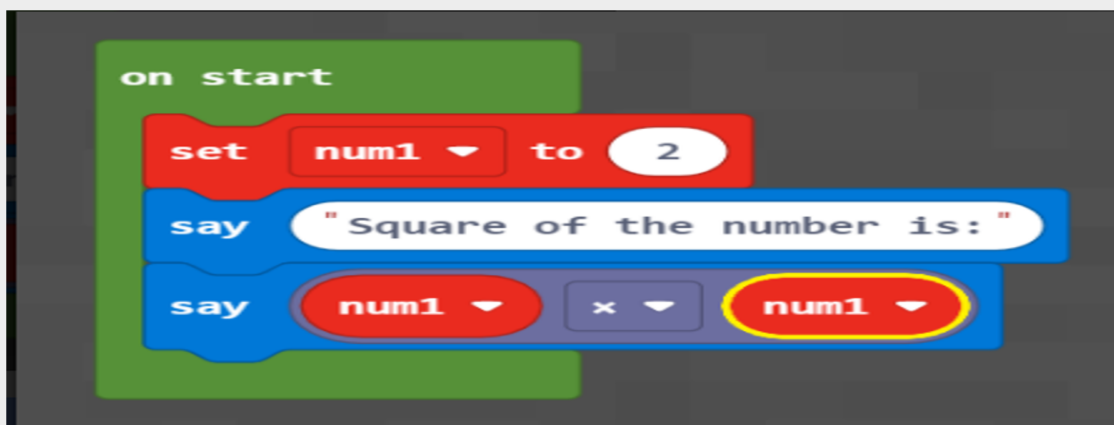
Problem Statement: You have learnt about finding squares in Mathematics. You can find these squares through programs too. Consider you have to create a function which returns the value of square of any given number and returns the result as a return value from the function. Can you write a block code for this function in Minecraft?

Learning Outcomes:

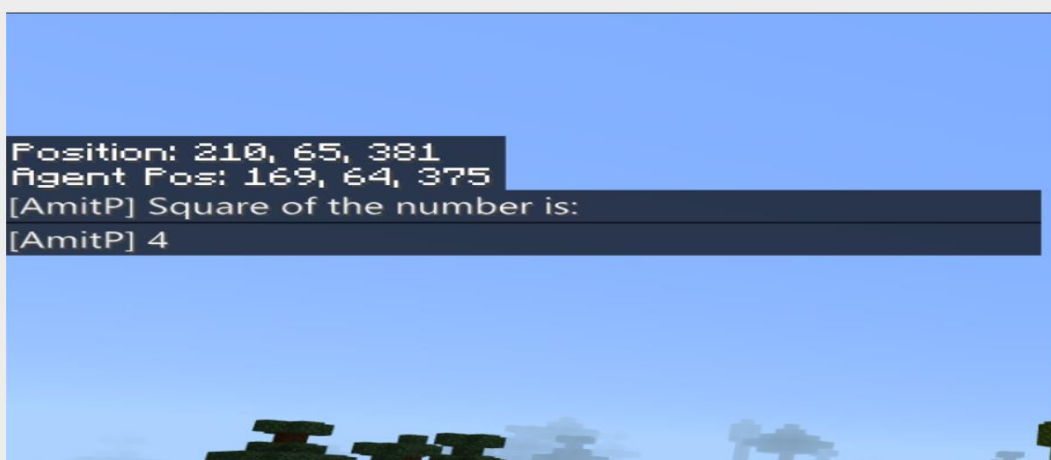
At the end of this exercise, you will learn:

- Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- Create procedures with parameters to organize code and make it easier to reuse.
- Create prototypes that use algorithms to solve computational problems by leveraging previous student knowledge.

Solution: First, create a new project in Minecraft as shown below.



Step 1: Click on “Make a Variable” button from “Variables” link in toolbox. Name the variable as “num1” Drag and drop “set num1 to” block from “Variables” link into the play area and set its value to 2. Once done, place this inside “on start block”. Place the “say” block inside the “on start block” as shown in the image.



Final output for activity finding the square of a number is 4

CAN A FUNCTION RETURN VALUE

Chapter: Fun with Functions

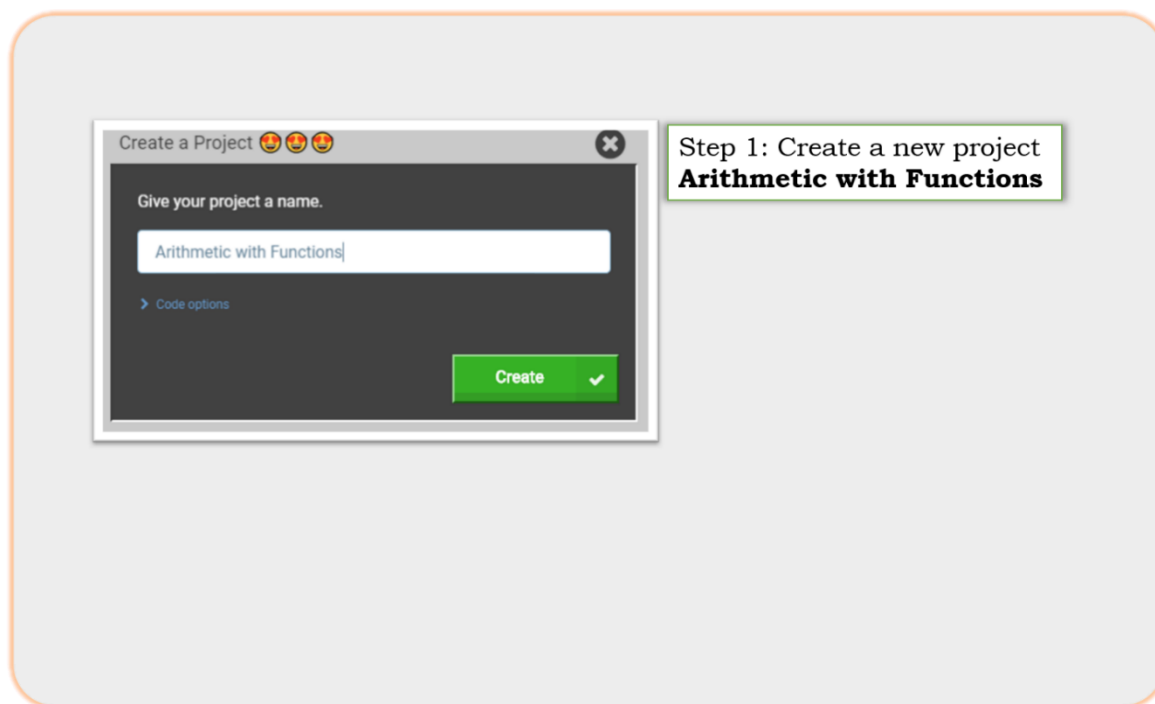
Problem Statement: You have learnt about finding squares and cubes of numbers in Mathematics. You can find these squares and cubes through programs too. Consider you have to create a function which returns the value of square and cube of any given number and returns the result as a return value from the function. Can you write a block code for this function in Minecraft?

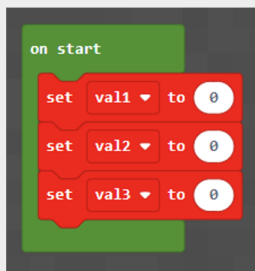
Learning Outcomes:

At the end of this exercise, you will learn:

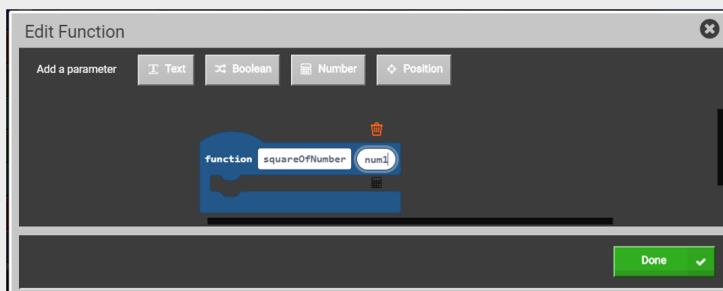
- Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
- Create procedures with parameters to organize code and make it easier to reuse.
- Create prototypes that use algorithms to solve computational problems by leveraging previous student knowledge.

Solution: First, create a new project in Minecraft as shown below.

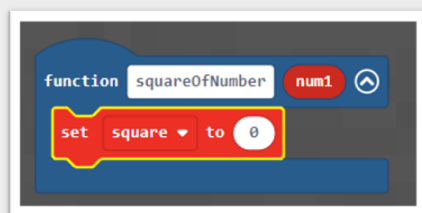




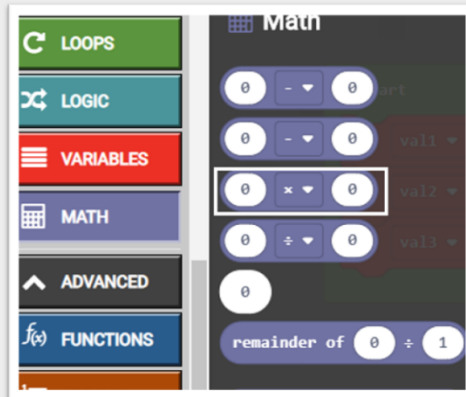
Step 2: Create three variables val1 , val2 & val3. Create three blocks “set val1 to” , “set val2 to” & “set val3 to” and place them under on “on start” block



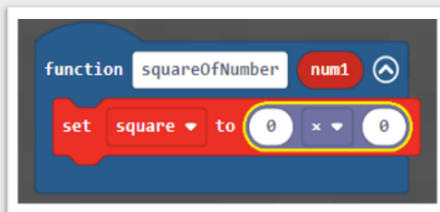
Step 3: Create a function named squareOfNumber having a numeric type parameter.



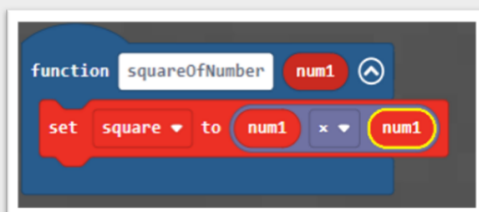
Step 4: Create a variable named square and create a block “**set square to**” and place it inside function squareOfNumber



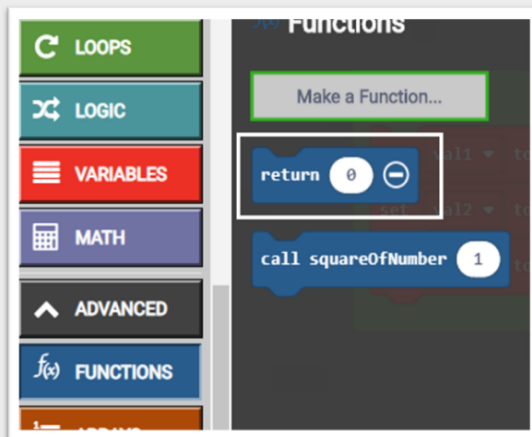
Step 5: Select the Math block type and select a multiplication block



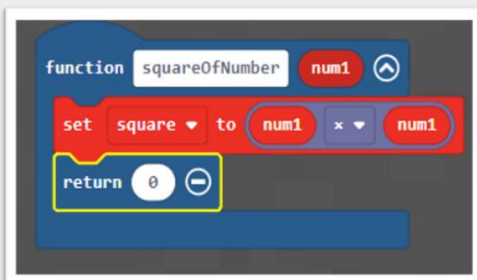
Step 6: Attach the multiplication block to the “set Square to” block inside the function squareOfNumber



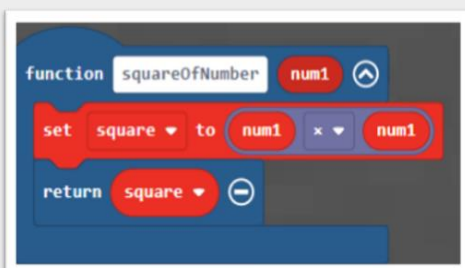
Step 7: Set the values of each number inside the multiplication block to num1



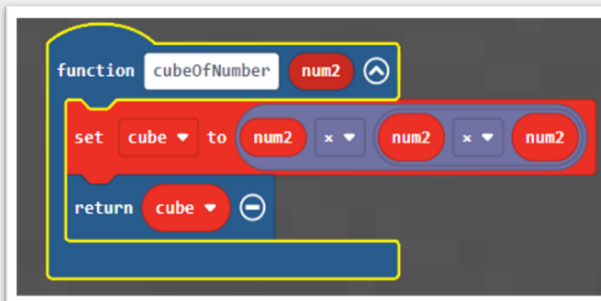
Step 8: Select block type Functions and select “return” block



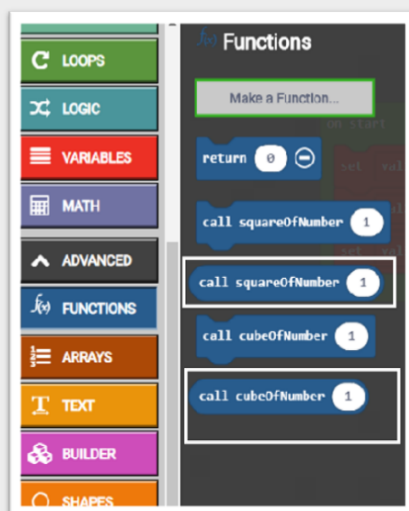
Step 9: Place the return block inside the squareOfNumber function after the “set Square to” block.



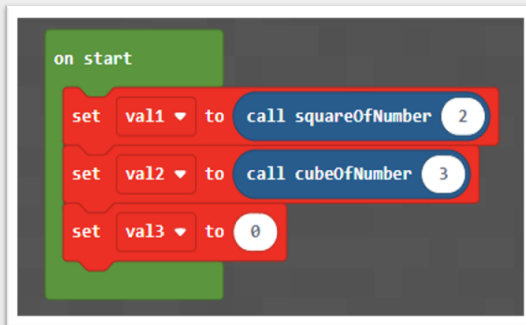
Step 10: Set value inside return block to square variable



Step 11: Similarly, we can create a function cubeOfNumber having a numeric parameter num2 (try to derive the shown block code as a practice exercise)



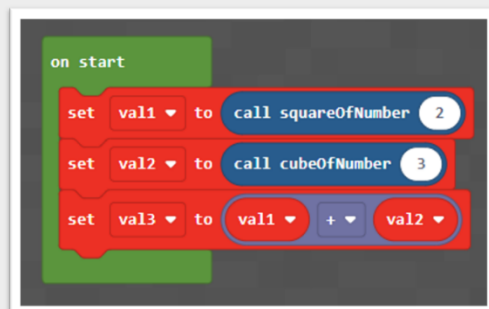
Step 12: Open block type Functions. Select one “call squareOfNumber” block and one “call cubeOfNumber” block



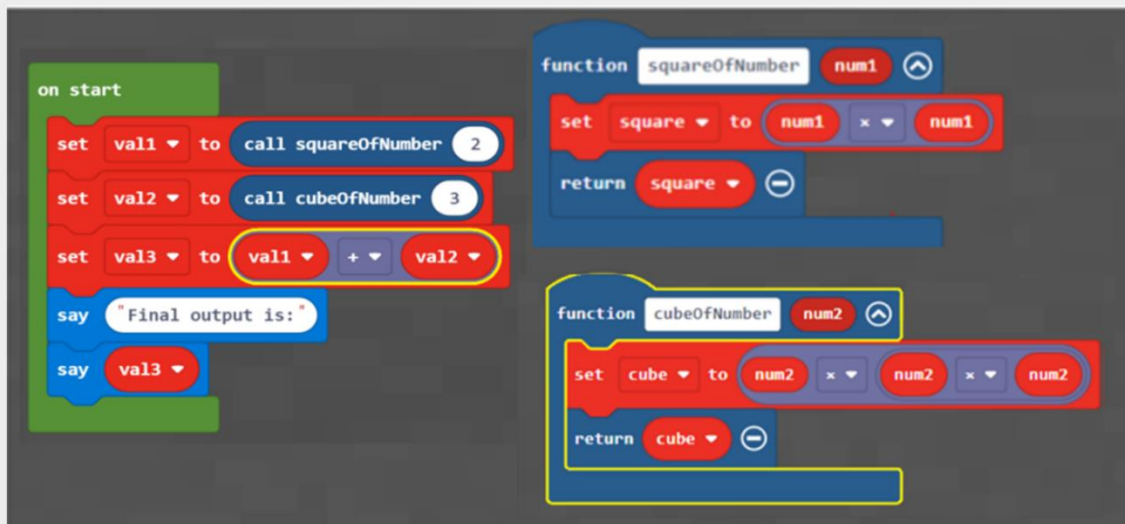
Step 13: Attach “call squareOfNumber” block to “set val1 to”.
Change value in “call squareOfNumber” to 2
Attach “call cubeOfNumber” block to “set val2 to”.
Change value in “call cubeOfNumber” to 3



Step 14: Select Math block type and select an addition block



Step 15: In the addition block set value on left hand side as val1 and right-hand side as val2.
Then attach the addition block to the “set val3 to” block



Step 16 : Place the “**say**” block inside the “**on start**” that we had created earlier. The final block code appears as shown in the image

```
Position: 210, 65, 381
Agent Pos: 169, 64, 375
[AmitP] Final output is:
[AmitP] 31
```

Final output for the activity can a function return value

CALCULATING AREA OF A CIRCLE

Chapter: Fun with Functions

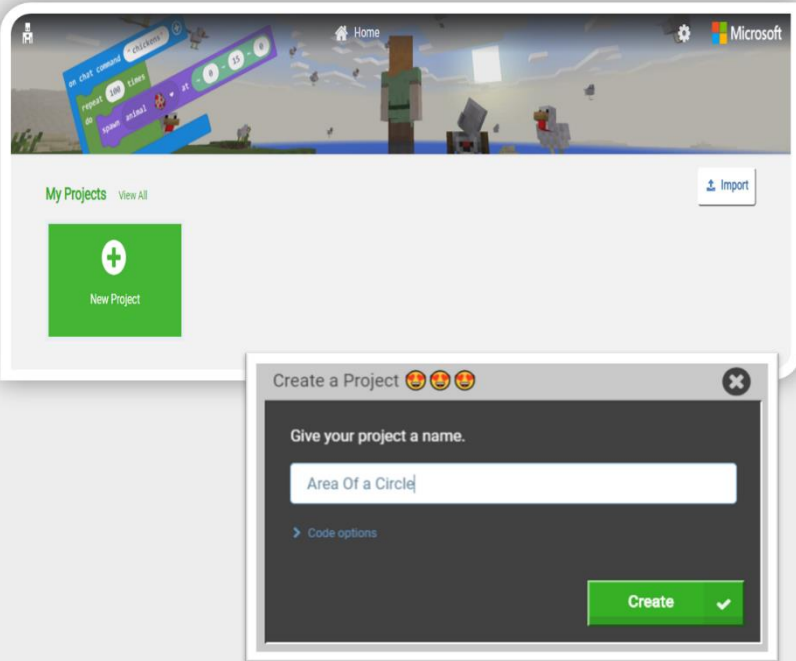
Problem Statement: You must have calculated area of a circle in your mathematical class many a times manually. Programming will help you do this same task with more accuracy and increased speed for larger data sets. Can you write a block code in Minecraft to calculate the area of a circle?

Learning Outcomes:

At the end of this exercise, you will learn:

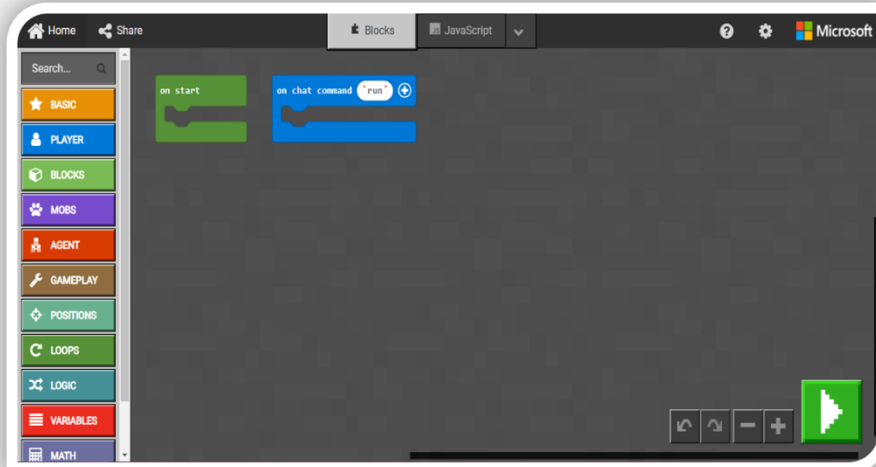
- Learn how to implement various programming concepts that we have learnt so far – variables, variable initialization, performing arithmetic operations on different variables and using sequencing to code.
- Learn how to write code for calculating area of various mathematical shapes.

Solution: Let us now see how we can calculate the area of a circle in Minecraft. First, create a new project as shown below.



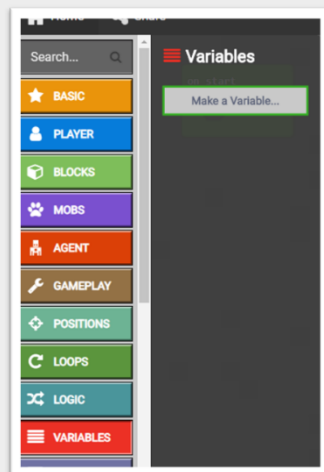
Creating New Project
You can create a new project by clicking on green box labeled as 'New Project'. A dialog box will appear prompting you to give a project name.

Giving Your Project A Name
You need to type down a name in the text and click on 'Create' button



Minecraft Code Editor

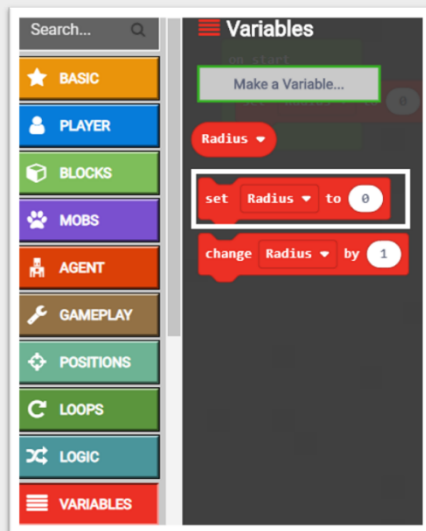
Once you create your project, you should see the editor like this.



Step 1: Go to block type Variables and select **Make a Variable**



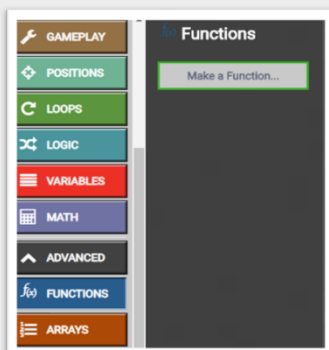
Step 2: Create a new variable called radius



Step 3: From the **VARIABLES** menu select “**Set Radius to**” block



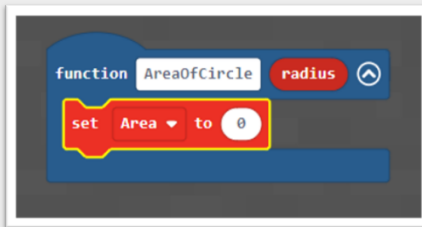
Step 4: Place the “**set Radius to**” block inside on start block and change the value to **5**



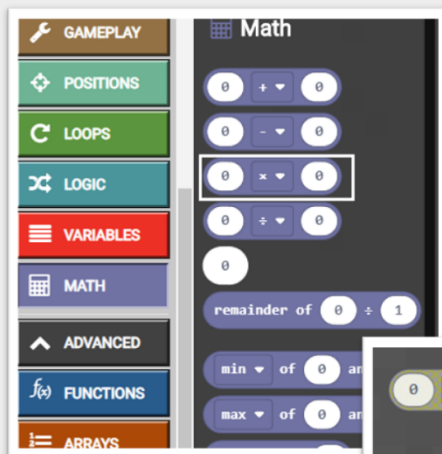
Step 5: Go to block type **FUNCTIONS** and click on “**Make a Function**”



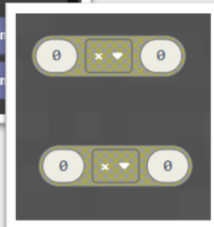
Step 6: Create a function with name “AreaOfCircle” having one numerical parameter named radius

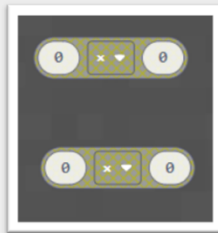


Step 7: Create a variable named Area select the “set Area to” block and place it inside function “AreaOfCircle”



Step 8: Click on the Math block type and select the instance of the highlighted. Need to do this twice to create two multiplication blocks





Step 9: Drag one of the multiplication blocks on top of another so that they become one multiplication block of 3 numbers.



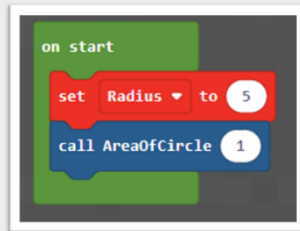
Step 10: Drag the multiplication block of 3 numbers to "Set Area to" block inside the function block "AreaOfCircle"



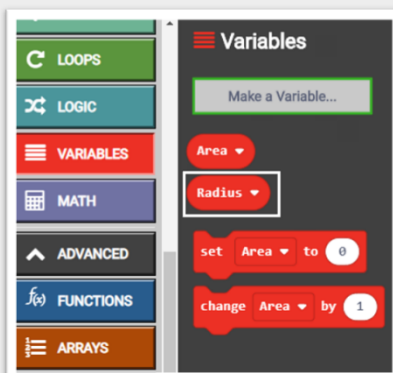
Step 11: Inside the function “AreaOfCircle”, inside the “set Area to” block, set the first number as 3.14 and the second & third number to the parameter radius



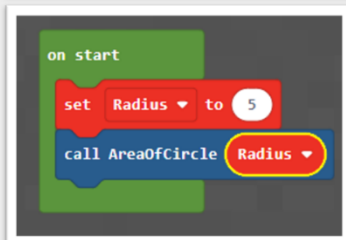
Step 12: Open the Function block type & select “call AreaOfCircle” block



Step 13: Drag the “**call AreaOfCircle**” block into “**on start**” block under “**set Radius to**” block



Step 14: Select the Radius variable from the **VARIABLES** block type



Step 15: Set value of the parameter passed to the **AreaOfCircle** function to **Radius**



Step 16: We now have our block code ready to calculate the area of a circle

CREATE AN ARRAY AND CALCULATE ITS LENGTH

Chapter: Understanding Arrays and Collections

Problem Statement: In this grade, you have learnt the concept of arrays and various actions that we can perform on an array in programming. Explain step by step how to create an Array and calculate its length using Minecraft.

Learning Outcomes:

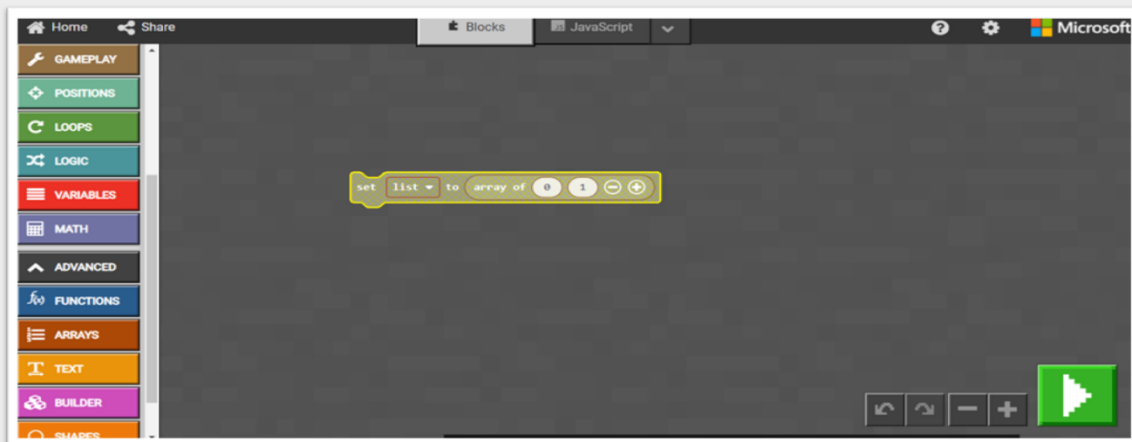
At the end of this exercise, students will learn:

- Learn how to translate between different bit representations of real-world phenomena, such as characters, numbers and images.
- Learn how to create an array and calculate its length.

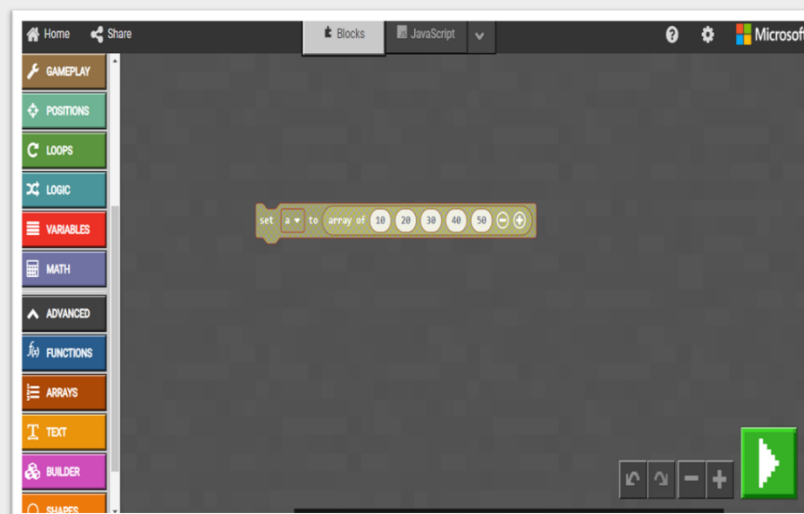
Solution: First, create a new project in Minecraft.



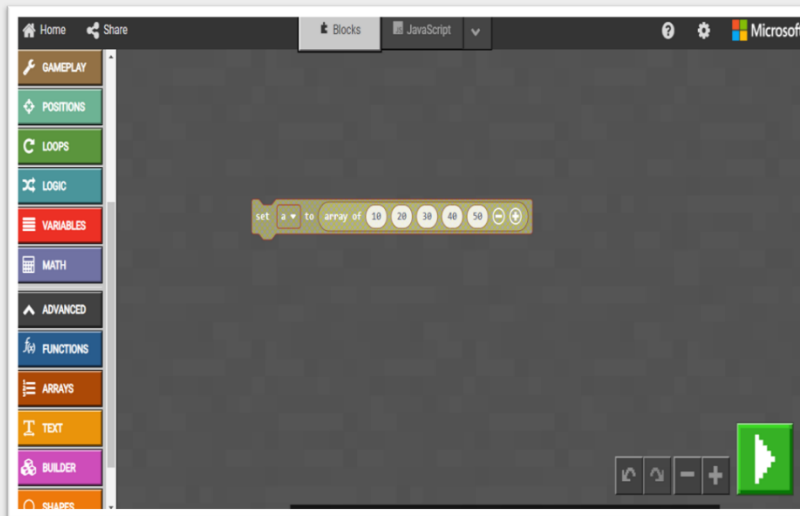
Step 1: Create a new project in Minecraft. Click on “Variables” link from toolbox and click on “Make a Variable”. Create a Variable “a” and Click on “Ok” button



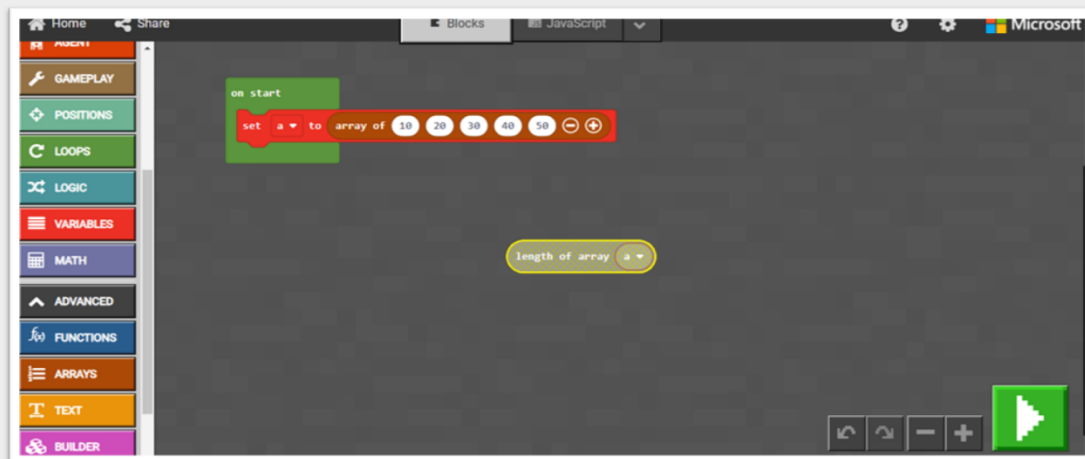
Step 2: From the Toolbox, click on link "Arrays". From the sub list, drag and drop block of "set list to" to the play area



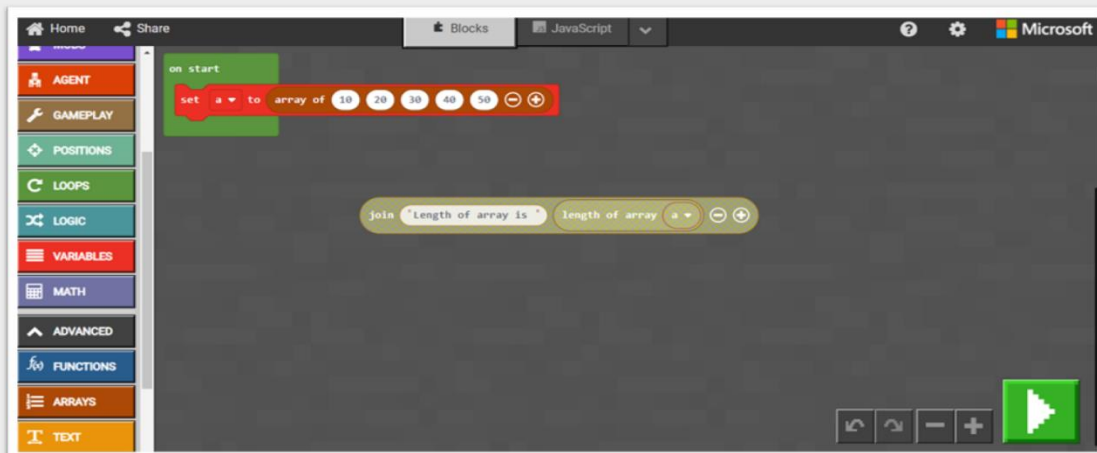
Step 3: In the set block, click on the drop down which says "list" and select variable "a" that we had earlier created. In the next text boxes you can rewrite values such as "10", "20", click on "+" icon and few more elements as shown in the image.



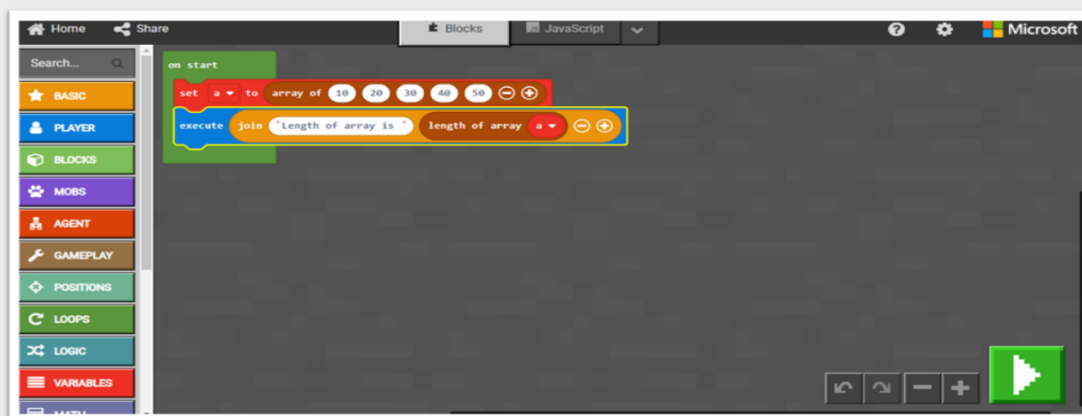
Step 4: Now click on “Loops” link from the tool box. Drag and drop block “On Start” into the play area. Following this, attach the “set” block that we had created into “On Start” block as shown in the image.



Step 5: Click on “Arrays” link from Toolbox. Drag and drop “Length of Array” block to the play area. In “Length of Array” block, select “a” in the drop down which has a default value of “list”



Step 6: Click on “Text” link from the Toolbox. Drag and drop “join” block into play area. In “join” block, replace first text box with “Length of array is “ and fix “length of array” block second text box as shown in the image.



Step 7: Click on “Player” link from the Toolbox and drag and drop “execute” block in the play area. Now, append “join” block that we had created into “execute” block and place “execute” block below “set” block on “on start” block as shown in the image. Finally, click on green play button in the bottom on the page. It will display an output “Array length is “5” on the console.



BUILDING A ZOO

Chapter: Understanding Arrays and Collections

Problem Statement: You must have played many games where you saw a zoo full of different animals and you must have performed different actions for adding deleting these animals and saving them from each other to proceed in the game. In this activity, you will implement the concept of arrays that you learnt about programming. To complete this activity, create an array in Minecraft and fill it up with animals of your choice.

Learning Outcomes:

At the end of this exercise, you will learn:

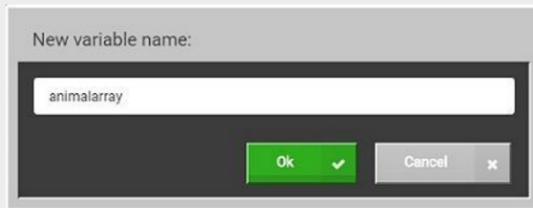
- Learn how to use flowcharts and/or pseudocode to address complex problems as algorithms.
- Create clearly named variables that represent different data types and perform operations on their values.
- Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
- Create procedures with parameters to organize code and make it easier to reuse.
- Use abstraction to decompose a problem into sub problems.
- Construct a program as a set of step-by-step instructions to be acted out.
- Implement problem solutions using a block based visual programming language.
- Generate and compare multiple possible solutions to a problem based on how well each is likely to meet the criteria and constraints of the problem.

Solution:

You can store animals in an array and spawn them wherever you like. We'll use this capability to build a fenced-in animal pen and create an instant zoo anytime we want. When this project starts up, it will create an array and fill it with animals of your choice.

Step 1: Create a new MakeCode project called "Zoo".

Step 2: In "Loops", there is an "on start" that will run its commands once, as soon as the project starts up. Drag that block into the coding Workspace.

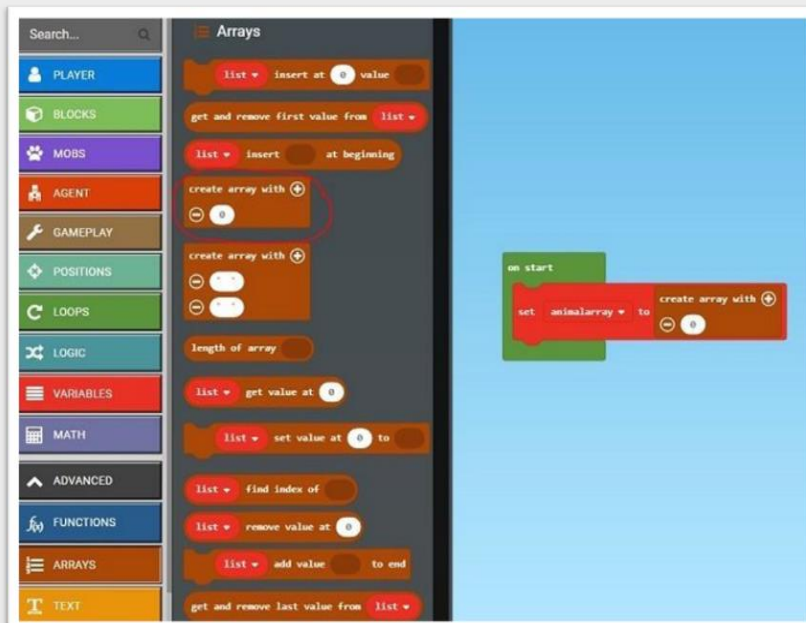


Step 3: From “Variables” , click the **Make a Variable** button. Name this variable “animalarray”, and click “Ok”

Step 4: From “Variables”, drag “set” into the “On start” block.

Step 5: Using the drop-down menu in “set”, select the animalarray variable.

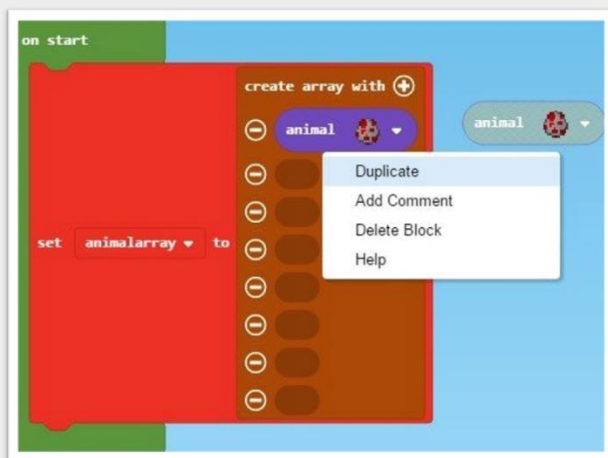
Step 6: Click on the Advanced tab in the Toolbox to display the “Arrays” Toolbox drawer.



Step 7: From “Arrays”, drag a “create array with” into “set animalarray” to.

Step 8: Click the Plus (+) sign on “create array with” to add 7 more slots in your array. The total length of your array should be 8.

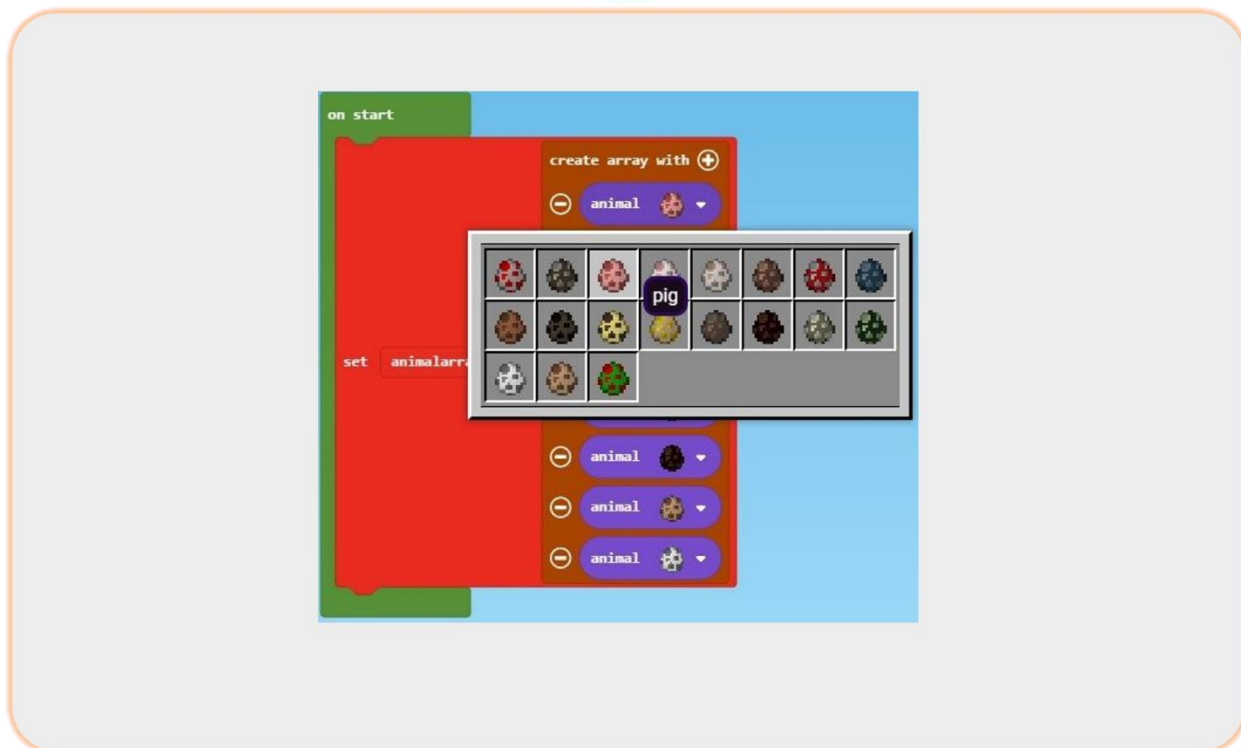
Step 9: From “MOBS”, drag an Animal block into the first slot of “create array with”.



Step 10: Populate the rest of your array with animal blocks. You can right-click on “animal” and select Duplicate to make copies of this block.

Step 11: Create a zoo with 8 different types of animals. Be aware that certain animals will eat other animals! For example, ocelots and chickens don’t get along very well. Think about what kind of zoo you want, and plan accordingly.

Step 12: Using the drop-down menus in the “animal” blocks, select different types of animals in your array.



Step 13: Now that you have your AnimalArray set up, let's work on creating a fenced-in enclosure for your zoo. You will use the "BUILDER" blocks for this. The Builder is like an invisible cursor in the game that can place blocks along a path very quickly. You will direct the Builder to go to a point in the southeast corner, and create a mark, which is an invisible point of reference. Then you will give it a series of commands to make it trace out a square. Finally, the builder is able to place fences along this path.

Step 14: From "PLAYER", drag an "on chat command" block to the Workspace.

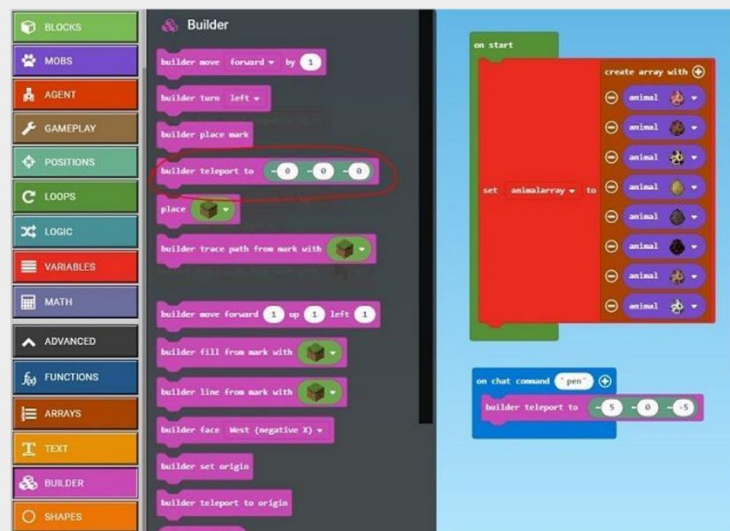
Step 15: Rename the command "pen".

Step 16: Click on the Advanced tab in the Toolbox to display the "BUILDER" Toolbox drawer.

Step 17: From "BUILDER", drag "builder teleport to" into "on chat command "pen""

Step 18: Recall that Minecraft coordinates are always specified in X, Y, Z coordinates where X is west to east and Z is north to south. We want the Builder to start in the northeast corner of the pen in relation to the player, so go ahead and change the coordinates to specify a location 5 blocks east and 5 blocks north of your position.

Step 19: In "builder teleport to", change the position values to (~5, ~0, ~-5).



Step 20: Let's make sure the Builder is facing the right way so that it draws the pen around you. After the builder is facing the correct direction, you can then have it place a starting mark.

Step 21: From "BUILDER", drag "builder face" out and under "builder teleport to". The default 'face West' is fine.

Step 22: Next, from "BUILDER", grab a "builder place mark" to put after the "builder face".

Step 23: From "LOOPS", drag a "repeat" loop and place it after "builder place mark". A square has four sides so repeating four times is great.

Step 24: From "BUILDER", drag a "builder move" into the "repeat" loop.

Step 25: Type 10 into "builder move" to make the sides of your pen 10 blocks.

Step 26: From "BUILDER", drag "builder turn" after the "builder move" block.

Step 27: From "BUILDER", place a "builder trace path from mark" after the "repeat" loop.

Step 28: Using the drop-down menu in "builder trace path from mark", select an Oak Fence.

Step 29: Now, open a Flat World in the Minecraft game, and type "pen" in the chat window. You should see a pen being built all the way around you! For an extra challenge, you might try to get the Builder to add a fence gate



Step 30: Now comes the fun part. The array is loaded up with animals, the pen has been built... it's time to let them loose! For this command, we will simply go through the entire array and for each animal in the array, we will spawn two of them a few blocks away from you but still within the pen.

Step 31: From "PLAYER", get an "on chat command" and rename it "zoo".

Step 32: From "LOOPS", drag a "for element" into your "on chat command "zoo"".

Step 33: In the "for element", use the drop-down menu for the 2nd slot to select animalarray.



Step 34: From “MOBS”, drag a “spawn animal” block and place it inside “for element”.

Step 35: From “VARIABLES”, drag the value variable into the “spawn animal” block, replacing the default chicken animal.

Step 36: Adjust the coordinates in “spawn animal” to (~3, ~0, ~0), so the animals will spawn a few blocks away from the Player.

Step 37: To create pairs of animals, right-click on the “spawn animal” block to Duplicate it. You could also use a loop here if you choose.

Step 38: Go back into your Minecraft world, and type the command “zoo” into the chat window, and watch the animals appear!

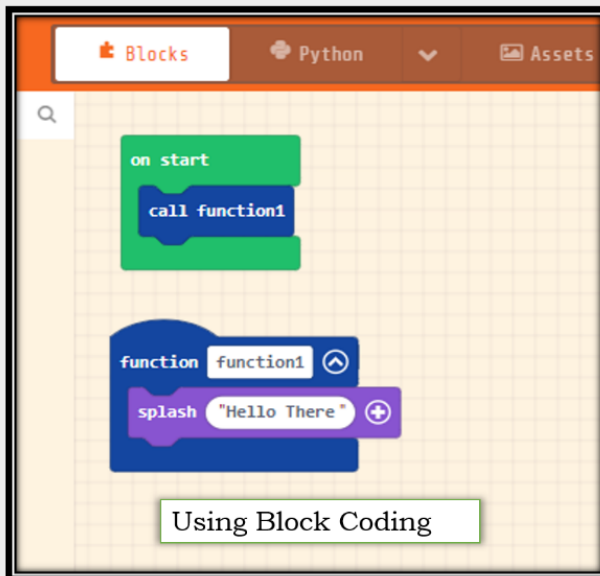


EXAMPLES OF SIMPLE FUNCTIONS IN ARCADE

Chapter: Fun with Functions

Problem Statement: In this activity, you will implement the concept of functions that you learnt about programming.

Example 1: Calling a function which has no parameters.



```
1 def function1():  
2     game.splash("Hello There")  
3  
4 function1()
```

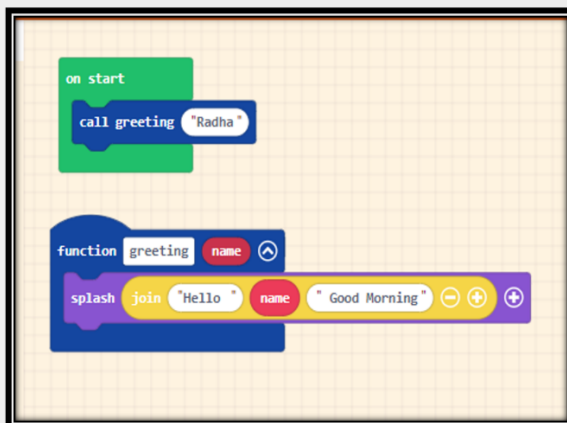
Using Python

Example of Calling a function without any parameter



Output for calling a function without any parameter

Example 2: Calling a function with a single parameter



Using Block Coding

```
1 def greeting(name):  
2     game.splash("Hello " + name + " Good Morning")  
3  
4 greeting('Radha')
```

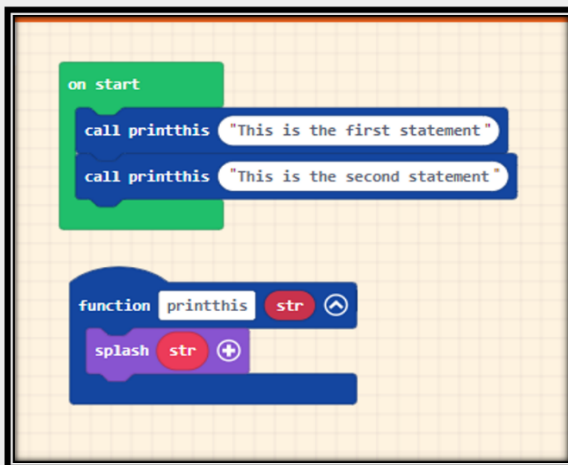
Using Python

Example of Calling a function with a single parameter



Output for calling a function with a single parameter

Example 3: Calling a function to print statements.



Using Block Coding



Using Python

Example of Calling a function with a single parameter



Click on A to print the next statement

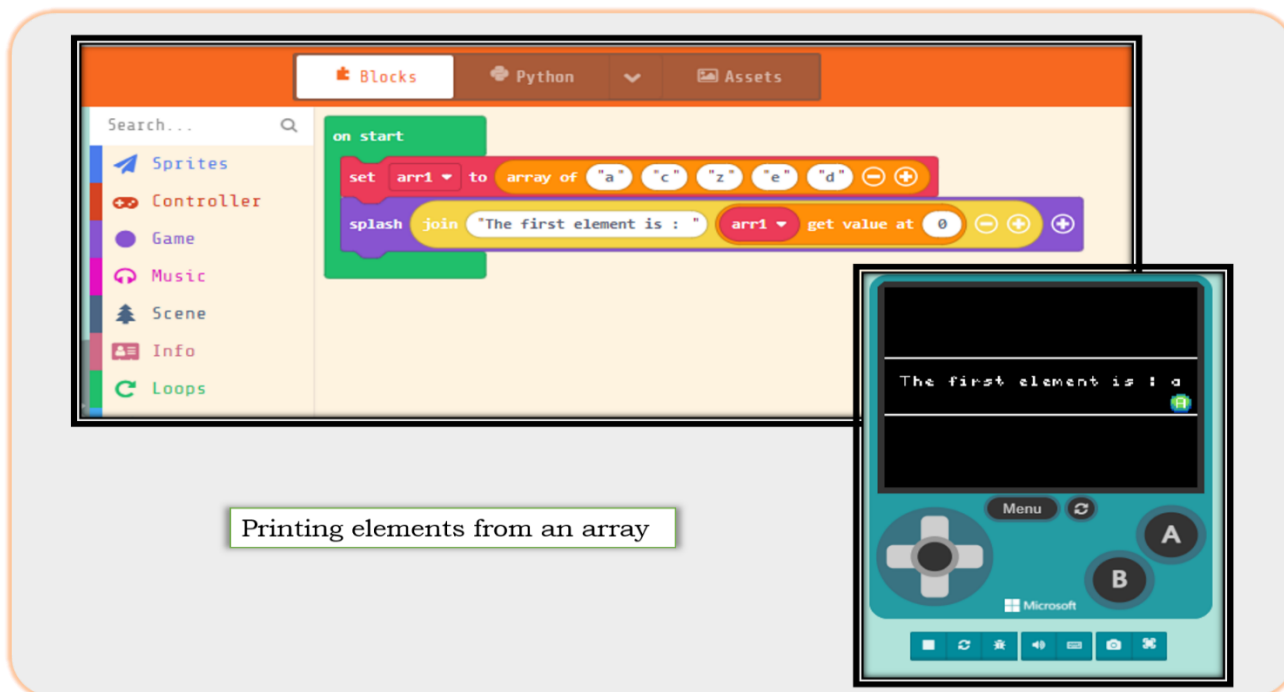
Output for example 3

EXAMPLES OF ARRAYS IN ARCADE

Chapter: Understanding Arrays and Collections

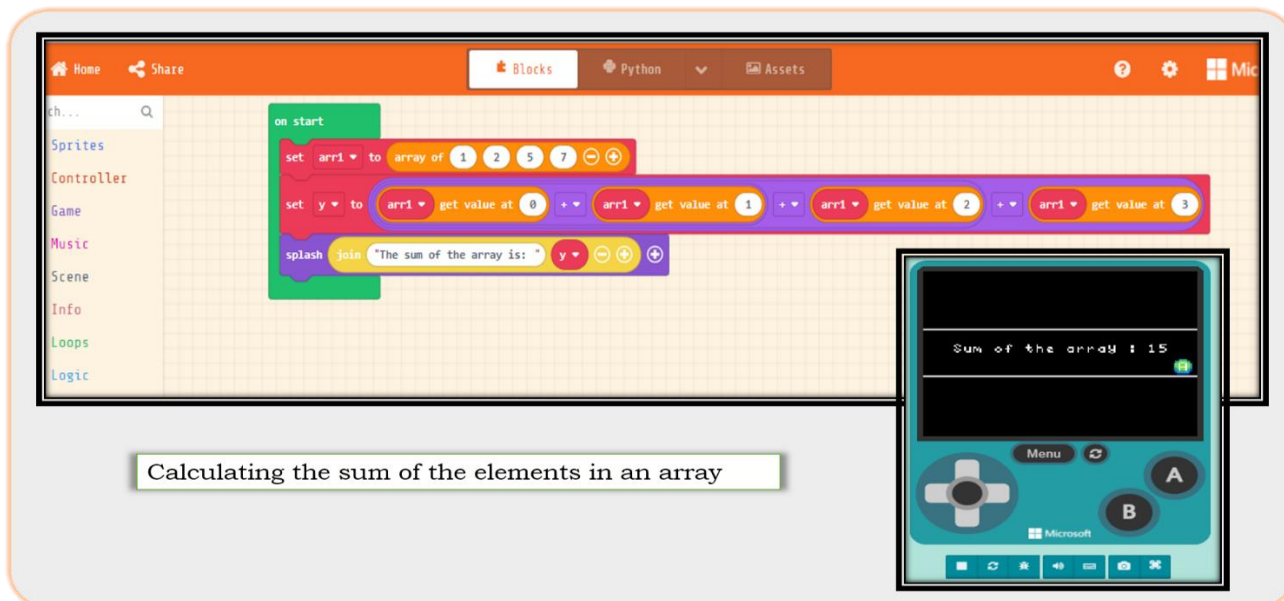
Problem Statement: In this activity, you will implement the concept of arrays that you learnt.

Example 1: Printing the first element of the array



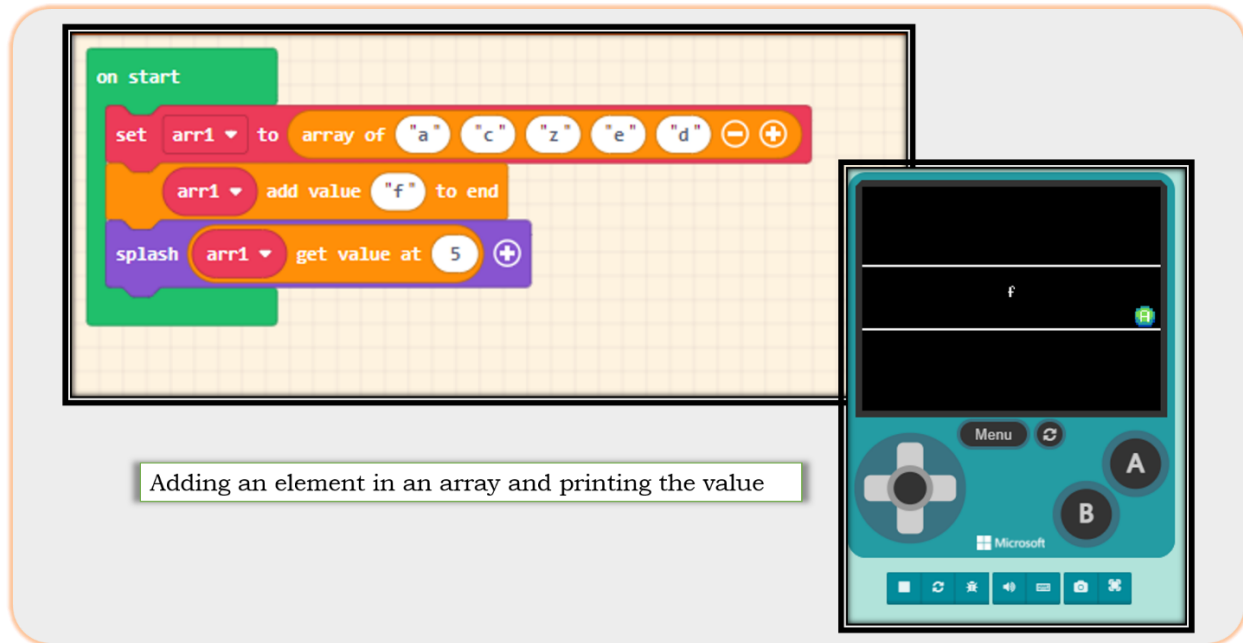
Printing elements from an array

Example 2: Calculating the sum of the elements in an array



Calculating the sum of the elements in an array

Example 3: Adding an element in an array



The screenshot displays the Arcade IDE interface. On the left, a script is written within an "on start" block:

```
on start
  set arr1 to array of "a" "c" "z" "e" "d"
  arr1 add value "f" to end
  splash arr1 get value at 5
```

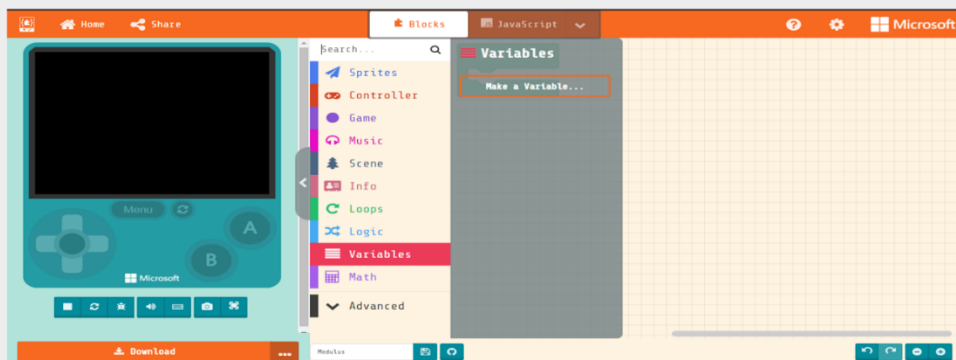
Below the script, a text box contains the description: "Adding an element in an array and printing the value". On the right, a game preview window shows a black screen with the letter "f" in the center, indicating the output of the "splash" command. The preview window also features a virtual joystick and buttons labeled "Menu", "A", and "B".

BUILDING A CALCULATOR

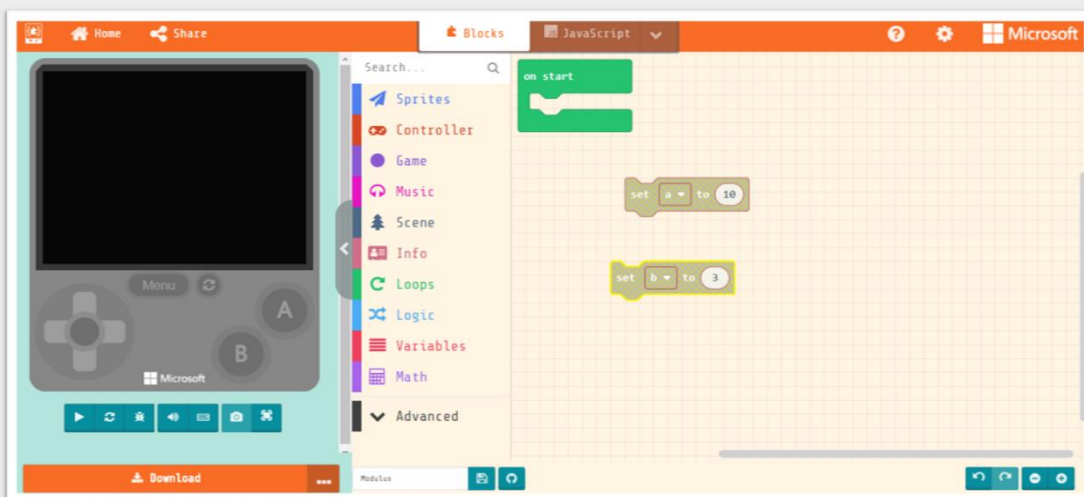
Chapter: Hello World with code

Problem Statement: In this activity, you will build a calculator in MakeCode

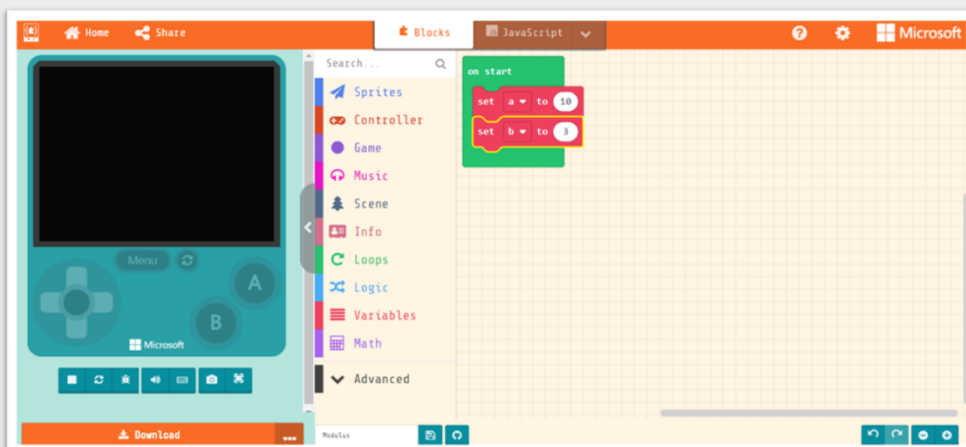
To understand these operations better, let us now perform the below activity. You need to open MakeCode editor for this activity.



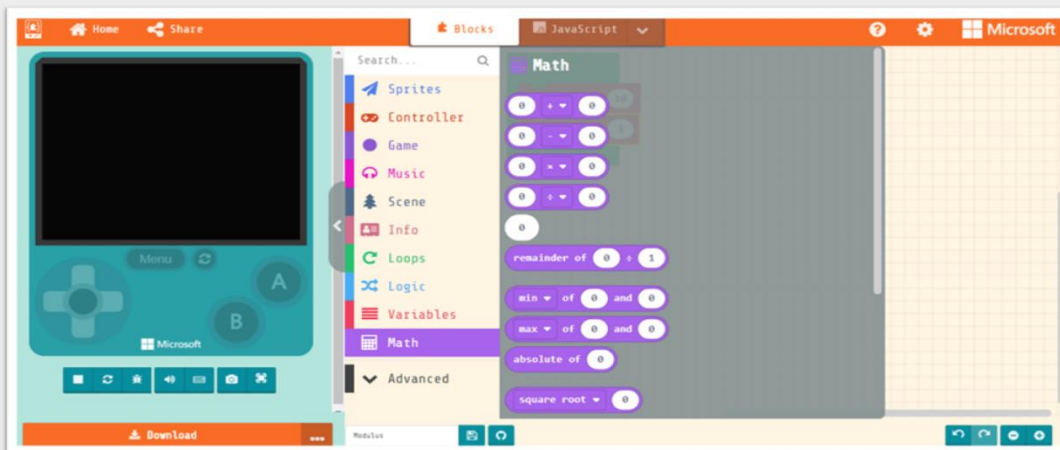
Step 1: In Arcade MakeCode Editor, click on the “Variables” link from the ToolBox and then click on “Make a Variable” link



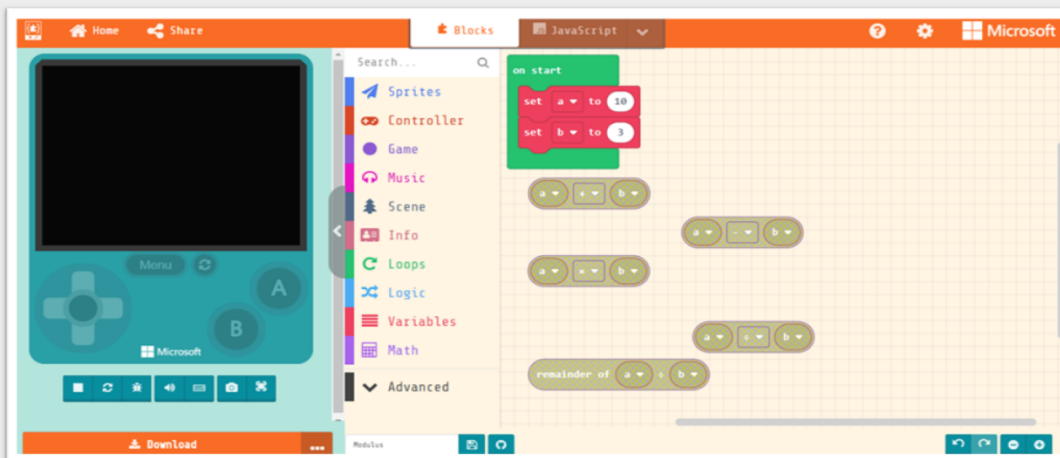
Step 2: Make a Variable “a” and set it value to “10”. Similarly, create another variable “b” and set its value to “3” as shown in the image



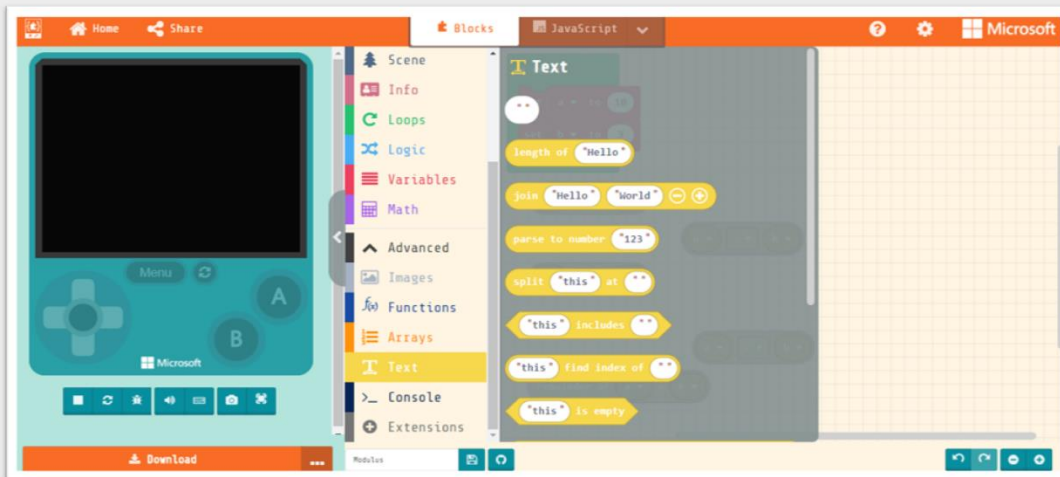
Step 3: Now Drag and drop blocks of both the variables in “on start” block



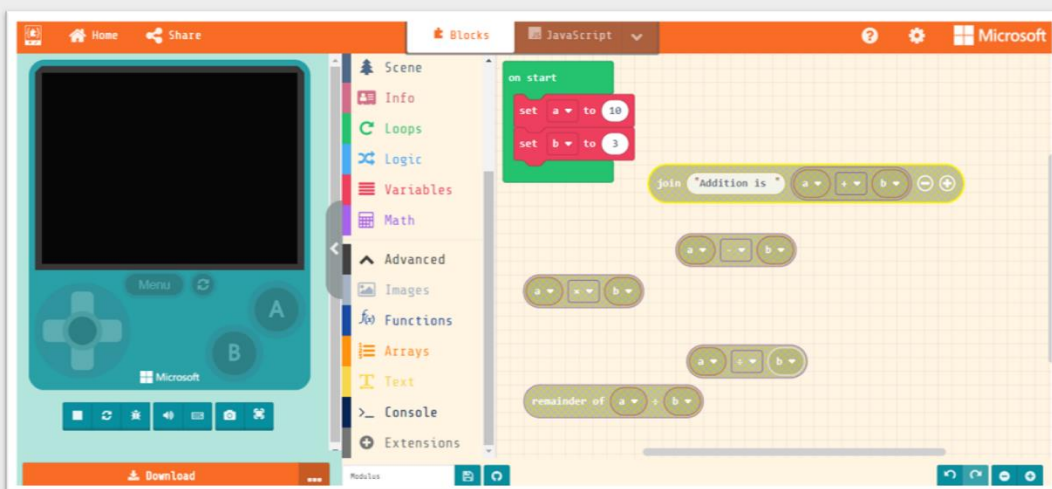
Step 4: Click on “Math” link from the Toolbox and drag and drop “Addition”, “Subtraction”, “Multiplication”, “Division” and “Remainder” Blocks in the Play Area



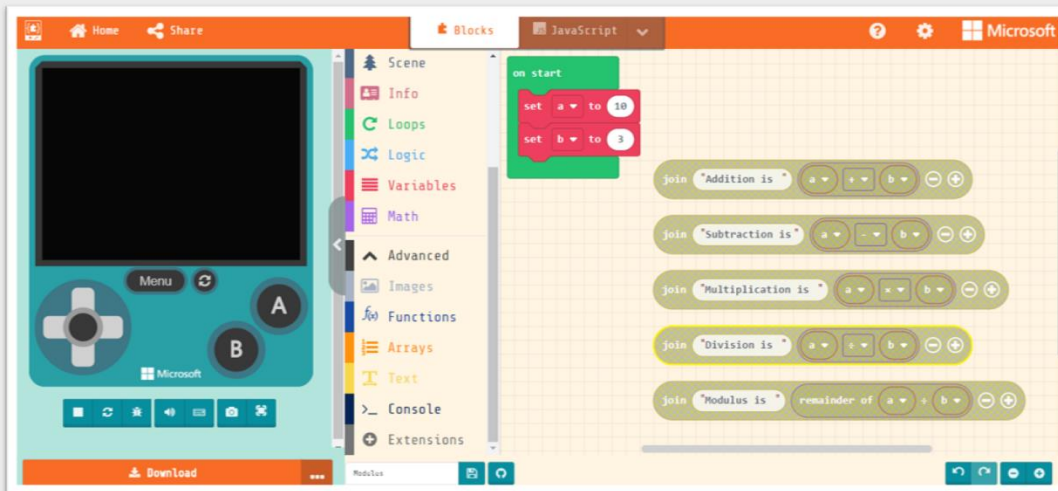
Step 5: Now, fix variables “a” and “b” in the mathematical blocks of the play area



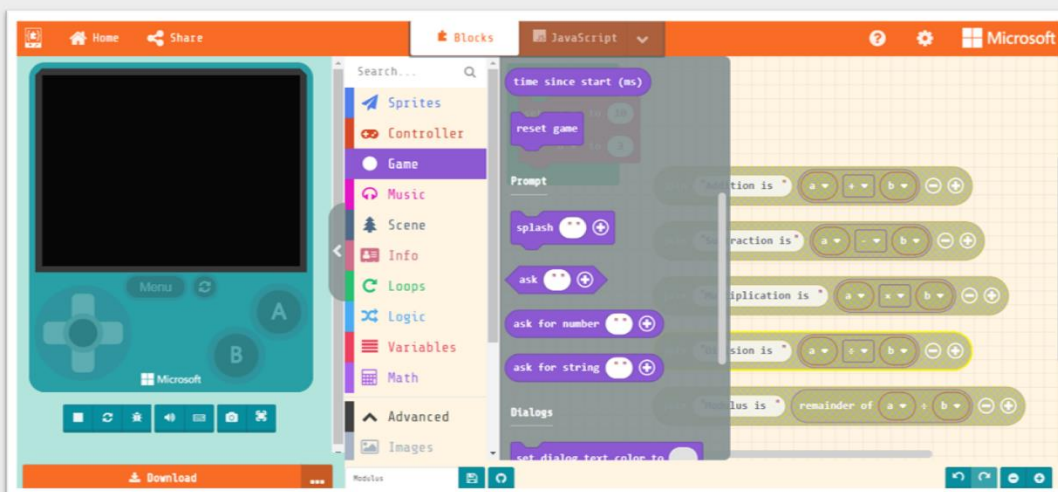
Step 6: Click on “Text” link from the Toolbox and drag and drop “Join” block to the Play area



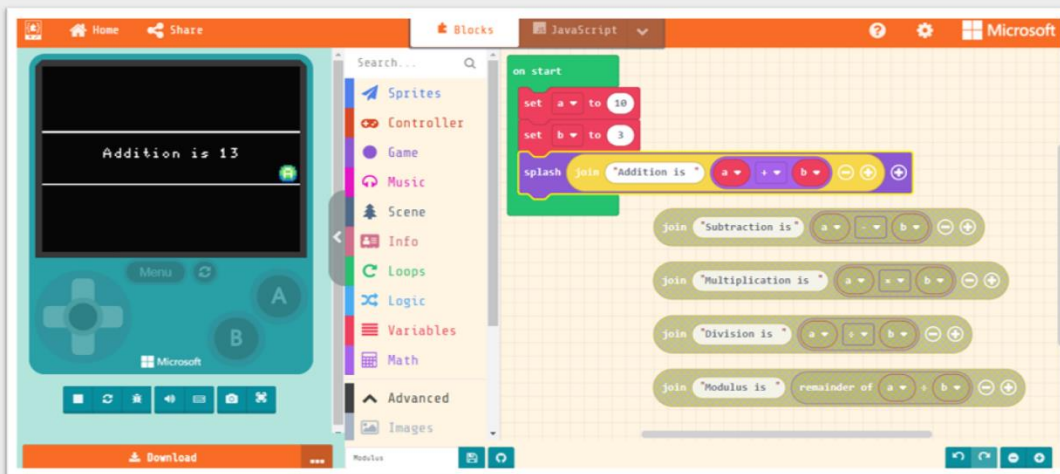
Step 7: In the “Join” block, rename first text box to “Addition is” and attach addition block in the second text box as shown in the image



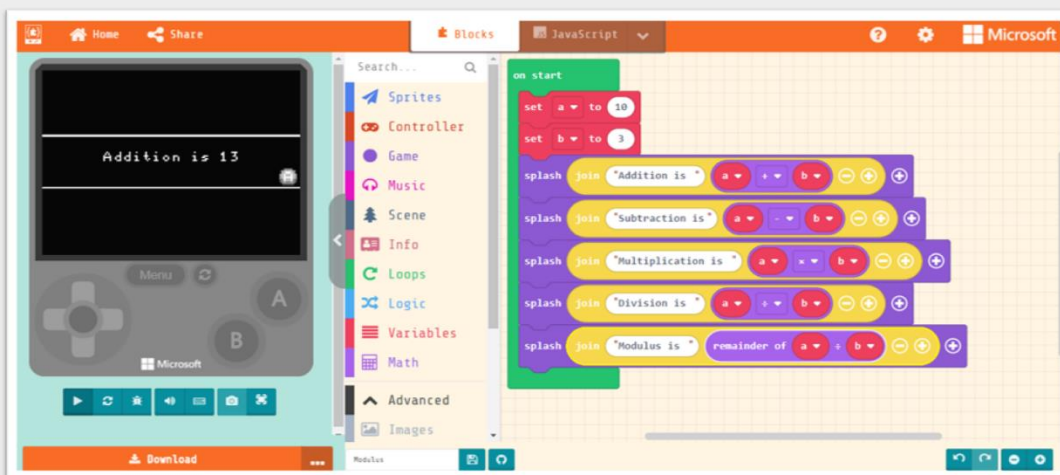
Step 8: Repeat Step 7 for other mathematical operations in the play area



Step 9: Click on “Game” link from the Toolbox and drag and drop “Splash” block in the play area



Step 10: Now attach Addition block in the Squash block. Later, fix this block to the “On Start” block as shown in the image



Step 11: Repeat Step 10 for rest of the mathematical operations as shown in the image



REFERENCES

1. Microsoft MakeCode Arcade. 2021. Microsoft MakeCode Arcade. [ONLINE] Available at: <https://arcade.makecode.com>. [Accessed 23 February 2021]
2. Microsoft MakeCode for Minecraft. 2021. Microsoft MakeCode for Minecraft. [ONLINE] Available at: <https://minecraft.makecode.com>. [Accessed 23 February 2021]
3. Microsoft MakeCode. 2021. Activity: We Built a Zoo. [ONLINE] Available at: <https://minecraft.makecode.com/courses/csintro/arrays/activity-1>. [Accessed 23 February 2021]
4. Association for computing machinery (acm). 2016. CSPathshala. [Online]. [16 March 2021]. Available from: <https://cspathshala.org>