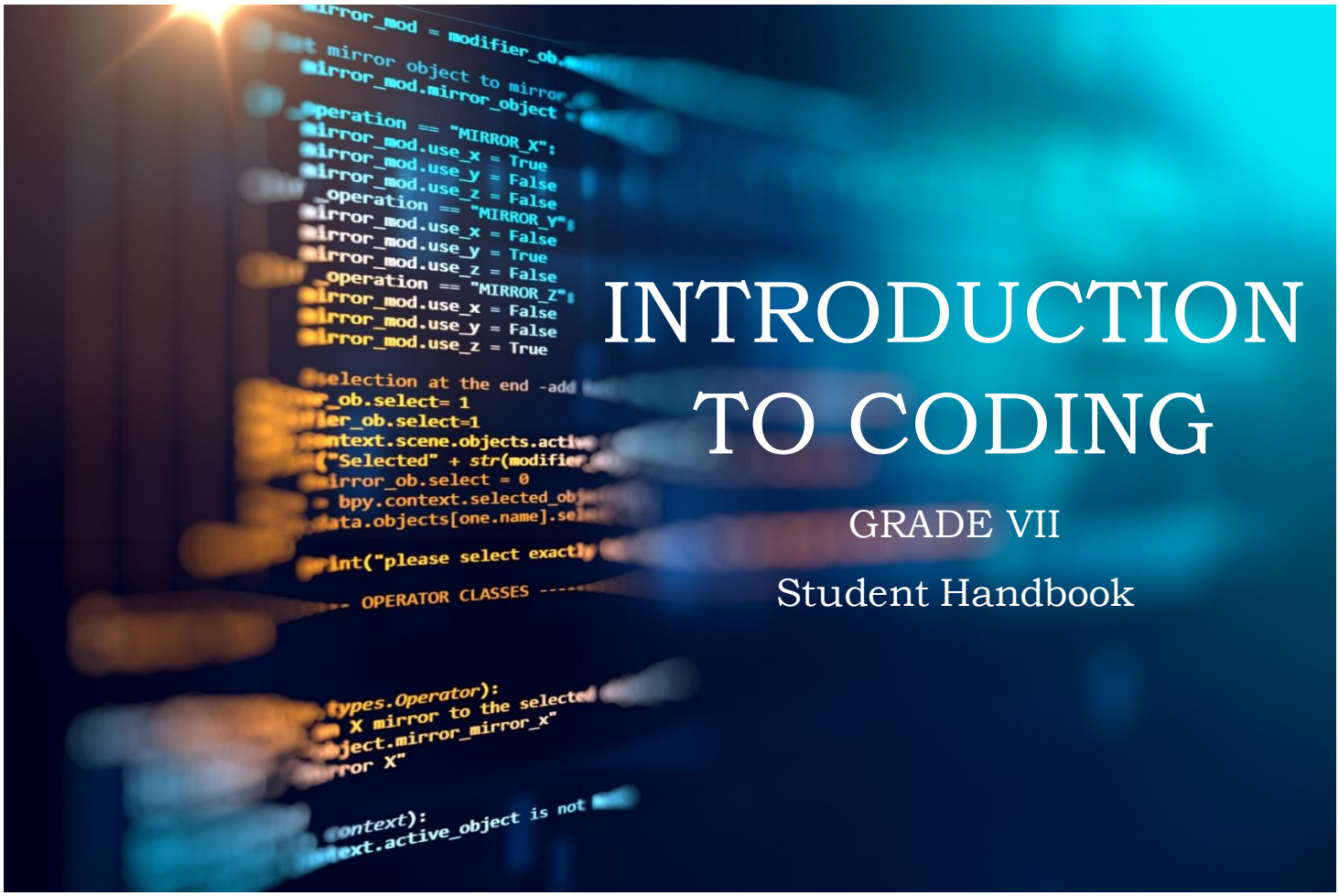


Version 1.0

Microsoft

INTRODUCTION TO CODING

GRADE VII

Student Handbook

ACKNOWLEDGEMENT

Patrons

- Sh. Ramesh Pokhriyal 'Nishank', Minister of Human Resource Development, Government of India
- Sh. Dhotre Sanjay Shamrao, Minister of State for Human Resource Development, Government of India
- Ms. Anita Karwal, IAS, Secretary, Department of School Education and Literacy, Ministry Human Resource Development, Government of India Advisory

Editorial and Creative Inputs

- Mr. Manuj Ahuja, IAS, Chairperson, Central Board of Secondary Education

Guidance and Support

- Dr. Biswajit Saha, Director (Skill Education & Training), Central Board of Secondary Education
- Dr. Joseph Emmanuel, Director (Academics), Central Board of Secondary Education
- Sh. Navtez Bal, Executive Director, Public Sector, Microsoft Corporation India Pvt. Ltd.
- Sh. Omjiwan Gupta, Director Education, Microsoft Corporation India Pvt. Ltd.
- Dr. Vinnie Jauhari, Director Education Advocacy, Microsoft Corporation India Pvt. Ltd.
- Ms. Navdeep Kaur Kular, Education Program Manager, Allegis Services India

Value adder, Curator and Co-Ordinator

- Sh. Ravinder Pal Singh, Joint Secretary, Department of Skill Education, Central Board of Secondary Education

ABOUT THE HANDBOOK

Coding is a creative activity that students from any discipline can engage in. It helps to build computational thinking, develop problem solving skills, improve critical thinking and exposure to real life situations to solve problems in various realms.

Therefore, CBSE is introducing 'Coding' as a skill module of 12 hours duration in classes VI-VIII from the Session 2021-2022 onwards. The idea is also to simplify the coding learning experience by nurturing design thinking, logical flow of ideas and apply this across the disciplines. The foundations laid in the early years will help the students to build the competencies in the area of AI, data sciences and other disciplines.

CBSE acknowledges the initiative by Microsoft India in developing this coding handbook for class VII students. This handbook uses block coding to explain concepts of coding and introduces python in MakeCode platform. It uses gamified learning approach to make learning experience more engaging. The book is intuitive with practical examples of theoretical concepts and applied exercises. There are mini projects that students can work on. Additionally, the handbook also focuses on creating exposure to ethics of coding and application of coding in other subjects like mathematics.

The purpose of the book is to enable the future workforce to acquire coding skills early in their educational phase and build a solid foundation to be industry ready.

RESOURCES FOR STUDENTS

Minecraft education edition

Minecraft education edition is a game-based learning platform that promotes creativity, collaboration, and problem-solving in an immersive digital environment. This platform provides a fun way of learning coding and design thinking concepts.
Visit https://education.minecraft.net/ for more details.

MakeCode

Microsoft MakeCode is a free, open source platform for creating engaging computer science learning experiences that support a progression path into real-world programming. It brings programming to life for all students with fun projects, immediate results, and includes both block and text editors for learners at different levels.
Visit https://www.microsoft.com/en-us/makecode for more details.

GitHub

GitHub is a storehouse where you can manage and collaborate on your code. It helps to maintain different versions of the code easily. GitHub Student Developer Pack gives students free access to the best developer, web development, gaming and many other tools at no cost enabling practical learning.
Sign up for the GitHub Student developer pack here
https://education.github.com/discount_requests/student_application?utm_source=2021-06-11-cbse

TABLE OF CONTENTS

ETHICAL PRACTICES IN CODING

Let us start this course by getting familiar with some fundamental principles of ethical coding.

Be honest and trustworthy

2. As you keep building software of your own, ensure you provide full disclosure of all software capabilities, constraints, and possible problems
3. Do not provide misleading claims about the capabilities of the software
4. Always be honest about your qualifications and competence to perform a task.
5. Try your best to adhere to commitments you have made

Give proper credit to another person's work

6. As you grow as a software developer, you will be utilizing code developed by other persons as well. It is essential to provide disclosure to all such work. Never take credit for another person's work
7. Do not claim ownership of work that others have shared as public resources

With these principles in mind let us get started with this book.

VARIABLES IN REAL LIFE

1.1 What will you learn in this chapter?

This chapter aims at teaching students how variables can be used in real life. We will learn how to initialize variables in programming, how to validate user input and performing mathematical operations on different data types.

1.2 What is Variable Initialization?

To use variables in programming, we need to create the variables and assign them a reference in computer memory. Once we create a variable, we also need to assign it a value before it is used at all. This process of assigning a value to variable is called Initialization.

A variable that is not initialized ever has an undefined value. Such a variable cannot be used until it is assigned with a certain value. In a case where a variable is declared (read – created) but it is not initialized, we can assign it a value in later steps using assignment operators.

In programming, a variable can be initialized using the below syntax:

```
Datatype VariableName = Value;
```

You cannot take any action on any variable till the time you initialize it. However, you can initialize the value of a variable to null. Do not confuse the value of null to variable that is not initialized. "Null" is still a value, a variable that is not initialized does not hold any value, not even null.

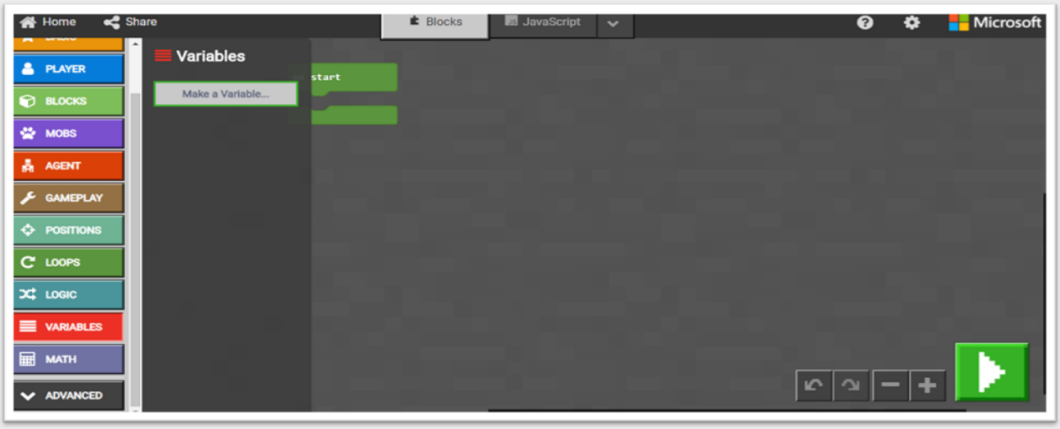
Note: In some programming languages like python, there is no command to declare variables. A variable is created the moment you first assign a value to it.

Example of declaring variables in python

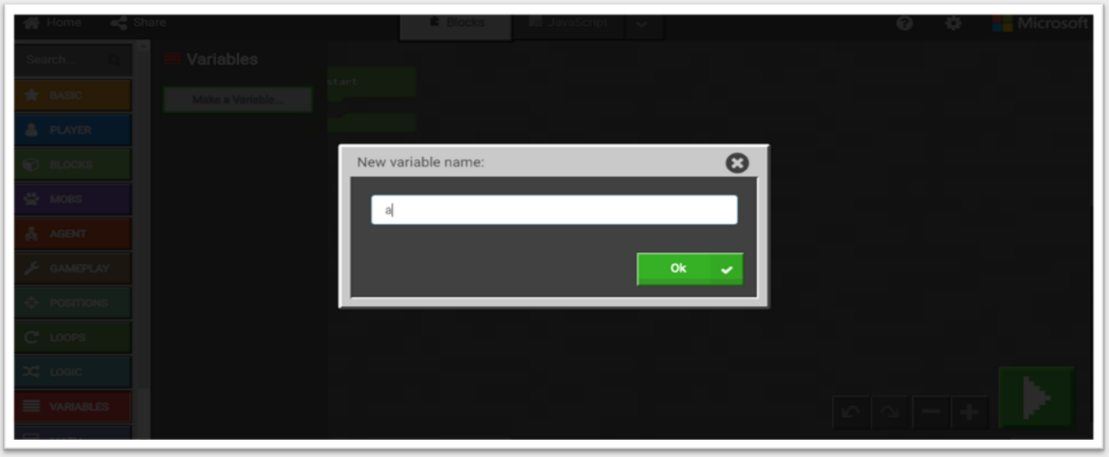
a = 5

b = "Hello"

Let us now do a practice exercise in **Minecraft: Education Edition** to understand how variable initialization works in programming. You setup Minecraft education edition from https://education.minecraft.net/get-started.

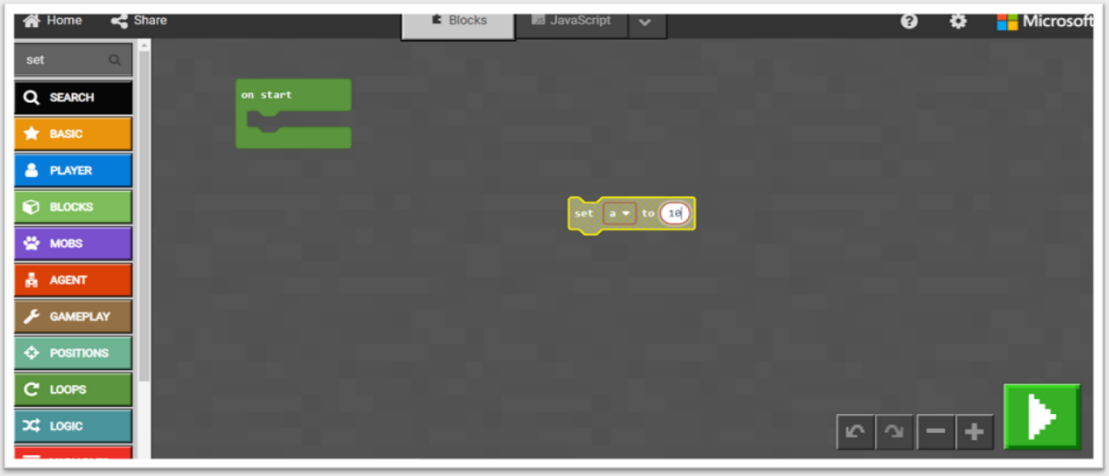


Step 1: Open Minecraft Editor, click on the "**Variables**" link from the toolbox and the click on "**Make a Variable**"

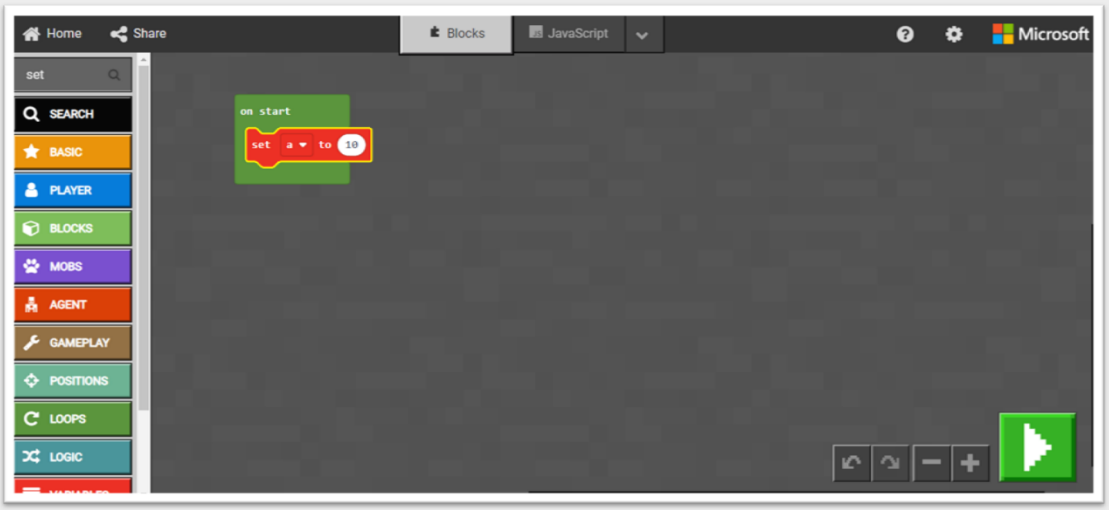


Step 2: Write "a" in the "**New Variable Name**" text box that appears on screen and click on green "**Ok**" button on the bottom of the page.

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.

Note: Minecraft 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 (https://code.org/)

1.3 Data Types in programming

Depending on the requirement, programming languages offer wide variety of data types to suit the programmers' needs.

Below are some basic data types available in Python:

1. Integer
2. Floating-Point Number
3. String
4. Boolean

Integer

Integer is a data type used to define integer variables. Integers can be of any length. They can be any positive or negative whole number.

Floating-Point Number

A floating-point number in Python is used to define decimal numbers and is accurate up to 15 decimals places.

String

A string in Python is a sequence of Unicode characters. It is used to define any form of strings ex. Text, Alphanumeric etc. In Python, we can use single quotes, double quotes or triple quotes to define strings. Multi-line strings in Python can be represented using triple quotes.

Boolean

Boolean data in Python is used to define Boolean variables. Boolean variables can hold only two values in it, either "true" or "false".

1.4 How do we validate user input in programming?

An important part of making sure that your program works fine is to validate that the user enters valid values in the places where you are expecting an input from the user. For example, consider someone asking you a question, "What did you have for lunch today?". To answer this question, you should answer the person what food item you had for lunch. Imagine you answering back "I have pink box in my hand". This is obviously, not an acceptable answer to the question that was asked and may confuse the person who asked the question.

Similarly, in programming, whenever you create a variable with a specific data type, and you are expecting a user to enter a certain value in that data type, you should make sure that the user enters the right type of value, which won't cause any error during program execution. This validation of Input plays a vital role while writing a program.

A computer program behaves exactly the way how a flowchart flow. Any deviation of data type in the user's input will result in an error by program.

A very common way of validating user input in flagging out the wrong input. To do this, common practice is to set a flag in program which has a default value "false". If the user input matches the

expected input, you will reset the flag to true and execute. If the user value is set to false, you will abort the execution by throwing an error message.

1.5 Math Operations in Programming

Now that we have understood how to declare and initiate a variable and how to validate the user input let us now understand how we can perform mathematical operations of these variables.

In programming, there are following operations that you can perform on variables.

Addition

Addition operation is used to perform mathematical addition of two variables. In programming, we refer to "+" as a symbol of addition. Please note that Addition can only be carried out on Integer, Float, Double and String datatypes only. You can use this operator to concatenate two strings on either side of the operator.

For example, if there are two float variables, "x" and "y" declared in your program. Where x = 1.1, and y = 2.2. If we perform addition operation these variables in programming, the result is going to be 3.3

Subtraction

Subtraction operation is used to perform mathematical subtraction of two variables. In programming, we refer to "-" as a symbol of subtraction. Please note that subtraction can only be carried out on Integer, Float, Double datatypes only.

For example, if there are two integer variables "a" and "b" declared in your program. Where a = 3, and b = 2. If we perform the subtraction operation of these variables in programming, the result is going to be 1.

Multiplication

Multiplication operation is used to perform the mathematical multiplication of two variables. In programming, we refer to "*" as a symbol of multiplication. Please note that multiplication can only be carried out on Integer, Float and Double datatypes.

Division

Division operation is used to perform the mathematical division of two variables. In programming, we refer to "/" as a symbol of division. Please note that Division can only be carried out on Integer, Float and Double datatypes.

Modulus

Modulus operation is used to perform the mathematical remainder of two variables. In programming, we refer to "%" as a symbol of modulus. Modulus operator divides variable on the left to the variable on right and returns the remainder. Please note that modulus can only be carried out on integer and float datatypes in Python. For example, if there are two integer variables "x" and "y" declared in your program. Where x = 10, and y = 3. If we perform modulus operation on these variables in programming, the result is going to be 1.

1.6 Quiz Time

Objective Type Questions

Question 1	Process of assigning value to a variable before it is being used is called:
Option 1	Initialization
Option 2	Operation
Option 3	Realization

Question 2	int k1, k2 = 23 is an example of:
Option 1	Variable declaration
Option 2	Adding values in a function
Option 3	None of them

Question 3	Which of the following is NOT a correct variable name?
Option 1	2bad
Option 2	Zero
Option 3	theLastValueButOne
Option 4	Year2000

Question 4	Which of the following declarations is NOT correct?
Option 1	Double duty;
Option 2	Float loan = 12.3;
Option 3	Boolean value = 21;
Option 4	Int start = 45, end = 99;

Question 5	Which of the following shows the syntax of an assignment statement?
Option 1	VariableName = expression;
Option 2	Expression = expression;
Option 3	Expression = VariableName;
Option 4	datatype = VariableName;

Question 6	What is an expression?
Option 1	The same thing as a statement
Option 2	An expression is a list of statements that make up a program
Option 3	An expression is a combination of literals, operators, variables, and parentheses used to calculate a value
Option 4	An expression is a number expressed in digits.

Question 7	What is the value of the following expression? (2 - 6) / 2 + 9
Option 1	7
Option 2	8
Option 3	9
Option 4	10

Standard Questions

1. What is variable initialization?
2. What is the syntax to initialize a variable in programming?
3. Why do you think validating user input is important in programming?
4. What are your opinions on naming convention of variables declared in programming? Write down five best practices that you think one should follow while declaring variables.

Higher Order Thinking Skills (HOTS)

1. Write a program to check if a String is palindrome or not.
2. Write a program to count the occurrence of the letter “a” in string “We are a family”.

Applied Project

Create a program in Arcade to display square of numbers from 1 to 10

1.7 What have you leant in this chapter?

By now you:

- Should have an understanding about variables
- Should have an understanding about initialization of variables.
- Should know how to apply different mathematical operations on different data types.

SEQUENCING WITH BLOCK CODING

2.1 What will you learn in this Chapter?

1. What is sequencing?
2. Why is it important to follow a sequence in programming?
3. How to reduce steps in a sequence with loops and conditional operators?

2.2 Recap of Loops

1. In programming, repetition of a line or a block of code is also known as iteration.
2. A loop is an algorithm which executes a block of code multiple times till the time a specified condition is met.
3. The break statement modifies the normal flow of execution while it terminates the existing loop and continues execution of the statement following that loop.
4. Whenever a program encounters a continue statement, the control skips the execution of remaining statements inside the loop for the current iteration and jumps to the beginning of the loop for the next iteration.
5. When there is a loop inside another loop, it is called a nested loop.

2.3 What is Sequencing?

An algorithm is a detailed step-by-step process that needs to be followed in order to complete a task or to solve a problem. There are three basic building blocks that can be used when designing algorithms:

- Sequencing
- Selection
- Iteration

These building blocks help us to convert any complex problem into a well-defined solution that can be understood and implemented by others using programming. For example, how would you design an algorithm for your morning routine?

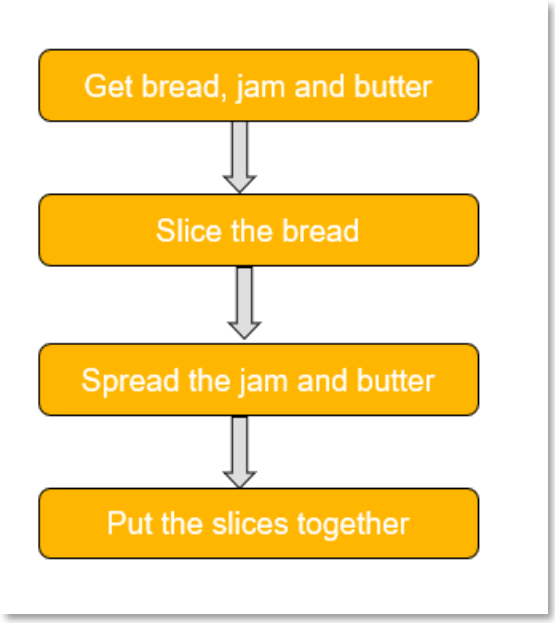
- Wake up
- Brush your teeth
- Take a bath
- Have breakfast
- Go to school

In this chapter, we will learn about the first building block of algorithms – sequencing.

A sequence is a list of activities that are done one after another. Sequencing refers to the specific order in which we need to perform the activities in order to get the desired output.

Designing an algorithm for How to Make a Sandwich

Every day, we do many activities in a sequence. For example, think of how you would make a sandwich. If you had to write down the steps for making a sandwich, would it be like the steps given below?



If you do these steps in any other order, would the result still be a sandwich? No!

Similarly, in programming, tasks need to be done in the correct sequence to get the desired output.

In both daily tasks and coding, if we don't put every step in the right sequence, the result will not be what we wanted. Sequencing is a foundational concept in programming, and everything we learn in the future will build on this concept.

2.4 Examples of Sequence, Selection and Iteration

To understand concept of sequencing in programming, look at the below algorithm to swap two numbers:

Step 1: Start

Step 2: READ num1, num2

Step 3: temp = num1

Step 4: num1 = num2

Step 5: num2 = temp

Step 6: PRINT num1, num2

Step 7: Stop

Over here, to reach the final output, you need to follow the algorithm step by step.

First, Read the two numbers that needs to be swapped. Then, assign value of num1 to a temporary variable called "temp". Next, assign value of num2 to num1. Later, assign value of "temp" to num2. Finally, Print the values of num1 and num2 which are now swapped.

Point to be noted over here is that the steps that followed to achieve this output are in a sequence. Skipping any step from this sequence or altering the sequence will lead to errors in the program or generation of wrong output.

This is how sequencing in programming works.

Now let us understand the concept of Selection using an example.

Look at below pseudocode to check if age entered by the user is of a senior citizen or not. Consider that person is

considered a senior citizen if his age is above 60 years old.

```
int age = 61;

if (age>=60)

        print("Person is a senior citizen");

else

        print("Person is not a senior citizen");
```

Over here, the pseudocode is trying to check if the age defined in the program if of a senior citizen or not.

If you follow the sequence of pseudocode, you will see that the program makes a "selection" of which flow to enter depending on the age defined in the program.

If the age is more than or equal to 60, the program enters the if block. If the age is less than 60, the program enters else block.

This is how concept of selection is applied in programming.

Moving forward, let us have a look at example of Iteration.

In programming, loops follow the iteration depending on the condition. Every loop iterates at least once if not more.

Consider a program to print all the natural numbers from 1 to 100. It will follow the flow as shown in the *Fig 2.1* flowchart.

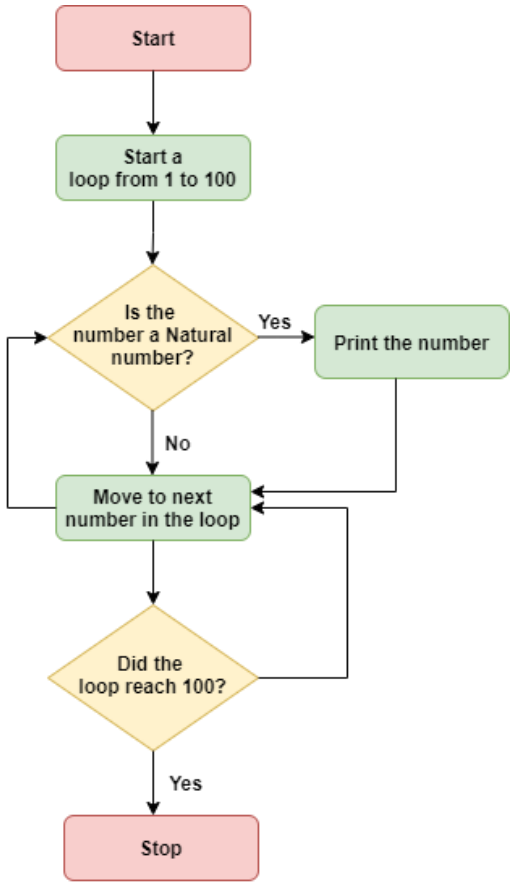


Fig 2.1 Flowchart to print natural numbers from 1 to 100

As you can see in the flow chart the flow of the program will repeat or iterate for each number from 1 to 100. For every single iteration it will check for the condition and take the appropriate workflow. This is how iteration works in programming. It will repeat the block of code for multiple times till the specified condition is achieved.

2.5 What is a Bug?

A bug is a terminology used to describe any unexpected problem in your program. Just like we learnt in the past topics, we follow a sequence of activities to complete a program. This program is expected to return a specific output. Any deviation in the expected and actual output of the program is known as a bug.

For example, suppose you create a boat which is expected to sail in ocean. Now when the boat is ready and you try to sail it in the water, you realize that there is a small hole in the bottom on the boat from where water is seeping in. This water seeping in the boat at a slow speed may create a problem for the boat in the long run. Thus, this hole in the bottom of the boat can be termed as a "bug" in programming.

2.6 Activity Drawing Rectangle

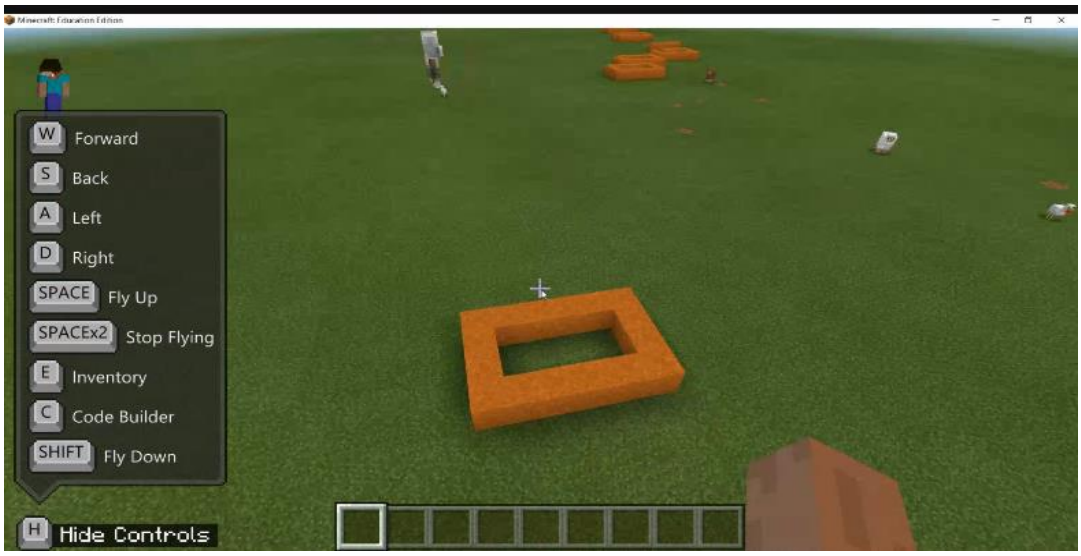
Let us now understand sequencing with another example.

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

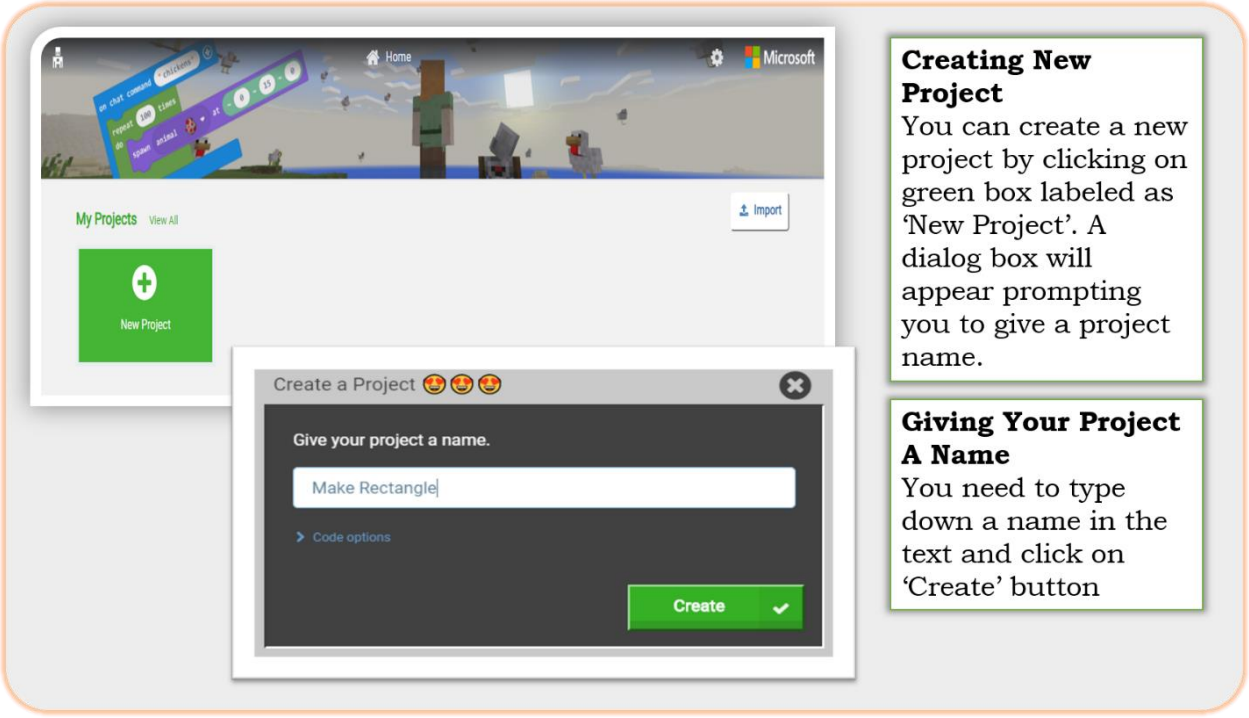
First, draw a line 5 cm in length
Then turn the cursor right by 90 degrees
Then draw a second line 3 cm in length
Then turn right by 90 degrees
Then draw a third line 5 cm in length
Then turn right by 90 degrees
Then draw a fourth line 3 cm in length

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/

Once you complete this exercise, the final output should look like as shown in the screenshot below:

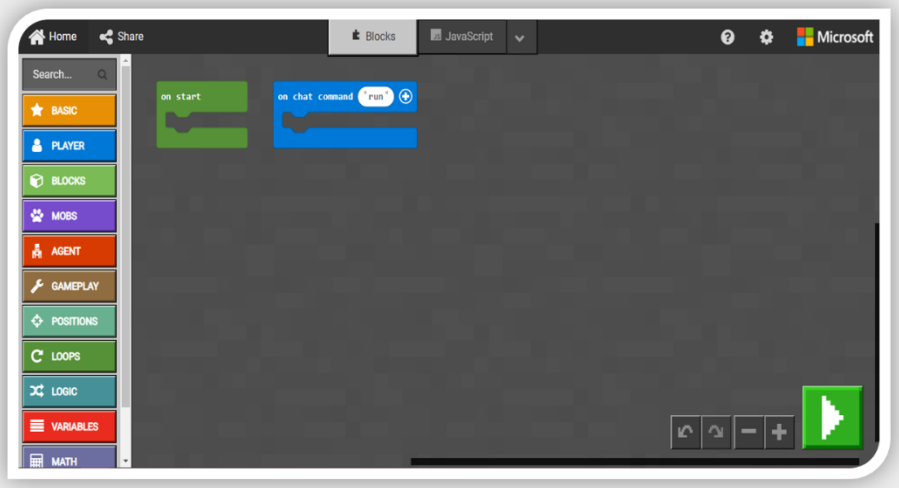


Let us now follow below steps to get this output on our screen:



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.

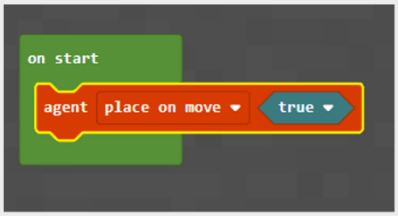
Giving Your Project A Name
You need to type down a name in the text and click on 'Create' button

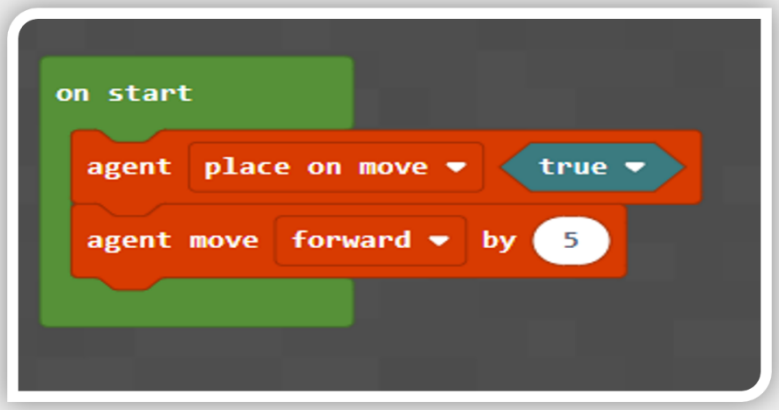


Minecraft Code Editor
Once you create your project, you should see the editor like this.

Now follow the following steps

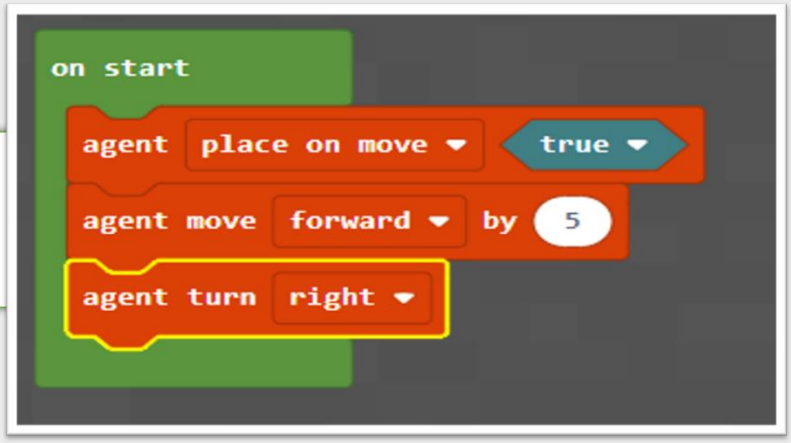
Step 1
From the Agent section, select the "place on move" block as shown below. Set the value to True and add it to the Start block.

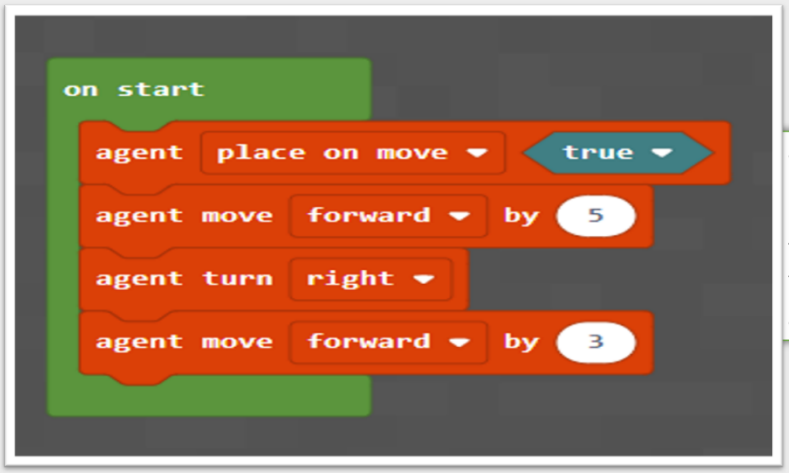




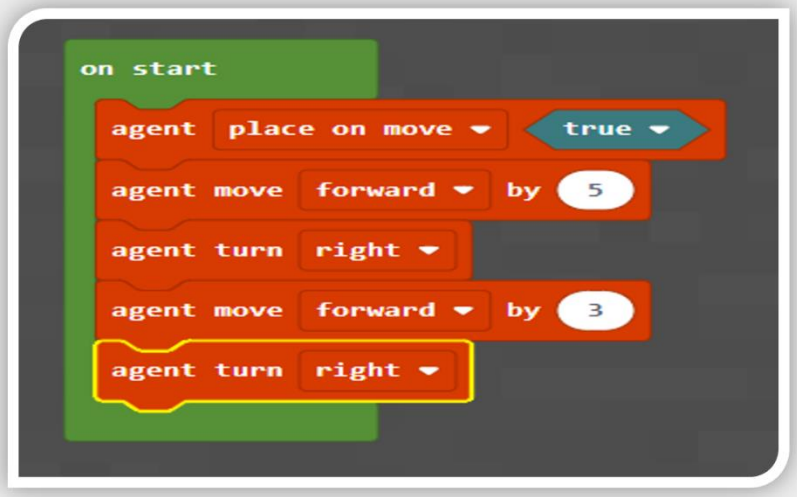
Step 2
From the Agent section, select the "move forward" block, add it to the Start block and set the value to 5 as shown.

Step 3
From the Agent section, select the "turn" block, add it to the Start block and set the value to right



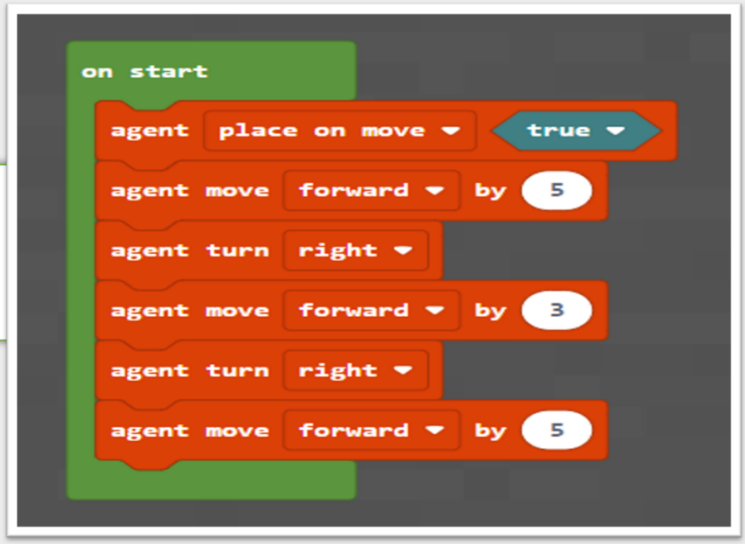


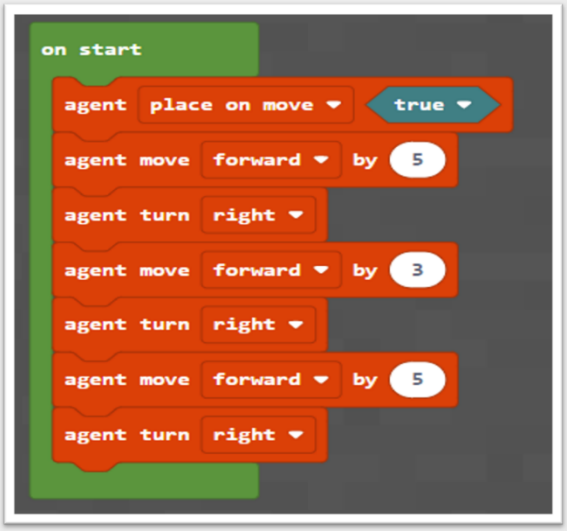
Step 4
From the Agent section, select the "move forward" block, add it to the Start block and set the value to 3 as shown



Step 5
From the Agent section, select the "turn" block, add it to the Start block and set the value to right as shown

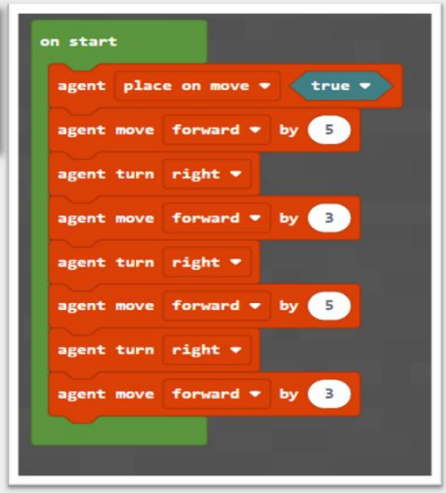
Step 6
You need to repeat Step 2. Placing a block which moves the agent forward by 5





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.

Note: Minecraft 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 (https://code.org/).

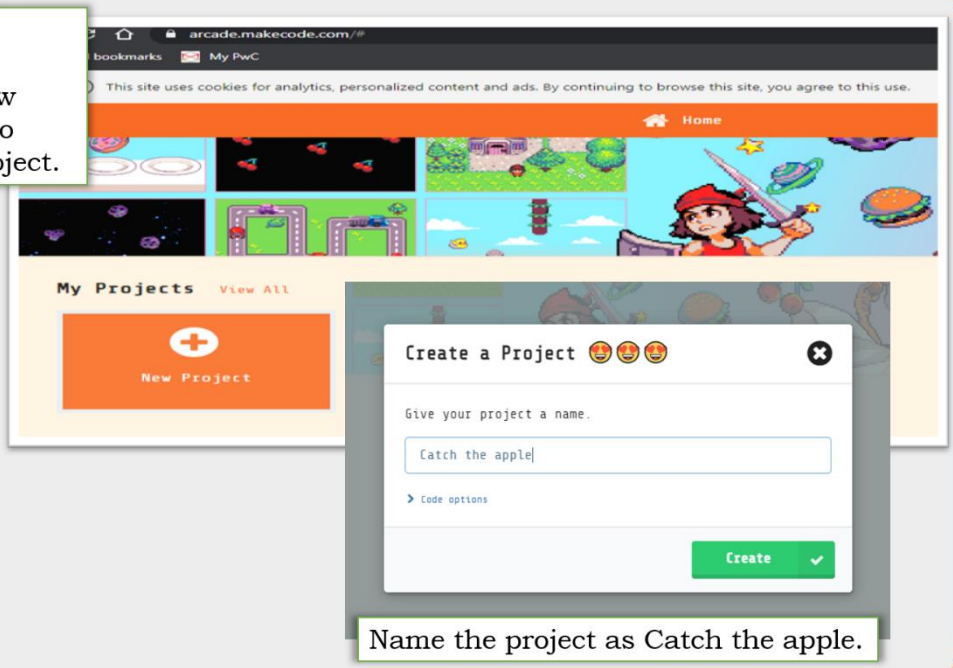
2.7 Fun Activity: Chase the Apple

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.

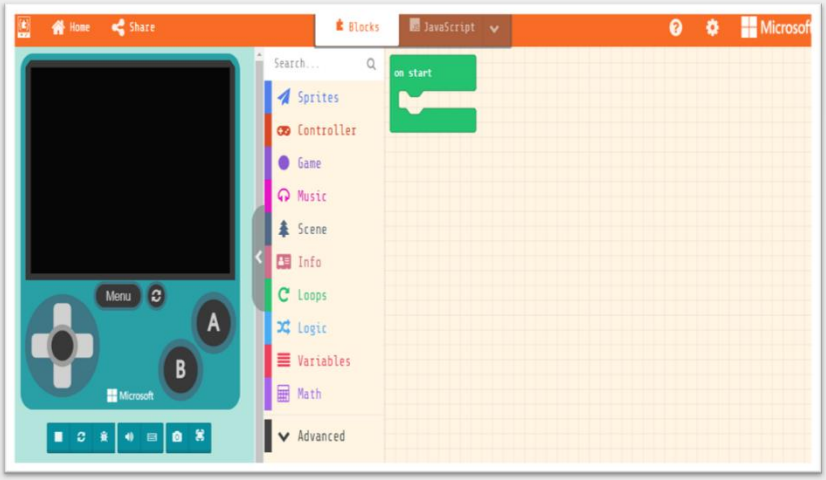
You should try creating this game on Arcade using the MakeCode editor which can be found here https://arcade.makecode.com

First, create a new project as shown below.

Creating New Project
Click on the 'New Project' button to create a new project.

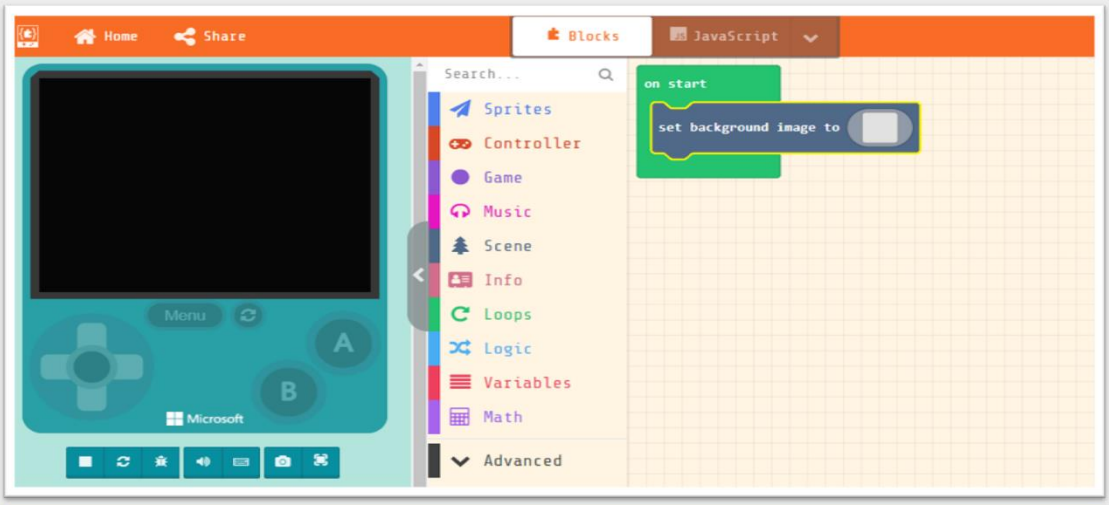


Name the project as Catch the apple.

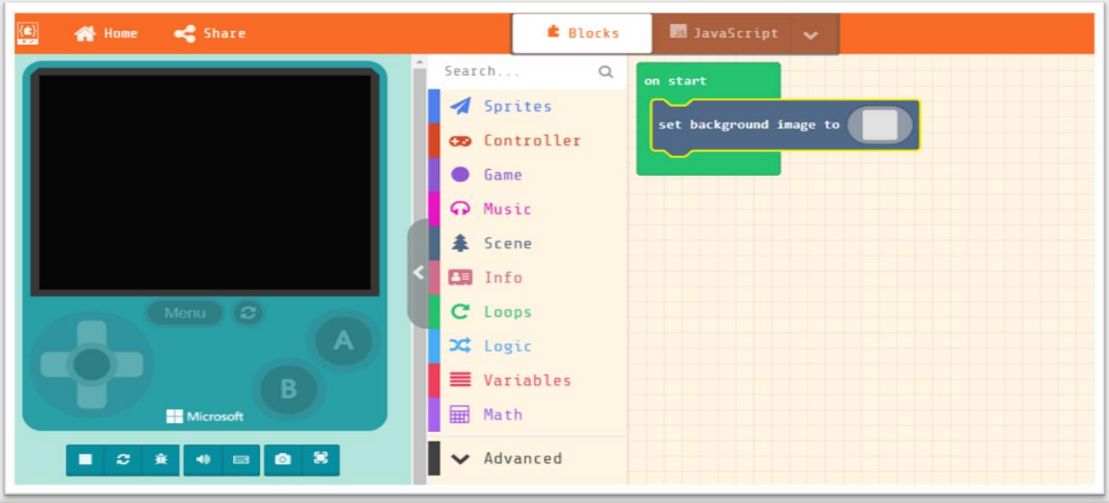


Once the project is created, you should see the editor.

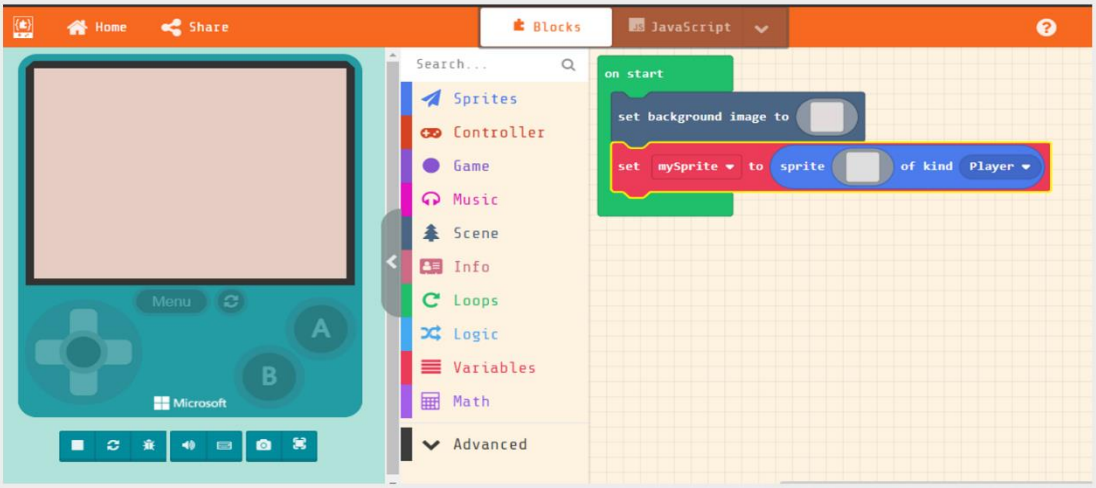
Now follow the steps mentioned below to create the game.



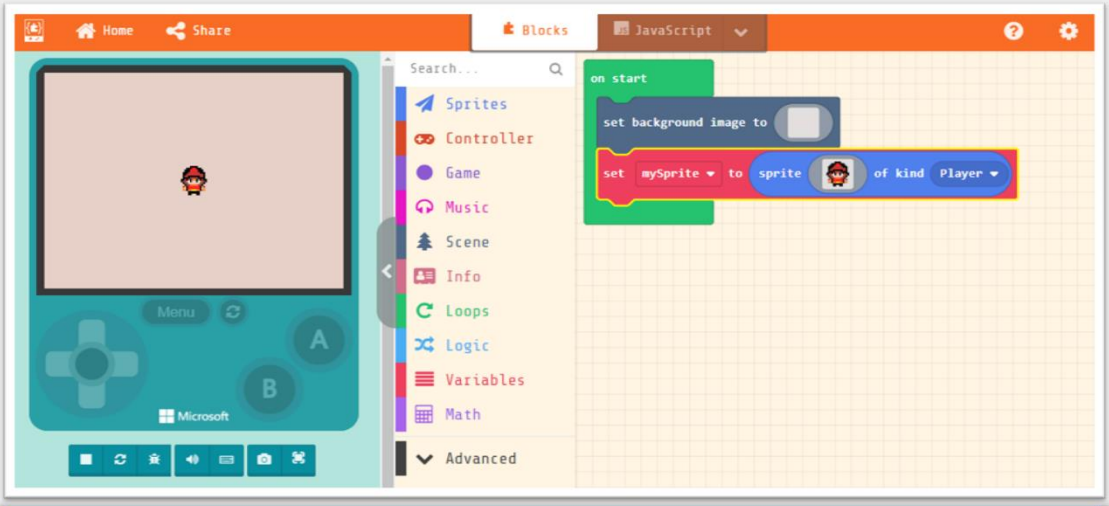
Step1: Open the Scene toolbox drawer and drag the “set background image” block into the “on start” block on your workspace.



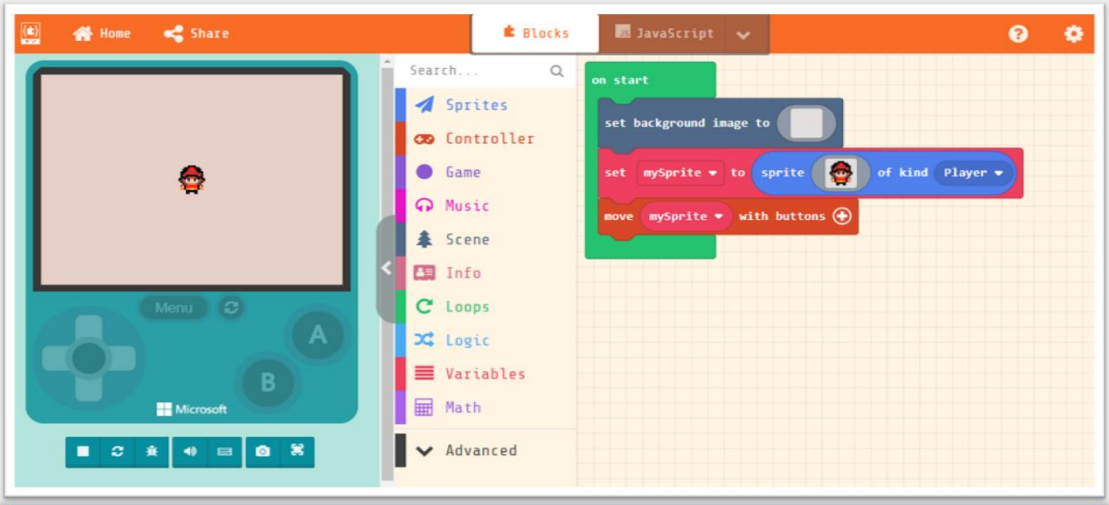
Step 2: In the “set background image” block, click on the gray square to open the image editor, select a color to fill in the background color.



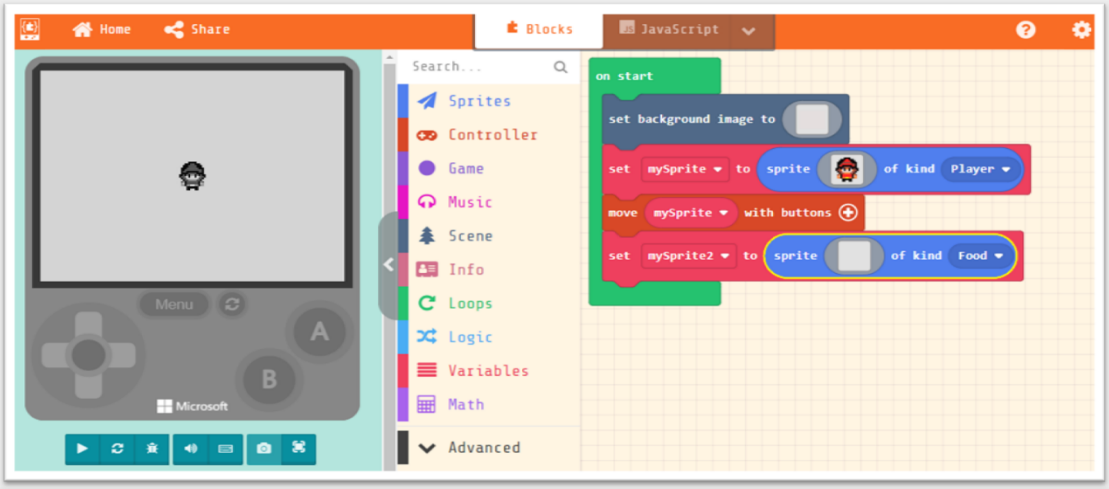
Step 3: Open the Sprites toolbox drawer and drag the first block you see, “set mySprite” into the “on start” block on your Workspace This will create a new Player character for the game.



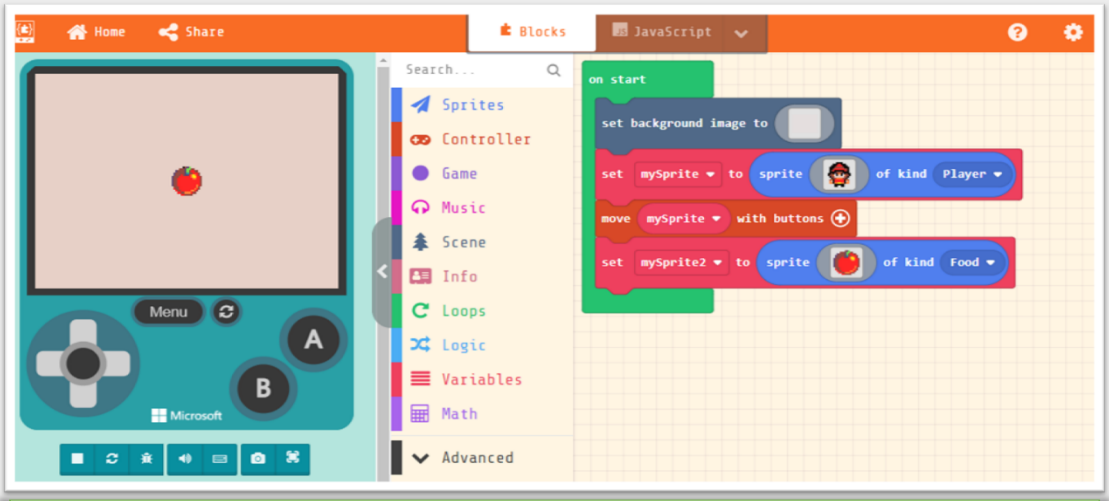
Step 4: In the **Player** block, click on the gray square to open the image editor and select the Gallery view. Find and select a character of your choice.



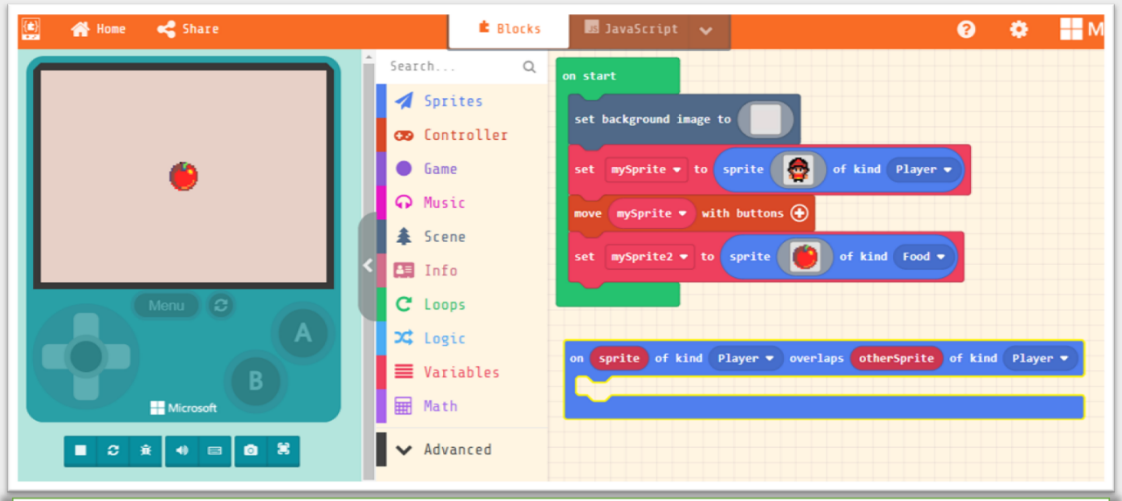
Step 5: Open the Controller toolbox drawer and drag the “move mySprite with buttons” block after the “set mySprite” block. This allows you to move your **Player** sprite around the screen with arrow keys



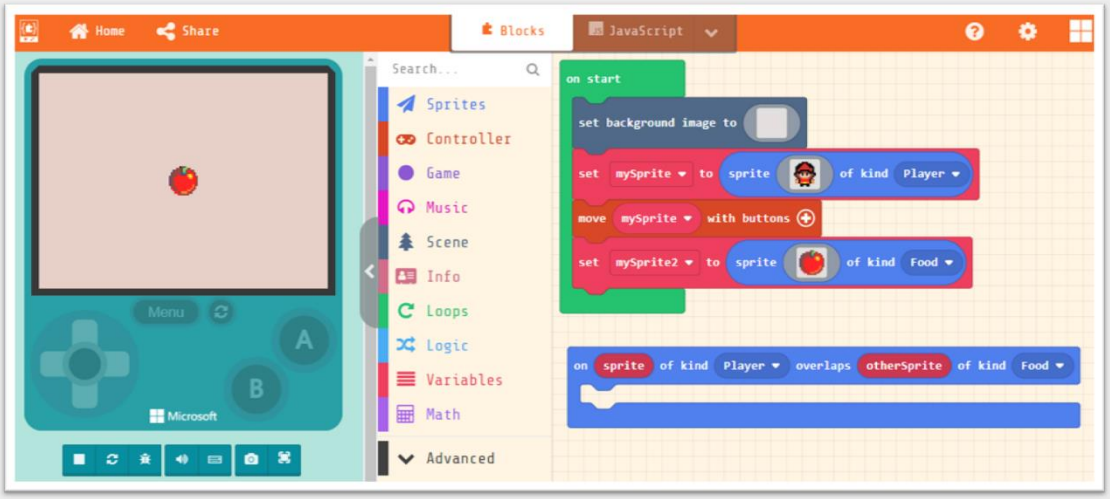
Step 6: Open the Sprites toolbox drawer and drag another “set mySprite2” block into the “on start” block on your workspace. In the kind of Sprite select “Food”.



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.

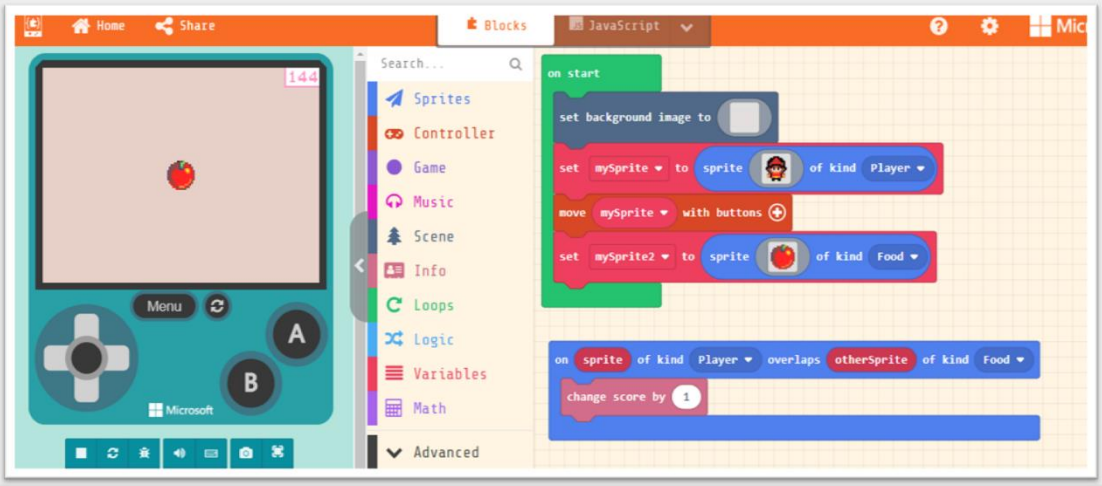


Step 8: Open the Sprites toolbox drawer and drag the “on sprite overlaps otherSprite” block onto your workspace (this can be placed anywhere)

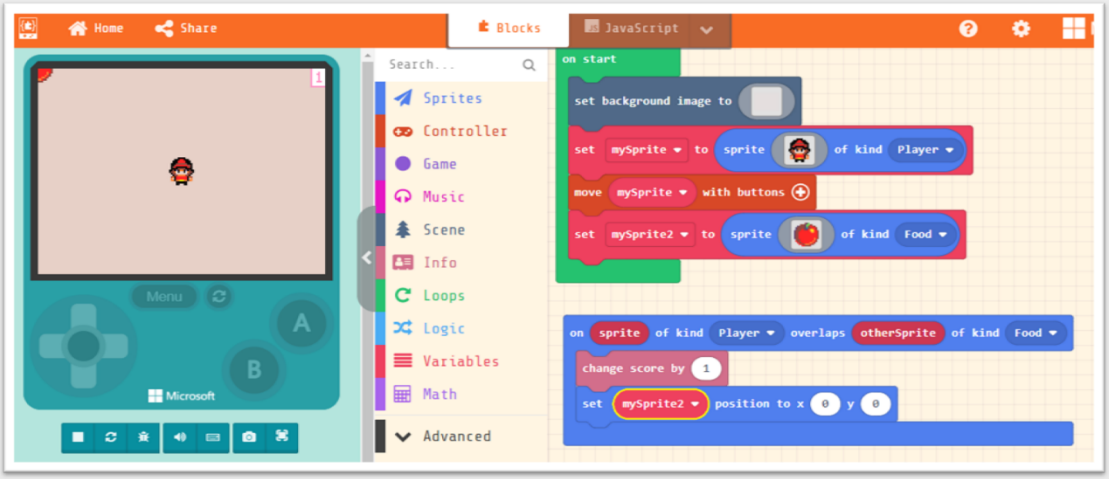


Step 9: In the "on sprite overlaps otherSprite" block, click on the second Player kind after otherSprite to open the menu. Select Food as the kind.

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.

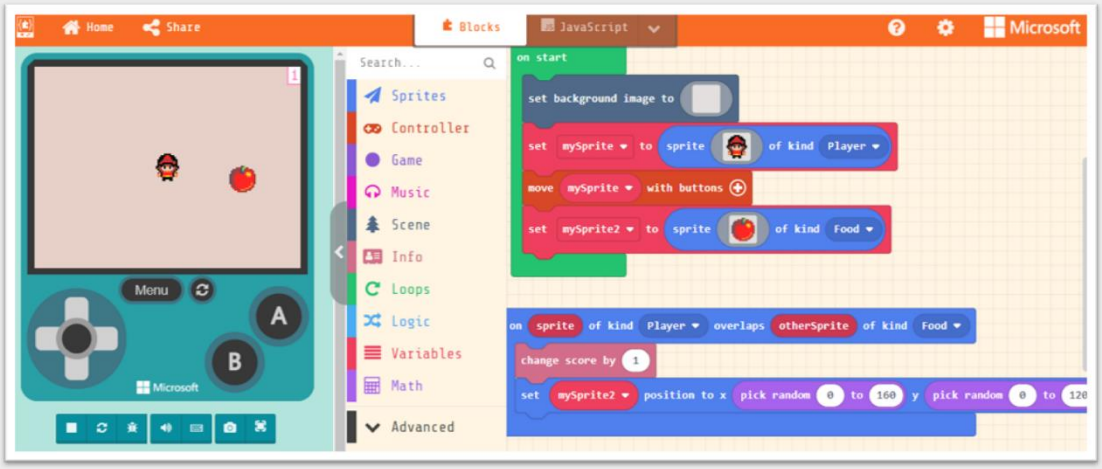


Step 11:Once the sprites overlap, let's set the position for the apple to random locations around the screen. To do so, open the **Sprites** toolbox drawer and drag the "set mySprite position" block into the "**on sprite overlaps otherSprite**" block.

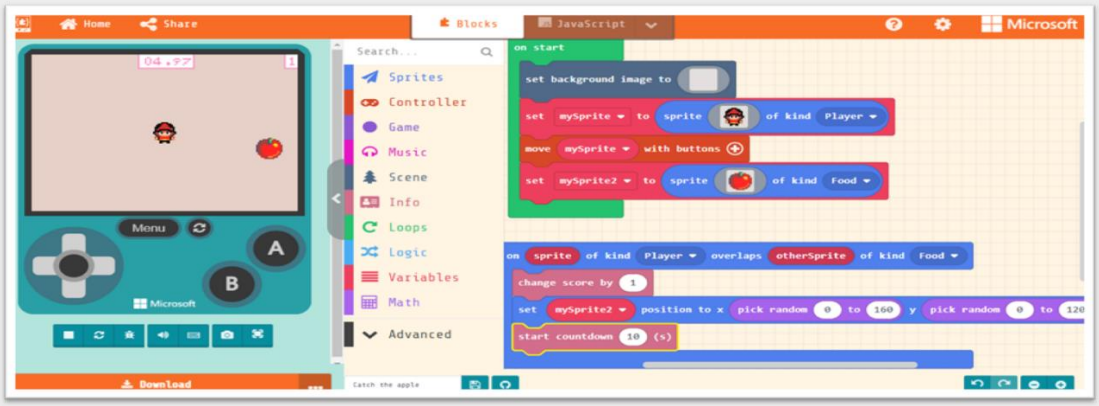


Step 12: In the the "set mySprite position" block into the "**on sprite overlaps otherSprite**" block, click on **mySprite** variable to open the menu, and select your **mySprite2** sprite.

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.

Output for Chase the Apple

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 (https://code.org/)

Activity

- Create a square of size 10 and spawn 2 sheep in it.
- Create two squares such that they have a common side. Spawn 2 chickens in one square and 2 cows

2.8 Types of Loops

As we have studied, use of loops make our code manageable and organized. Mainly, there are three different types of loops:

1. While Loop
2. For Loop
3. Nested Loop

While Loop

The While Loop can execute a set of commands till the condition is true. While Loops are also called conditional loops.

Once the condition is met then the loop is finished.

For Loop

For Loop is needed for iterating over a sequence. A for loop executes for a specific number of times.

Nested Loops

Any loop in program may contain another loop inside it. When there is a loop inside another loop, it is called as a nested loop. How it works is that first success condition of outer loop triggers the inner loop which runs and reaches completion. This combination of loops inside loop is helpful while working with requirements where user wants to work of multiple conditions simultaneously. There is no restriction on how many loops can be nested inside a loop.

2.9 Apply Loops and Conditionals with sequencing

Now, let us discuss the other two important aspects of an algorithm – selection and iteration.

Selection refers to the situation in which the algorithm needs to make a choice between two or more alternatives. It is like asking questions like “Is it raining?”. If the answer to the question is true, the algorithm follows one path. If the answer is false, the algorithm follows a different path.

Iteration refers to the process of doing the same task over and repeatedly, until a certain task is complete, or a certain condition is met. The iteration can also be set to run for a specific number of times. For example, think of a race in which a car must go around a track five times. The car will keep going around the track repeatedly until it completes five laps. Once that is done, it will exit the track.

Now let us see how we can combine all the three building blocks of algorithms – sequencing, selection, and iteration.

In real life, complex algorithms can have hundreds, if not thousands, of steps in a sequence. However, often while working on a sequence, you will notice

that certain activities in the sequence are repeated. We can reduce the number of steps of the sequence by using a loop along with certain conditions to check when the loop should stop.

For example, in the last exercise, while creating a rectangle, we had to turn right after drawing a line and we had to do so three times.

Can we use a loop to reduce the number of steps

2.10 Is there a better way to apply sequencing?

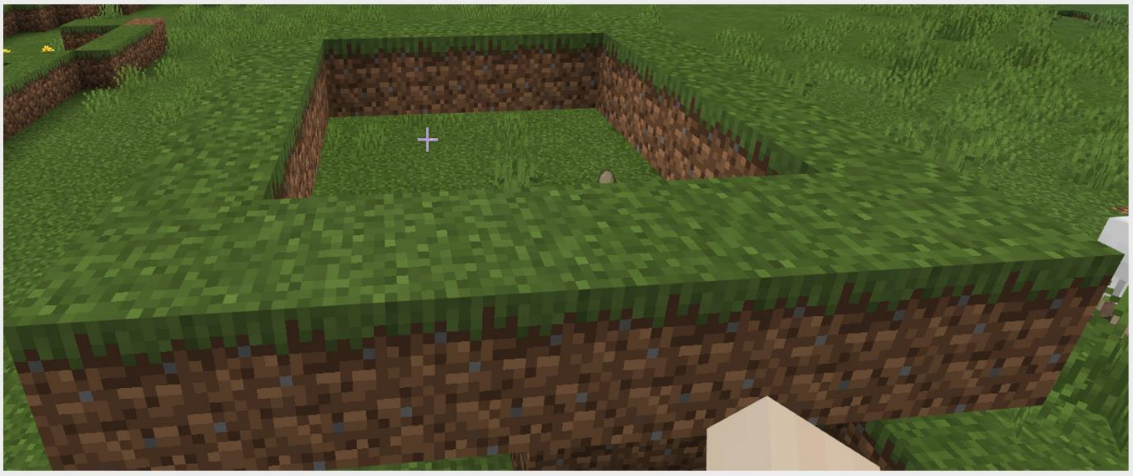
Activity: Better way to create a square

With an example, let us now understand sequencing with loops and conditionals. What would be the steps to draw a square of side 5 cm on your screen?

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.

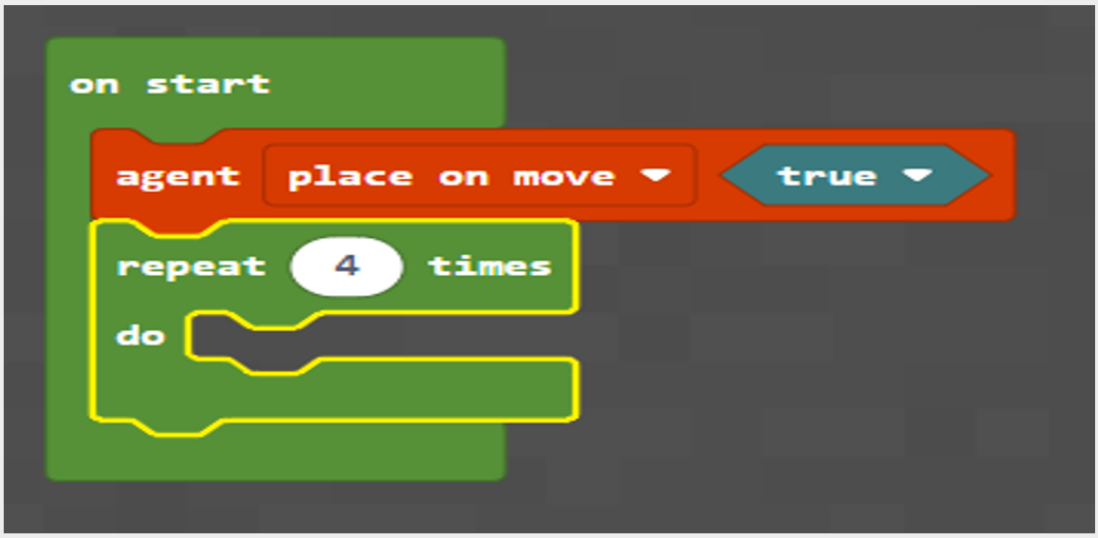
First, create a new project called "Make Square". Once you create your project, you should see the editor below

At the end of this exercise, the final output should look like the one shown in the image below:

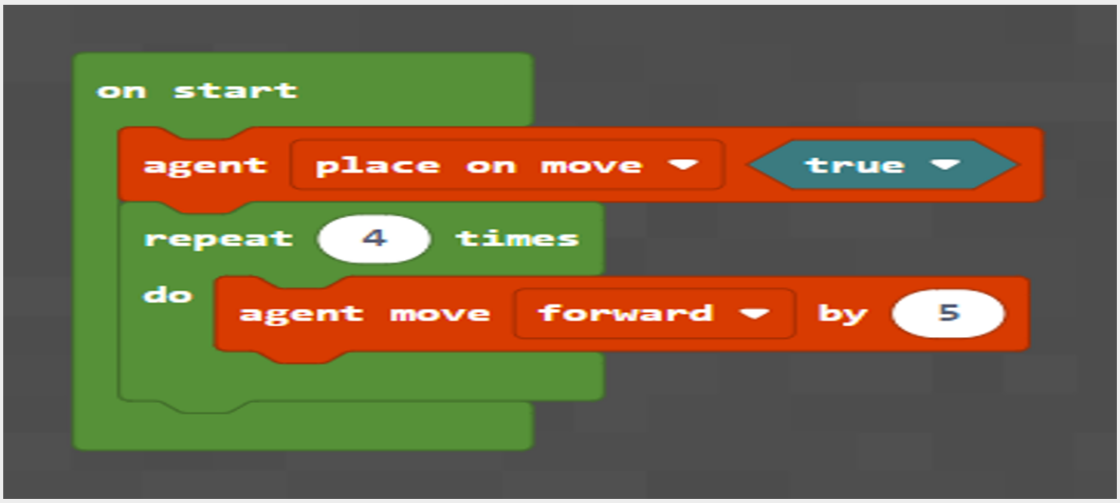


Final output for activity better way to create a square

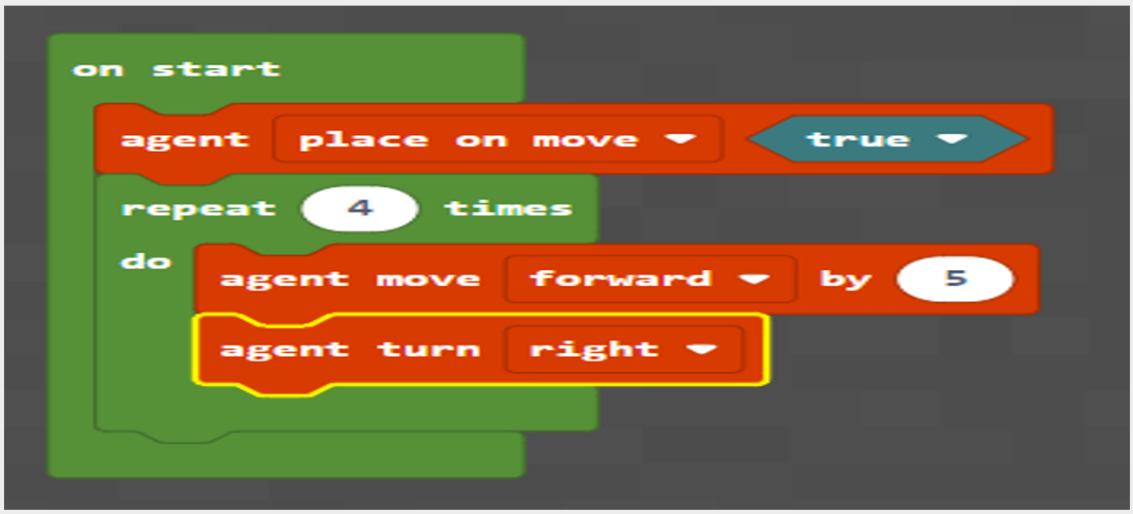
Step 1: From the Agent section, select the "**place on move**" block as shown below. Set the value to True and add it to the **start** block.



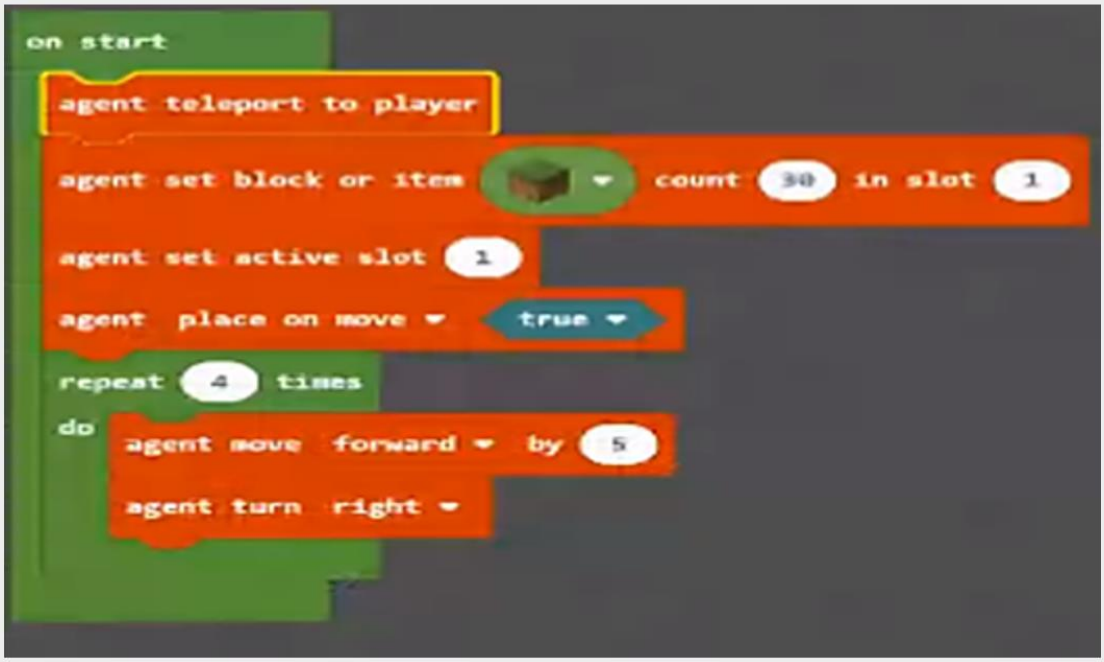
Step 2: From the loops section, select the **repeat** block and add it to the **Start** block as shown below.



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

Note: Minecraft 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 (https://code.org/).

2.11 Activity: Distributing Birthday Sweets

In this activity, we will learn about iteration.

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.

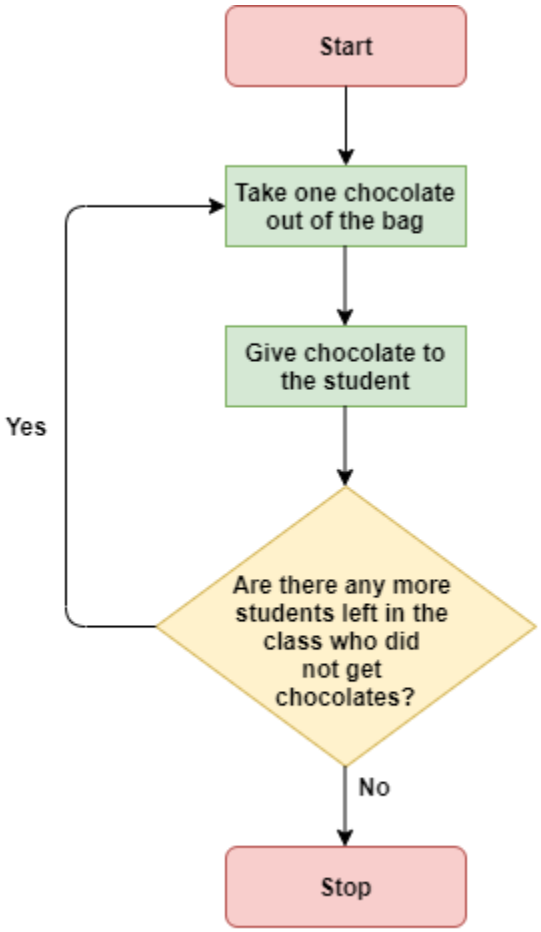


Fig 1.0 Distributing Birthday Sweets

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.

2.12 Quiz Time

Objective Type Questions

Question 1	What is a bug in programming?
Option 1	A feature in a program that causes it to run correctly
Option 2	A feature in a program that can predict output.
Option 3	A feature in a program that generates incorrect output.
Option 4	All the above

Question 2	What is sequencing in programming?
Option 1	A programming structure, where steps are performed in an order.
Option 2	A programming structure, that repeats itself until a condition is met.
Option 3	A feature in a program that generates incorrect output.
Option 4	All the above

Question 3	What is looping in programming?
Option 1	A programming structure, where steps are performed in an order.
Option 2	A programming structure that repeats itself until a condition is met.
Option 3	A feature in a program that generates incorrect output.
Option 4	All the above

Question 4	What is selection in programming?
Option 1	A list of instructions which are followed in a set order
Option 2	A list of instructions where there is a choice based on a condition
Option 3	A list of instructions which will loop based on a condition
Option 4	A list of instructions that will loop forever

Question 5	What is iteration in programming?
Option 1	A list of instructions which are followed in a set order
Option 2	A list of instructions where there is a choice based on a condition
Option 3	A list of instructions which will loop based on a condition
Option 4	A list of instructions that will loop forever

Standard Questions

1. Explain what an algorithm is with the help of an example.
2. Explain a bug in any program/application that you have encountered in real life.
3. Explain if sequencing, selection and iteration can be used together? Support your answer with a proper explanation.
4. Explain three examples where you can apply loop and conditionals to simplify sequencing

Higher Order Thinking Skills (HOTS)

1. Write an algorithm to print cube of numbers from 1 to 10.
2. Draw a flowchart to explain the iterative process that Arun followed while distributing birthday sweets.

Applied Project

Create a rectangle of length 5 and height 3 using a loop (repeat block) in Minecraft.

2.13 What have you learnt in this chapter?

By now you:

- Should have an understanding about the basics of algorithms.
- Should have an understanding about sequencing and its importance.
- Should know how to apply loops and conditions to simplify sequencing.

FUN WITH FUNCTIONS

3.1 What will you learn in this chapter?

At the end of this chapter, you will understand the use of functions in programming. You will know:

- Usefulness of using functions in code.
- How to define and call a function.
- Parameters in a function.
- Different type of values returned by a function.

3.2 What exactly are Functions?

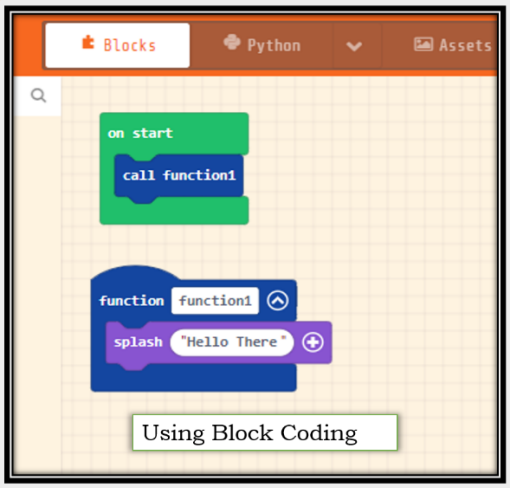
What do you think of when you hear the word pattern? Well, a pattern is one when something or a series of things are repeated in sequence. We all may have seen patterns before, like in color arts, music or even mathematics.

Similarly, there can be patterns in code too. As a programmer, there will be times when you want the computer to repeat the certain lines of code in sequence. If there is a block of code that you want your computer to repeat lots of times, the tool that you may want to use is a function.

A function is a block of code made up of a set of steps that results in a single specific action. The programmer will give this action a simple name. Giving a simple name to a function, increases the chances that the set of steps can easily talked about and reused again and again in the program.

3.3 Examples of Functions in Arcade

Example 1: Calling a function which has no parameters.



Using Block Coding

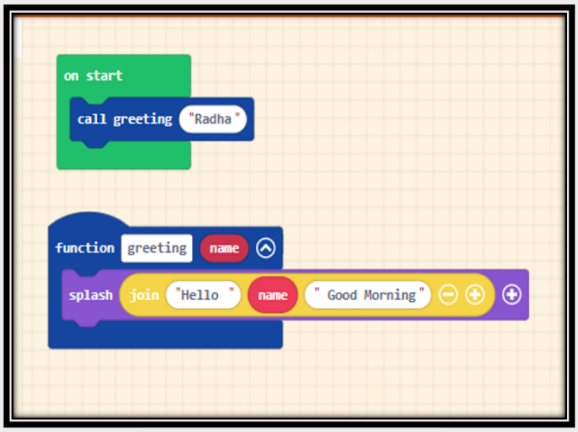
```
1  def function1():
2      game.splash("Hello There")
3
4  function1()
```

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

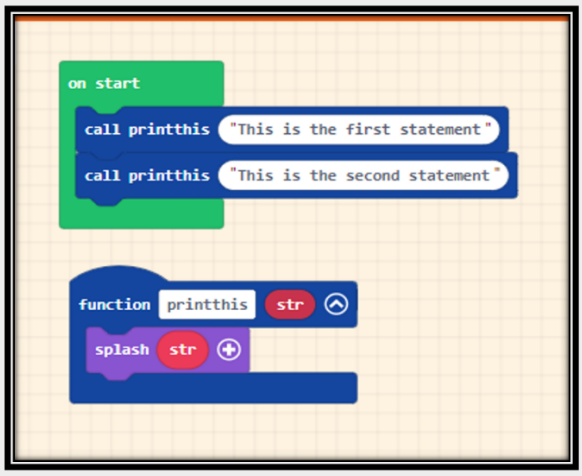
```
1  def greeting(name):
2      game.splash("Hello "+ name + " Good Morning")
3
4  greeting('Radha')
```

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

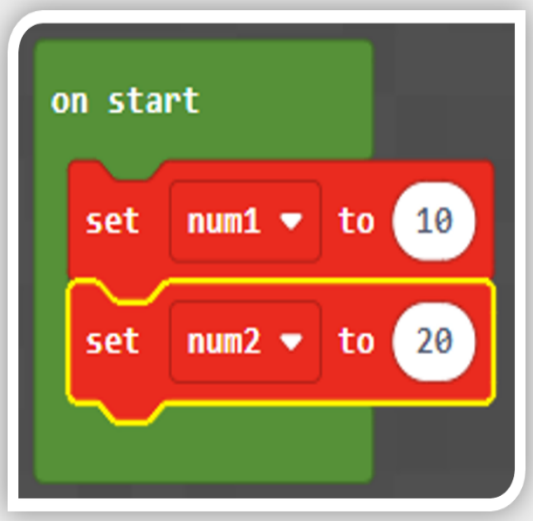
Click on A to print the next statement

Output for example 3

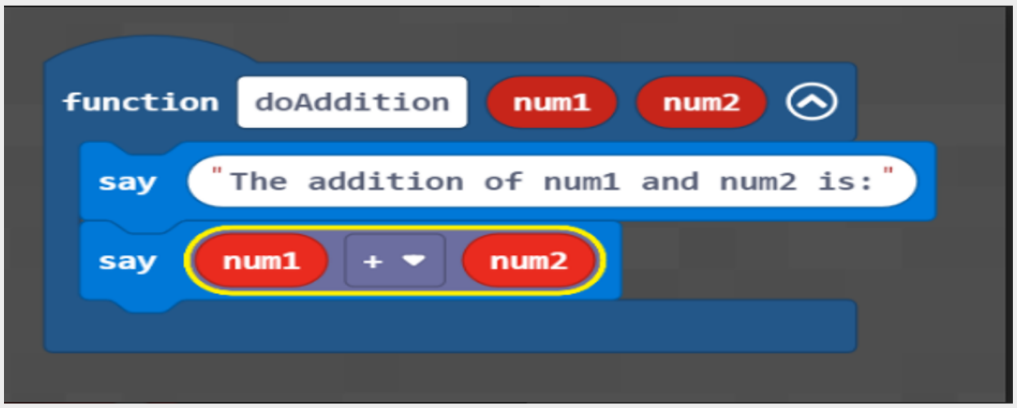
3.4 Activity: Adding two integers

In Minecraft Editor, click on "Make a Function" button from "Functions" link in toolbox.

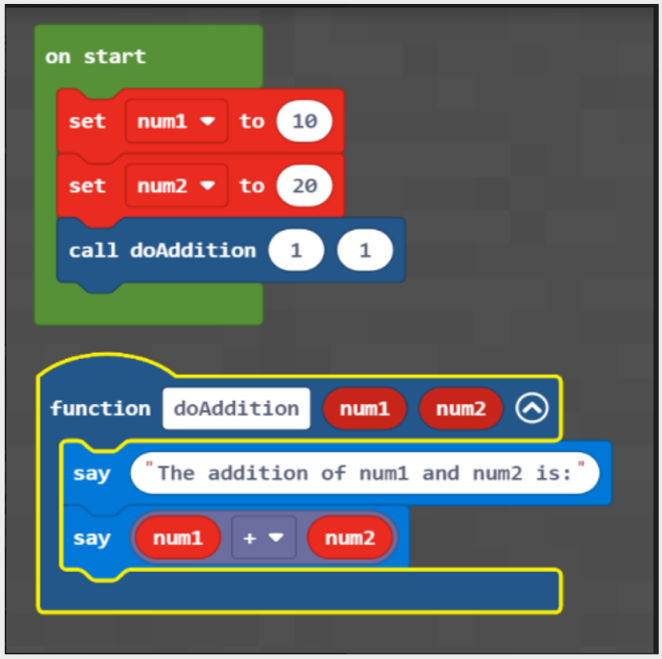
Step 1:
Click on "**Make a Function**" button from "**Functions**" link in toolbox. Name the Function as "*doAddition*". Click on "**Number**" button and add "*num1*" and "*num2*" as input parameters. Once done, click on "**Done**" button



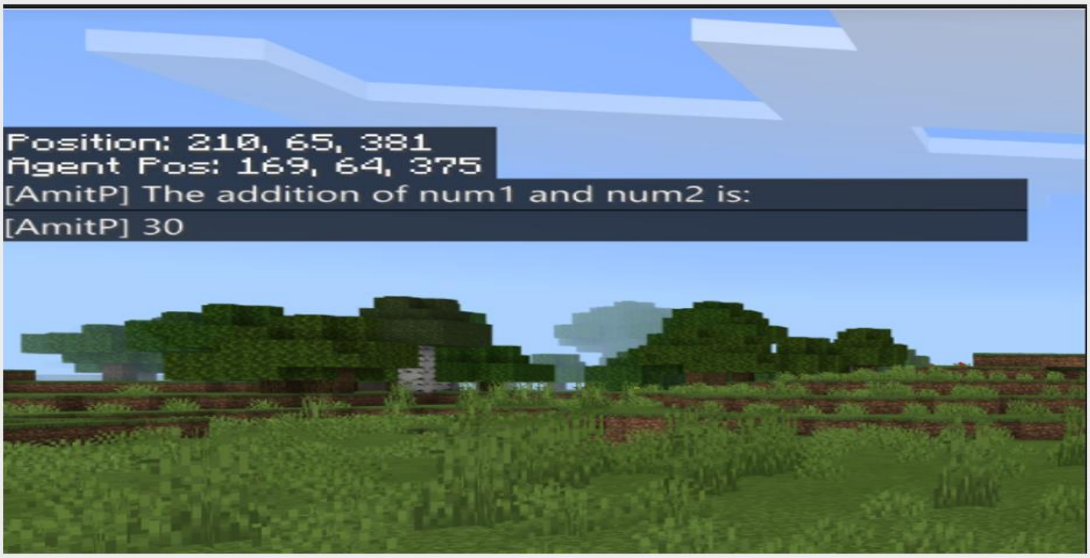
Step 2:
Create two variables "num1", "num2" and set its value to "10" and "20" as shown in the image. Attach these blocks to the "on start" block.



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

3.5 How to reduce redundancy using Functions?

To illustrate this, let us take a very basic human habit of drinking water.

If we look at the action of drinking water, it involves 4 main steps only:

- Take a glass of water.
- Sip the water from the glass.
- Gulp the water down the throat.
- Put down the glass when we are no longer thirsty.

Although the action "Drink water" constitutes of 4 steps, yet while describing our day today life routine to someone else, we use the phrase like "Drink water". This action of "Drink water" is easy for the other person to understand and covers up the above 4 steps in a single action. Thus, every time we include "Drink water" in our routine, it will automatically cover the above 4 steps.

Similarly, the main idea behind using a function in your code is to keep the code **DRY** (**D**on't **R**epeat **Y**ourself). Cutting out repeated commands helps to minimize errors, keeps code short, and saves programming time.

3.6 Advantages of using Functions

Few of the advantages of using functions are:

Increases readability makes code organized and easy to understand.

Reduces code length: redundant code is removed and replaced by functions.

Reusability: Code reusability increases.

3.7 What are different Function Parameters?

Consider a problem statement, where we are required to calculate the area of a circle. To do so, we need to know the radius of the circle whose area we are required to calculate.

However, the formula for calculating the area of a circle always remains the same.

e.g. Function to calculate & print area of a circle

```
Area of Circle (radius)
{
    Area = 3.14 * radius * radius
    Print Area
}
```

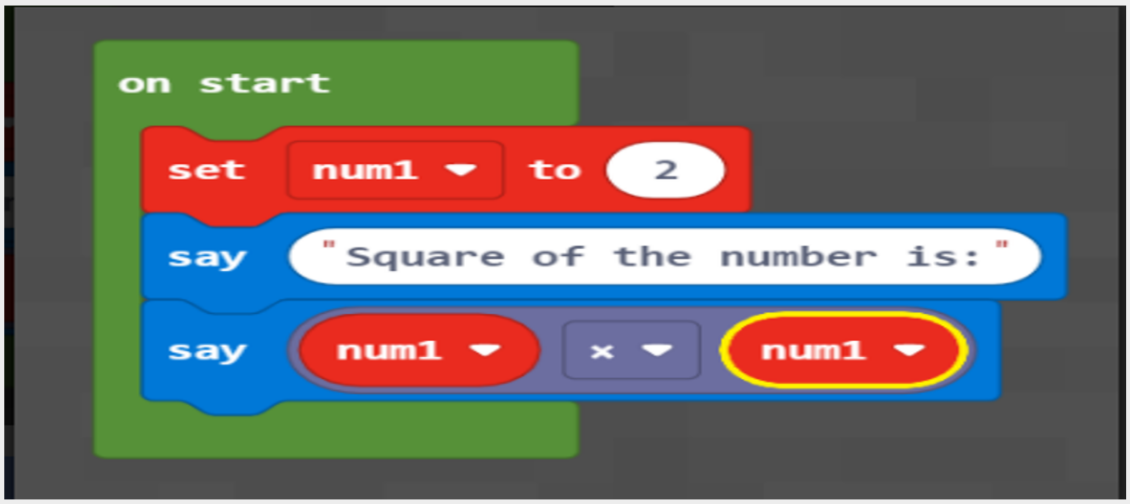
Here ***radius*** is the function parameter for the function **Area of Circle**

Similarly, if we want to calculate the area of a rectangle, we need to know the length and breadth of the rectangle. Here too, the formula for calculating the area of the rectangle always remains the same.

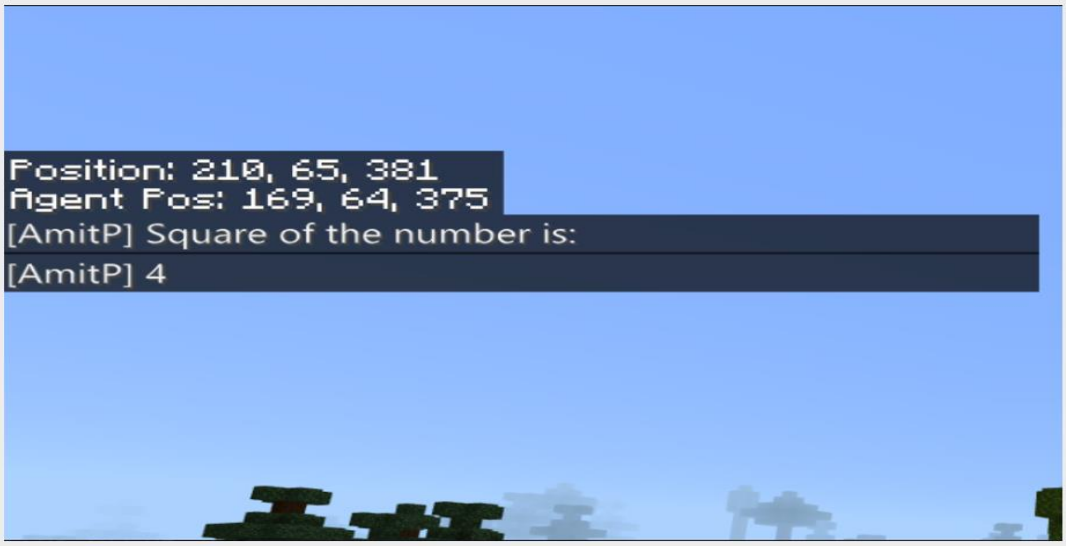
So, in the above-mentioned scenarios, if we consider the task of calculating the area of circle & square as functions respectively, then the formulae used to calculate the area can be considered as the body of the function. However, to get a concrete value from the functions, we need to provide the value of radius in case of calculating the area of a circle & the value of length & breadth to calculate the area of the rectangle.

Thus, the variables accepted by a function to perform the set of tasks defined in its body are called **function parameters**.

3.8 Activity – Finding the square of a number



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

3.9 Activity – Arranging the Books

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

Suppose you have a lot of books with you. However, all these books are mixed up. Now, 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*.

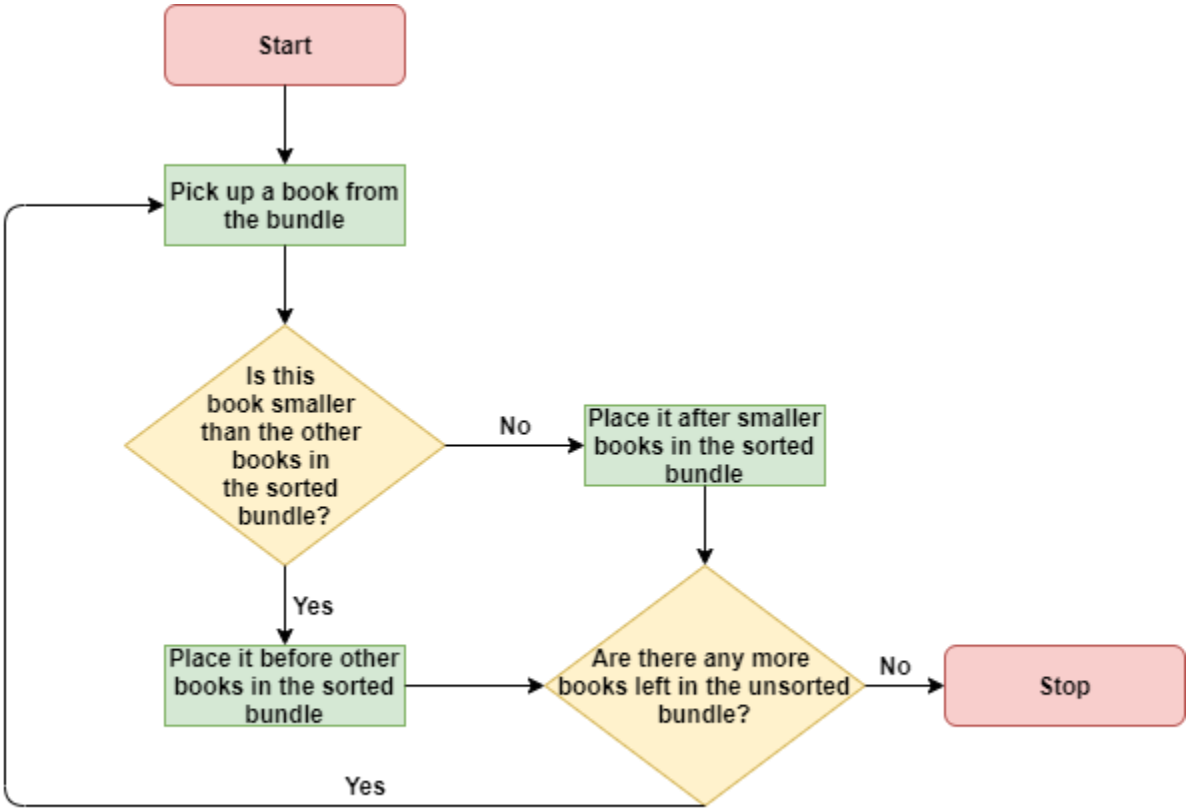


Fig 2.0 Arranging Books

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

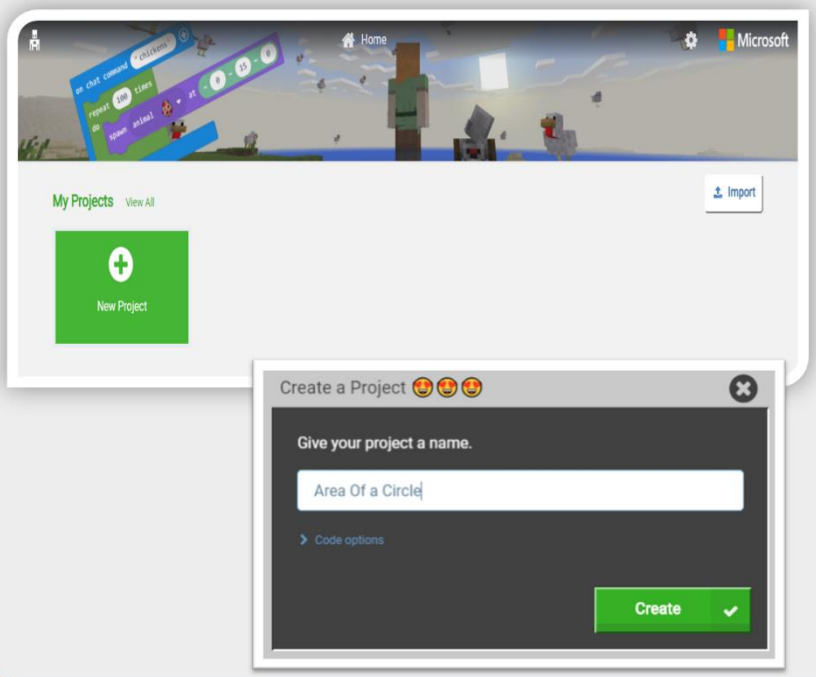
Sorting is helpful because:

1. Sorting a collection helps you answer questions like "which is the tallest/smallest/fastest item in this collection?"
2. Arranging things in order can make other algorithms work better.

3.10 Activity: Calculating area of a circle

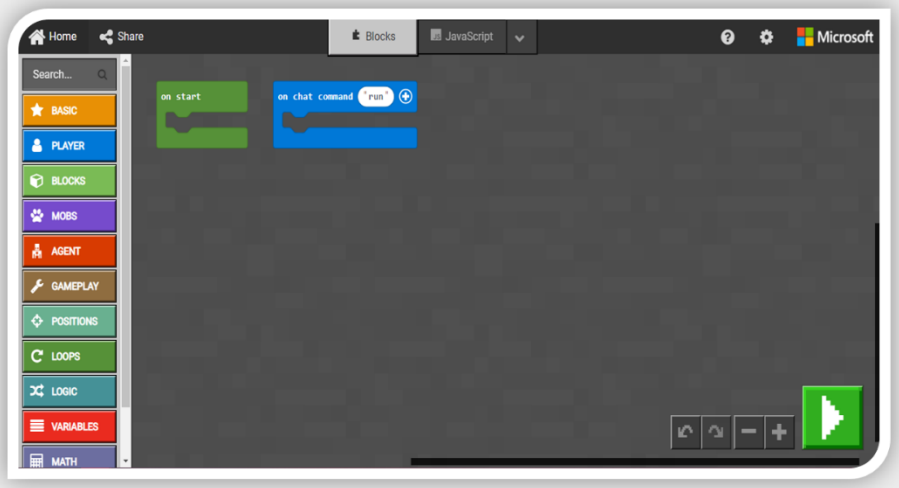
Let us now see how we can calculate the area of a circle in Minecraft using the above steps. You should try this exercise on Minecraft using the MakeCode editor for Minecraft, which can be found here https://minecraft.makecode.com/

First, create a new project, as shown below.

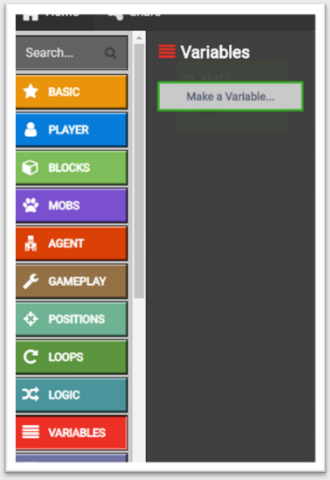


Creating New Project
You can create a new project by clicking on green box labeled as 'New Project'. A dialog box will appear prompting you to give a project name.

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

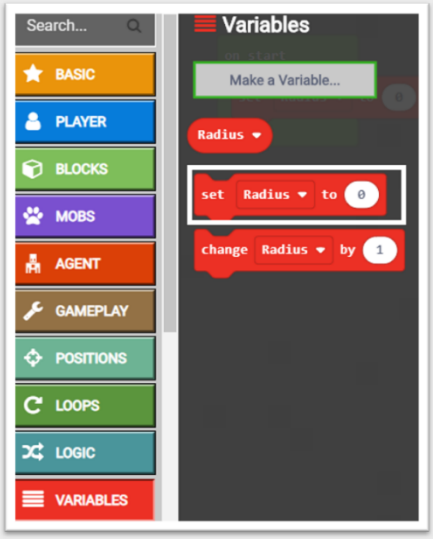


Minecraft Code Editor
Once you create your project, you should see the editor like this.



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

Step 2: Create a new variable called radius

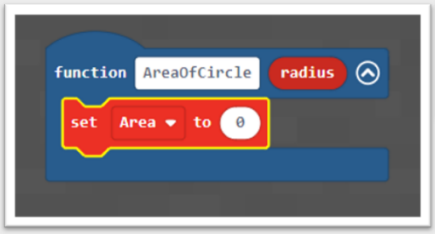


Step 3: From the **VARIABLES** menu select "**Set Radius to**" block

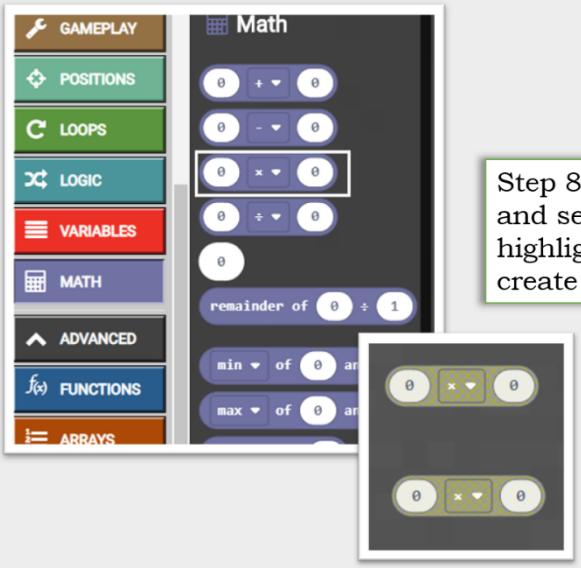
Step 4: Place the "**set Radius to**" block inside on start block and change the value to **5**

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

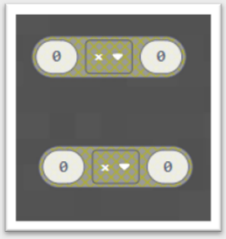
Step 6:Create a function with name "AreaOfCircle" having one numerical parameter named radius



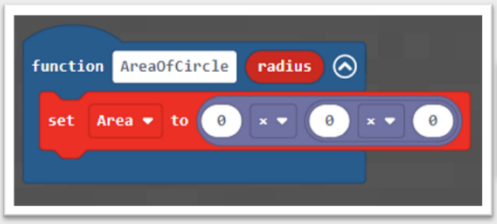
Step 7: Create a variable named Area select the "set Area to" block and place it inside function "AreaOfCircle"



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.

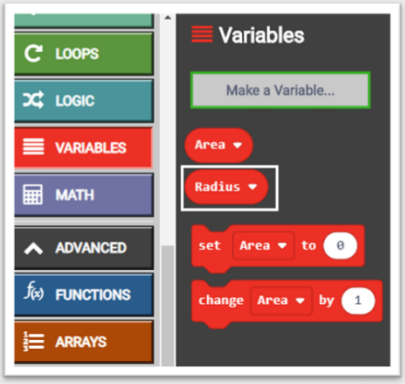


Step 10: Drag the multiplication block of 3 numbers to “Set Area to” block inside the function block “AreaOfCircle”

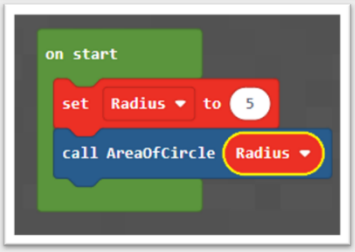
Step 11: Inside the function “AreaOfCircle” , inside the “set Area to” block , set the first number as 3.14 and the second & third number to the parameter radius

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

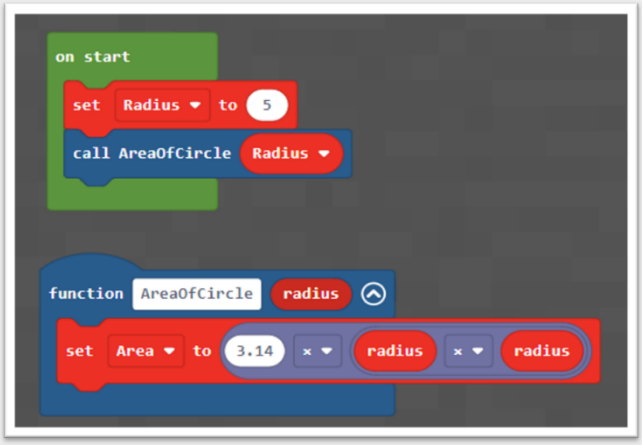
Step 13: Drag the “**call AreaOfCircle**” block into “**on start**” block under “**set Radius to**” block



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

3.11 Can Function return a value?

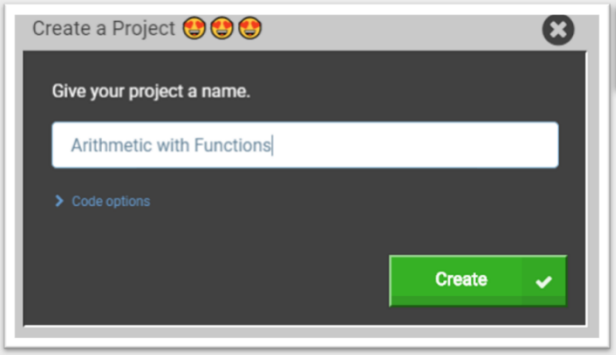
Till now we have only used functions in a way wherein once a function is called, execution of all the logic and display of the output was done inside the function.

However, as discussed before, the main purpose behind using functions is to get rid of repetitive chunks of code. Thus, the usefulness of using a function in a program comes to the forefront, when an operation performed inside a function gives back a value, which can be used later in the program to generate meaningful results.

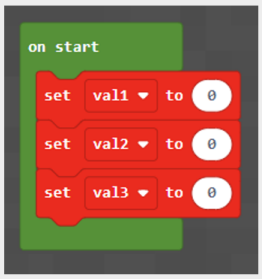
As an example, consider a scenario where in we calculate the square of a number using a function, calculate the cube of another number using another function and then add the results generated from these two functions and print them.

```
square_of_number(input1)
{
        result1 = input1 * input1
        return result1
}


cube_of_number(input2)
{
        result2 = input2 * input2 * input2
        return result2
}

main function()
{
       val1 = square_of_number(2)

       val2 = cube_of_number(3)

       val3 = val1 + val2

}
```

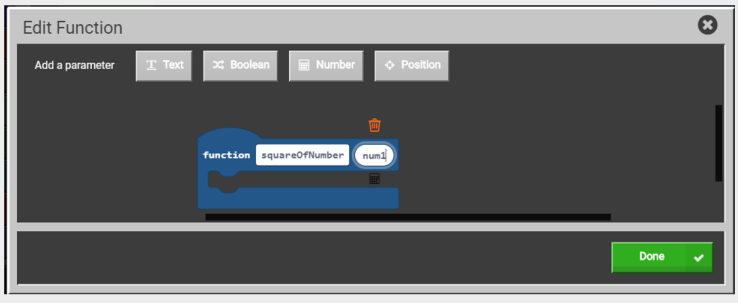
Let us see how we can implement the above example via Minecraft



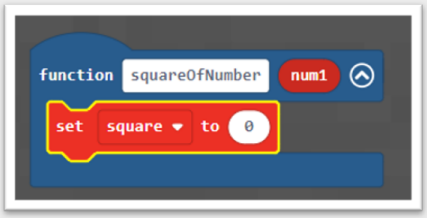
Step 1: Create a new project **Arithmetic with Functions**



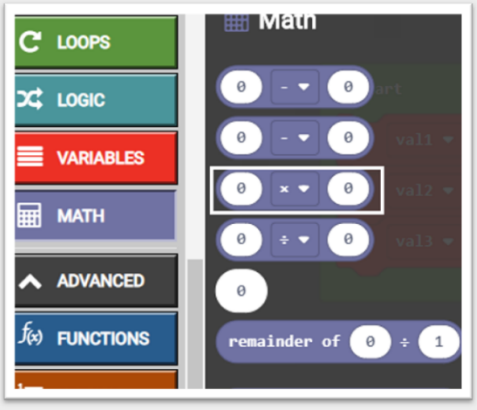
Step 2: Create three variables val1 , val2 & val3. Create three blocks “set val1 to” , “set val2 to” & “set val3 to” and place them under on “on start” block



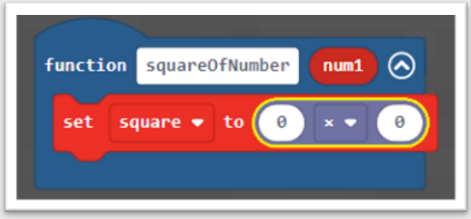
Step 3: Create a function named squareOfNumber having a numeric type parameter.



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

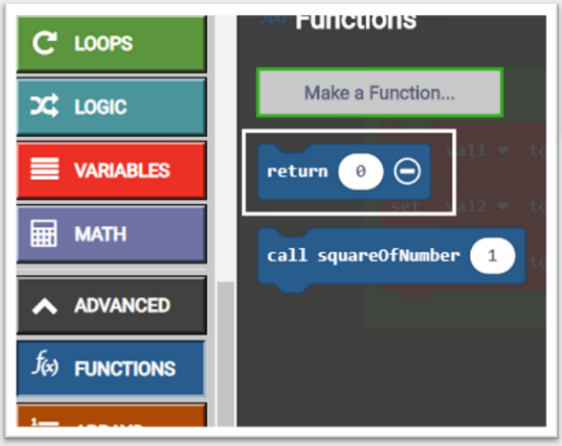


Step 5: Select the Math block type and select a multiplication block

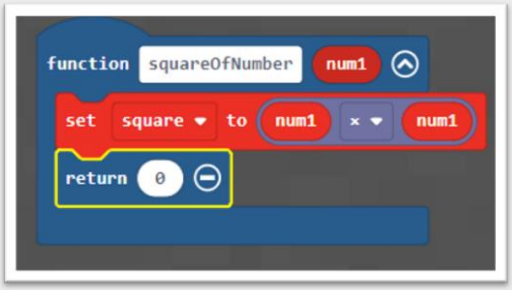


Step 6: Attach the multiplication block to the “set Square to” block inside the function squareOfNumber

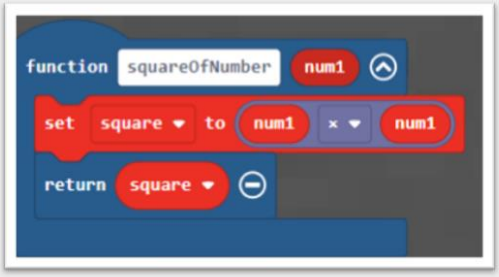
Step 7: Set the values of each number inside the multiplication block to num1



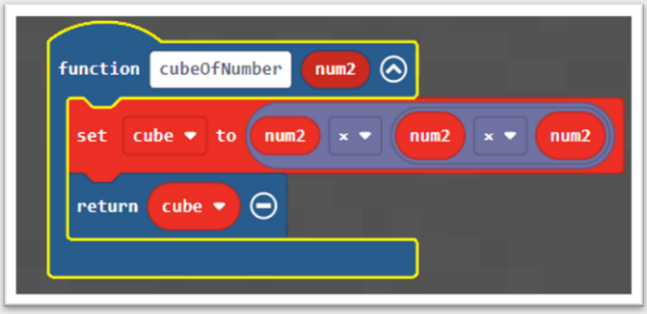
Step 8: Select block type Functions and select “return” block



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

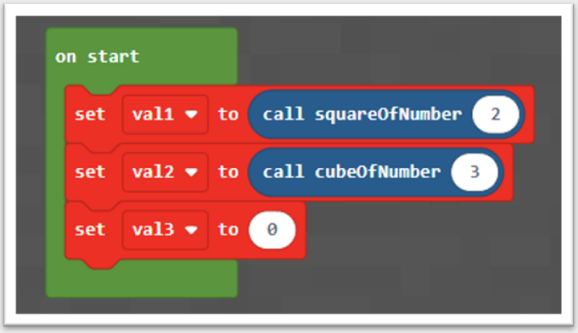


Step 10: Set value inside return block to square variable



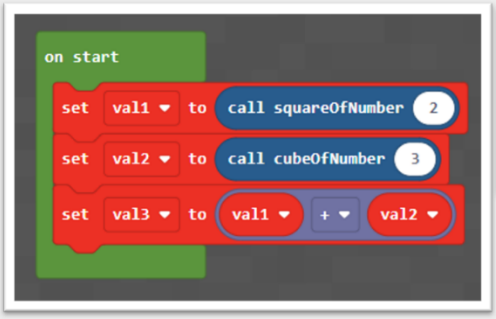
Step 11: Similarly, we can create a function cubeOfNumber having a numeric parameter num2 (try to derive the shown block code as a practice exercise)

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

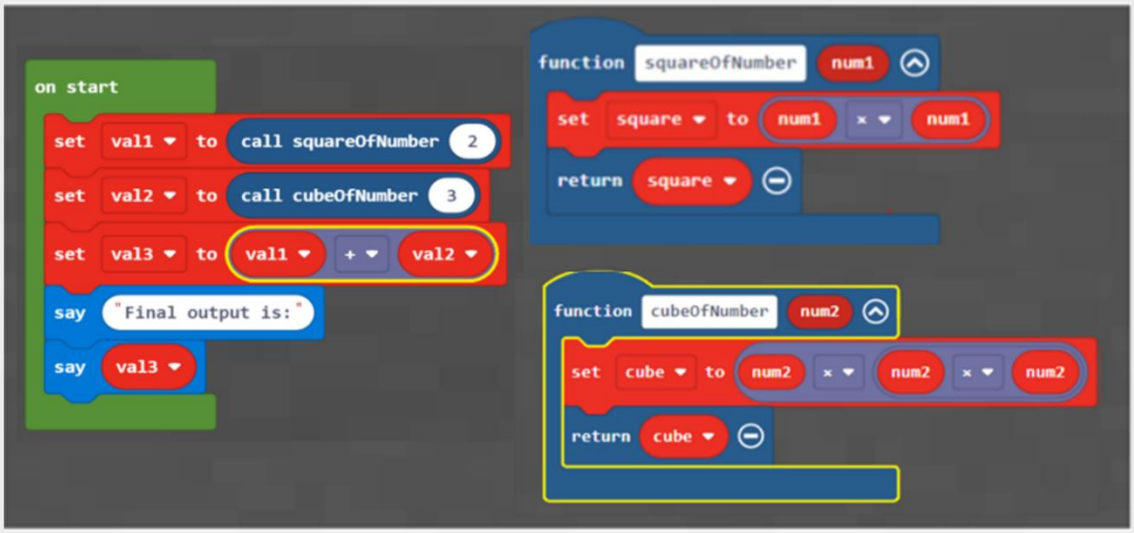


Step 13: Attach “call squareOfNumber” block to “set val1 to”.
Change value in “call squareOfNumber” to 2
Attach “call cubeOfNumber” block to “set val2 to”.
Change value in “call cubeOfNumber” to 3

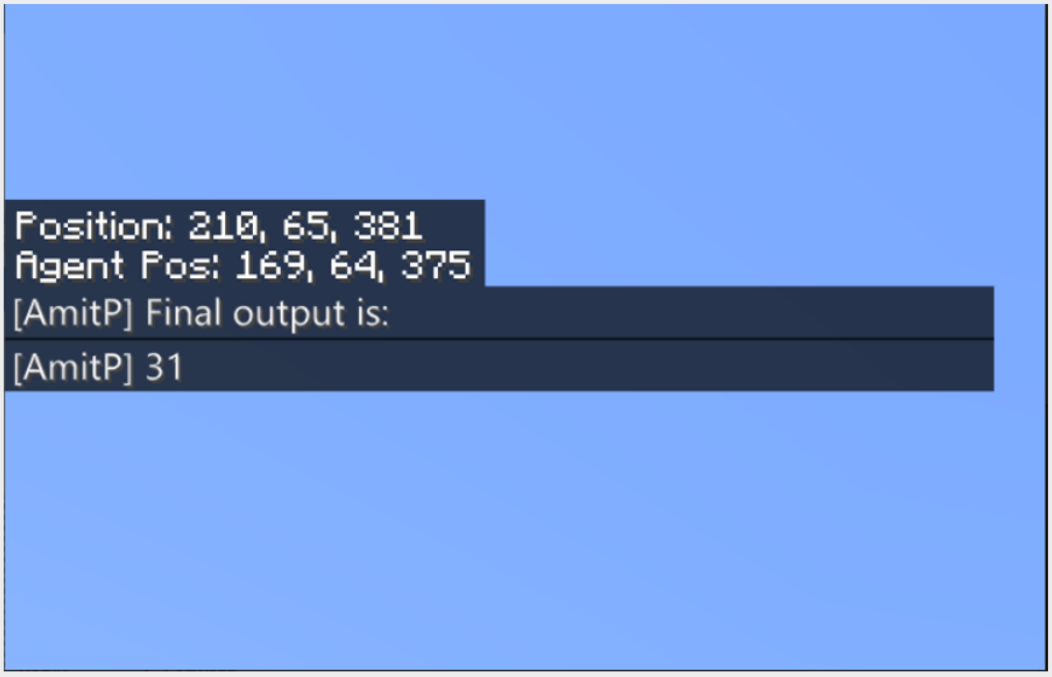
Step 14: Select Math block type and select an addition block



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



Final output for the activity can a function return value

Note: Minecraft 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 (https://code.org/)

3.12 What is an event?

An event is something that happens. In the context of coding, events are a generalization of all the things that the program can respond to.

Some common examples of event include:

1. Clicking on the button of a web page
2. A web browser fully loading a web page
3. Key press inside a desktop application

3.13 What are Event Handlers?

An event handler is a piece of code associated with an event in the code. An event handler gets executed, when that event occurs. For example, in MakeCode platform if you want a player to move forward when the user clicks on the forward icon you would create a code block like below.

This "on player walk" block is an event handler. Agent mode forward is the action that occurs with this event.

3.14 Quiz Time

Objective Type Questions

Question 1	Function in a program decreases readability
Option 1	True
Option 2	False

Question 2	Function in a program increases code reusability
Option 1	True
Option 2	False

Question 3	Functions in programming help in reducing code redundancy
Option 1	True
Option 2	False

Question 4	A function can return a value
Option 1	True
Option 2	False

Question 5	What do we call the variables found in the definition of the function on which further operations may be performed?
Option 1	Return value
Option 2	Parameter(s)
Option 3	Data types
Option 4	None of the above

Microsoft

Standard Questions

1. Explain five advantages of using functions.
2. What are function parameters?
3. Explain how functions help in reducing redundancy of code.
4. Can functions return a value? Support your answer with proper explanation.
5. Name three scenarios where using functions will help you reduce redundancy of code
6. Explain parameterization of functions. Where can and why do you think parameterization is useful?
7. Explain what the output of below program will be:

```
def test(x, y):


    if x > y:
        return y


def test1(x, y, z):
        return test(x, test(y, z))


print(test1(2, 7, -1))
```

8. Find an error in the below program:

```
def sum(numbers):
        total = 0
        for x in number:
                total += x
        return total
print ( sum ( ( 1 , 2 , 3 , 4 , 5 ) ) )
```

Higher Order Thinking Skills (HOTS)

1. Write a function to print even numbers from 1 to 50
2. Write a function to do sort the elements [34,12,89,56,98,101] from biggest to smallest.
3. Write a function to find the highest number in the array of [1, 87, 09, 18, 11]
4. Write a function to find the modulus of 3 and 6.

Applied Project

The practice challenges should be done on the Minecraft platform.

1. Create a block code which takes radius and height of a cylinder as input to a function and computes its volume. The volume computed should be returned by the function.
2. Create a block code which computes product of (sum of two random numbers) with the (difference of two random numbers) using functions.

3.15 What have you learnt in this chapter?

By now you:

- Should have an understanding about the usefulness of using functions in code.
- Should know how to define ,call a function and pass parameters in a function.
- Should know how different types of value are returned by a function.

UNDERSTANDING ARRAYS & COLLECTIONS

4.1 What will you learn in this chapter?

This chapter will teach you how we can work with large set of data in programming. What are Arrays and its practical implementation in computers. This chapter will also give you an overview of collections. We will work on some fun exercises to get better understanding of these concepts

4.2 What are Collections?

A collection is nothing but a container that groups multiple items into a single object. Collections are used to store data. We can also retrieve and manipulate data that is stored in the Collections.

In real world, you can consider a collection of cards, a telephone directory or a mailbox to be an example of Collections.

Advantages of using Collection

- Collections allow programmers to group multiple items into a common set
- Grouping of items makes it easier to identify and perform different operations on them like retrieving and modifying data
- It increases the performance of the program
- It increases productivity and reduces operational time.

4.3 Activity – Algorithm for a perfect square

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 ways to improve the algorithm?

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.

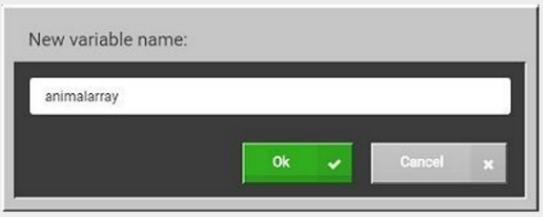
As you must have noticed, the number of times we multiply, and compare is reduced significantly in the above algorithm. This is how we can use optimized algorithm to find the square of numbers.

4.4 Activity Building a Zoo

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.

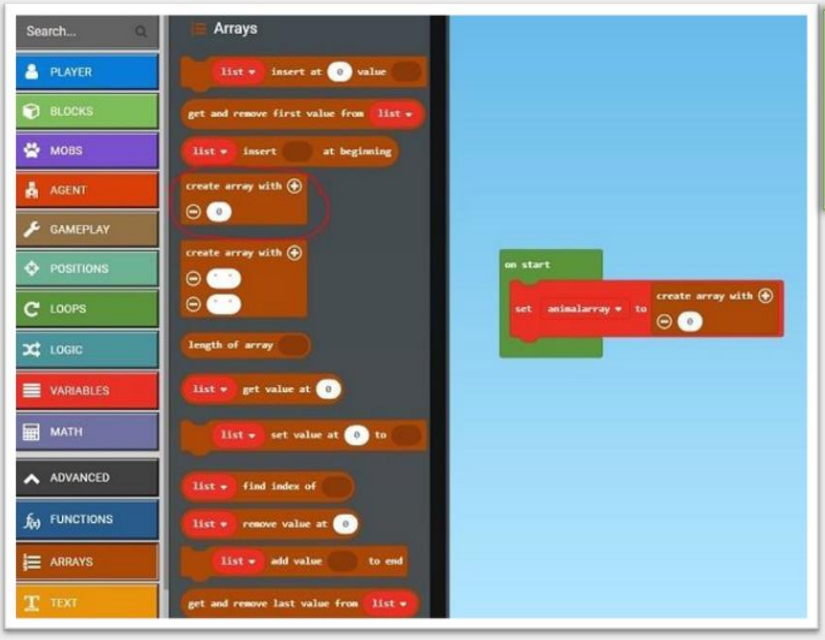


Step 3: From “Variables” , click the **Make a Variable** button. Name this variable “animalarray”, and click “Ok”

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

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

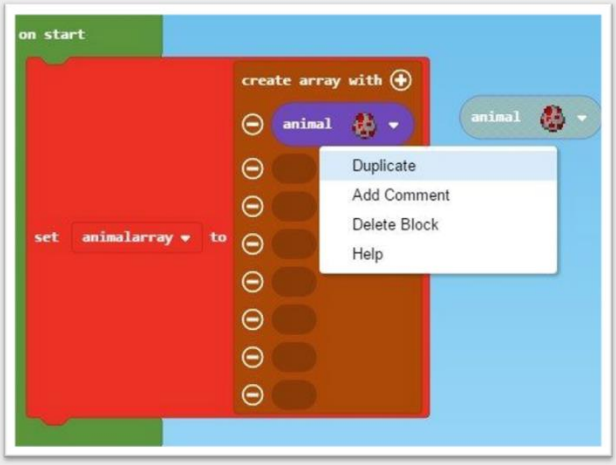
Step 6: Click on the Advanced tab in the Toolbox to display the “Arrays” Toolbox drawer.



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.

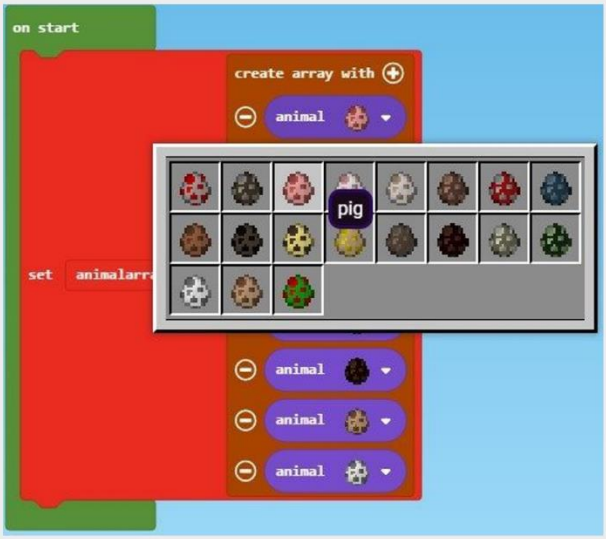
Step 9: From "MOBS", drag an Animal block into the first slot of "create array with".



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 can place fences along this path.

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

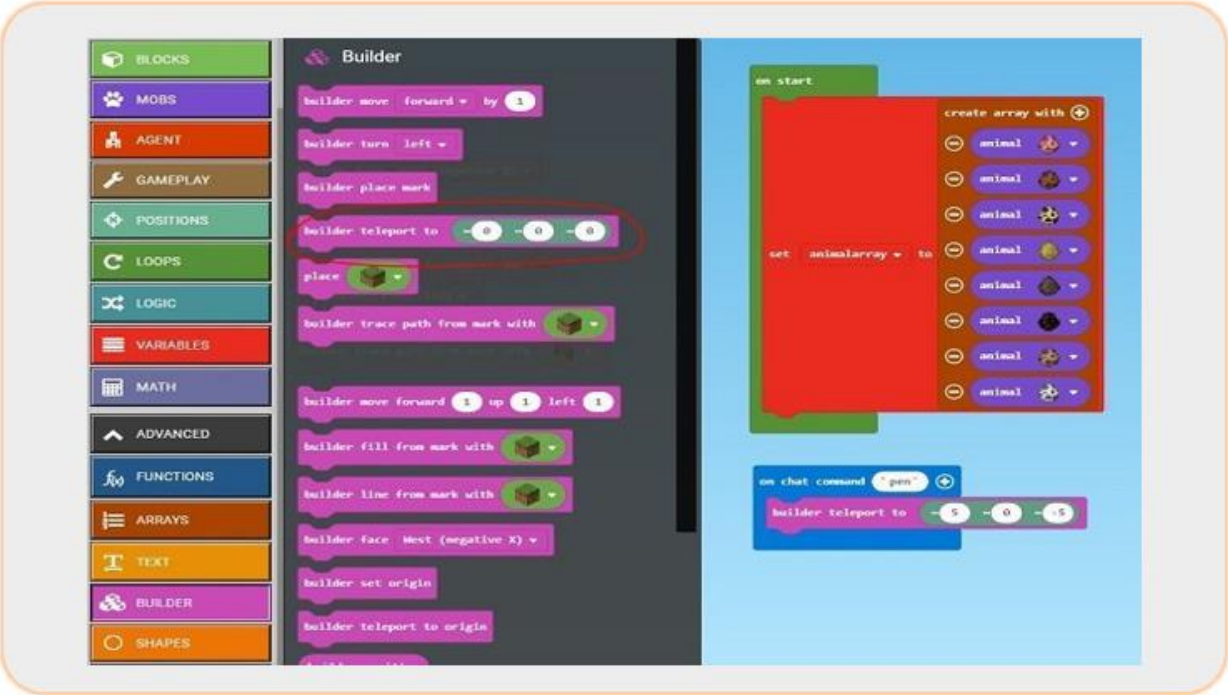
Step 15: Rename the command "pen".

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.

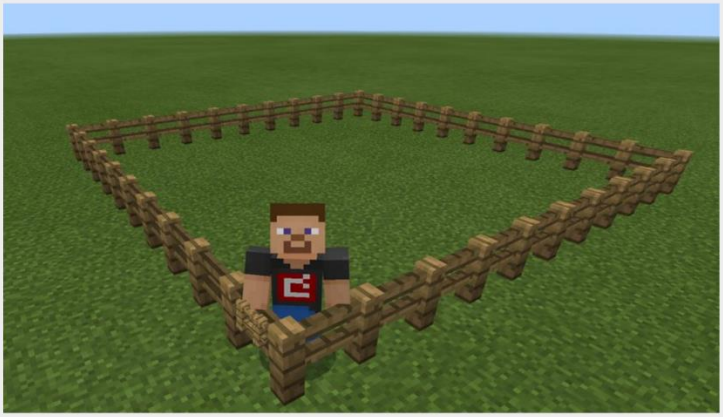
Step 25: Type 10 into "builder move" to make the sides of your pen 10 blocks.

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!

Note: Minecraft 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

4.5 What are Arrays?

An Array is a collection of similar data type variables. Arrays do not support different data types in same collection. For example, you can make an Array of Integers as well as another Array of Strings. However, you cannot make an Array having Integer and Strings in same Collection.

In real world, you can consider books in a library to be the example of arrays where all the shelves have a common data type (read – book) in it.

Arrays improve readability of code by using a single variable for a large set of data. However, there are following limitations that we need to note while using Arrays.

Limitation of Arrays

- You can only store variables with same data types in an Array
- The variables in an Array are always ordered sequentially with index starting with 0

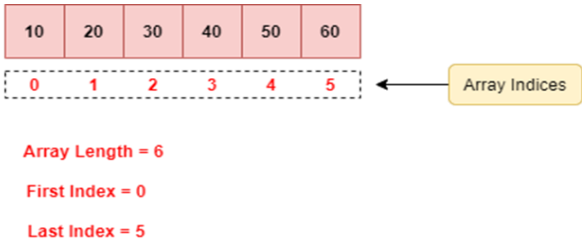
Consider an Array with variables of Integer Data Type stored in it.

This array can be declared in following format:

arr = [10, 20, 30, 40, 50, 60];

Using the above syntax, we can deduce that an array named "arr" is declared with data type as integer i.e this array can only hold integer values. We have initialized this array with six values i.e 10, 20, 30, 40, 50, 60.

Below is the diagram that displays how data and indexes are structured in such an Array:

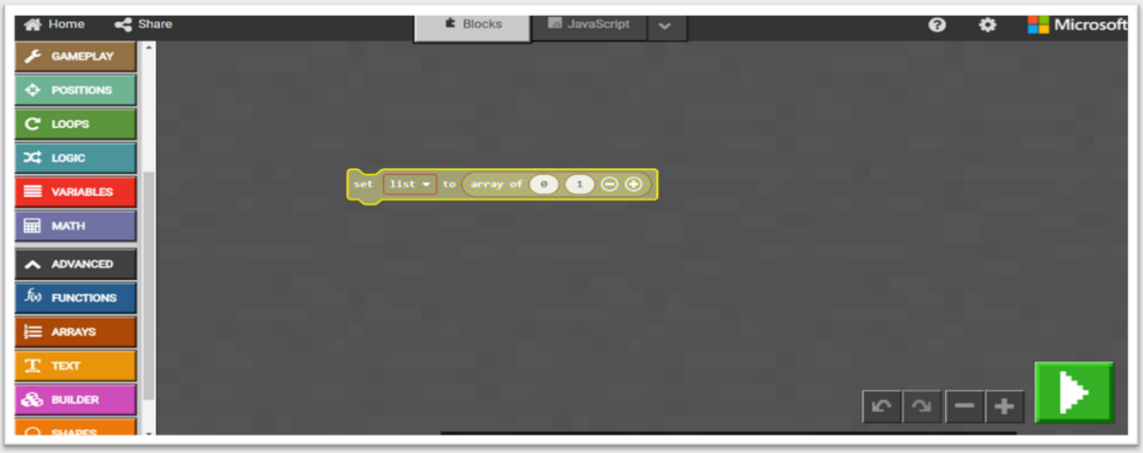


Now, let us try to implement similar example using Minecraft Platform. Next exercise will help you understand step by step how to create an Array and calculate its length using Minecraft.

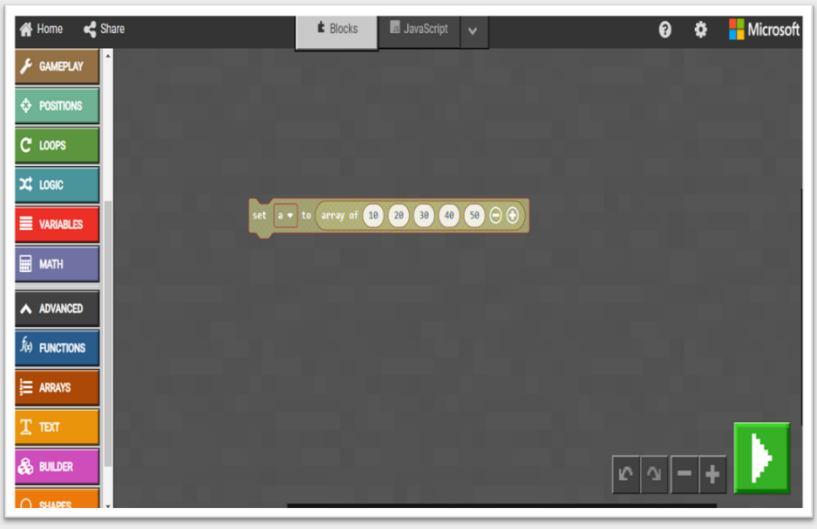
Once you complete this exercise, your final output should look as shown in the below screenshot:

Let us now follow below step by step procedure to replicate this output on our screen:

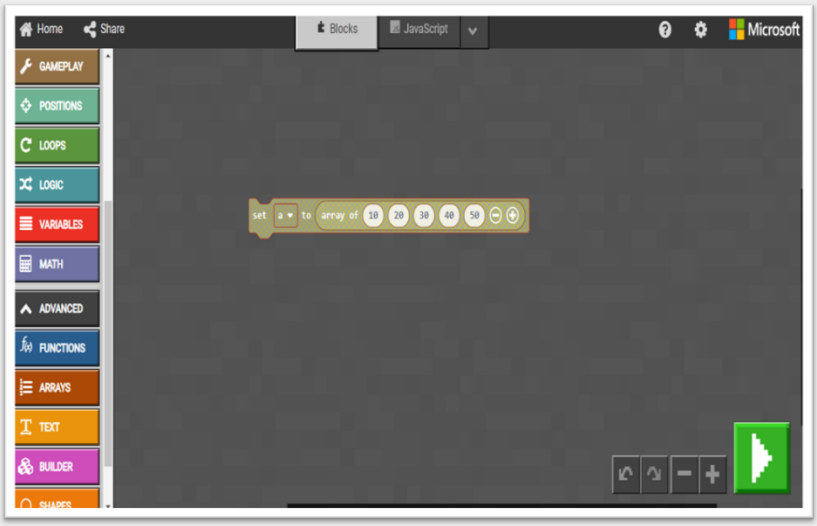
Step 1: Create a new project in Minecraft. Click on “Variables” link from toolbox and click on “Make a Variable”. Create a Variable “a” and Click on “Ok” button



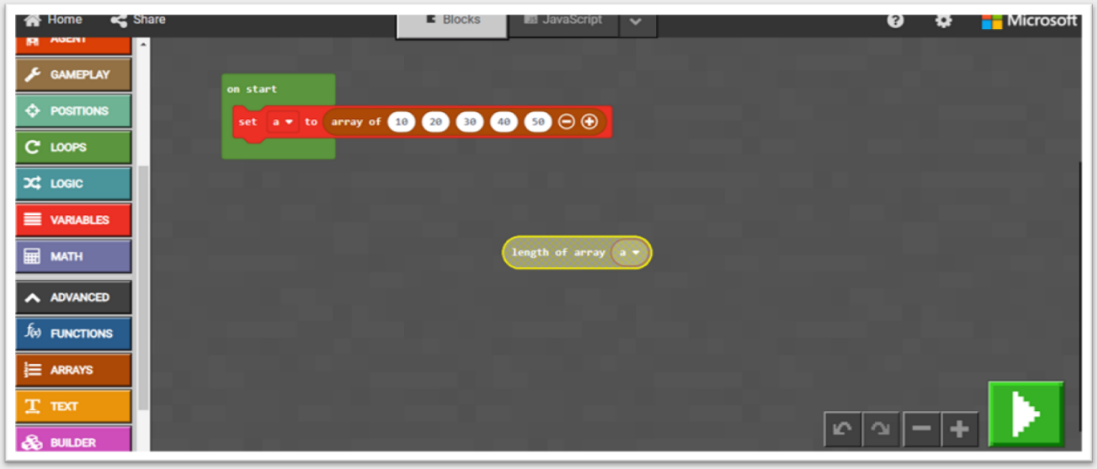
Step 2:From the Toolbox, click on link “Arrays”. From the sub list, drag and drop block of “set list to” to the play area



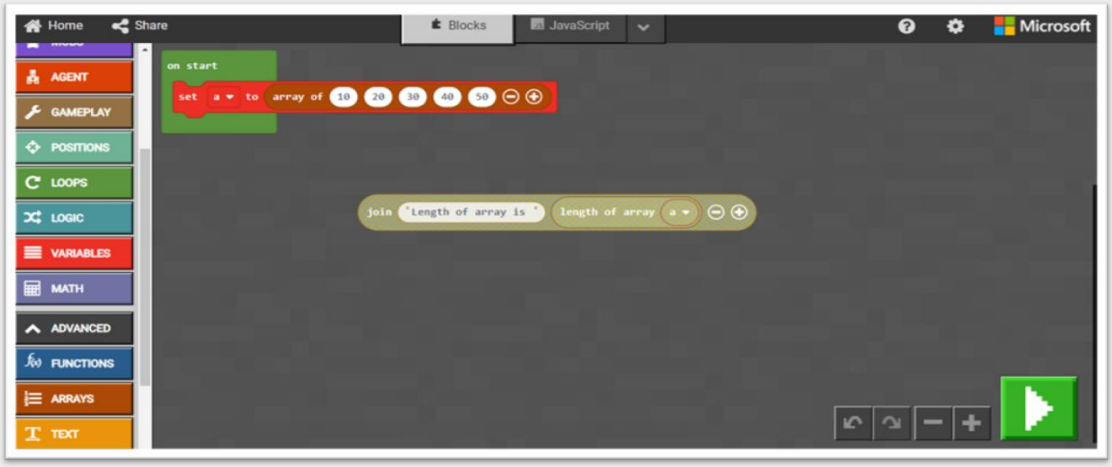
Step 3: In the set block, click on the drop down which says “list” and select variable “a” that we had earlier created. In the next text boxes you can rewrite values such as “10”, “20”, click on “+” icon and few more elements as shown in the image.



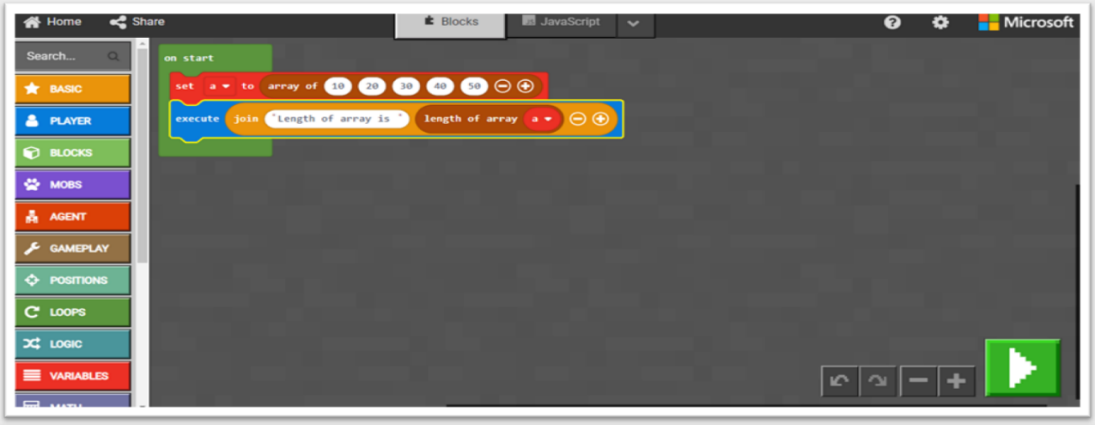
Step 4: Now click on "Loops" link from the tool box. Drag and drop block "On Start" into the play area. Following this, attach the "set" block that we had created into "On Start" block as shown in the image.



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.

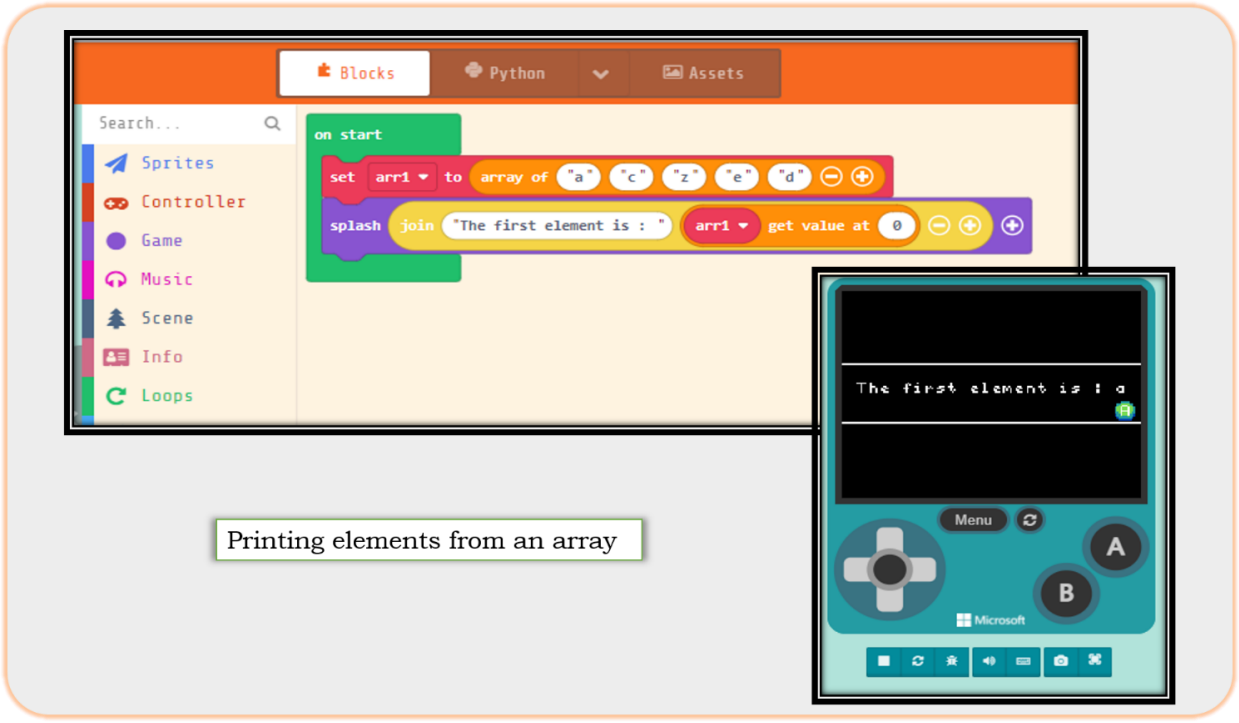


Step 7: Click on "Player" link from the Toolbox and drag and drop "execute" block in the play area. Now, append "join" block that we had created into "execute" block and place "execute" block below "set" block on "on start" block as shown in the image. Finally, click on green play button in the bottom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

Note: Minecraft 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

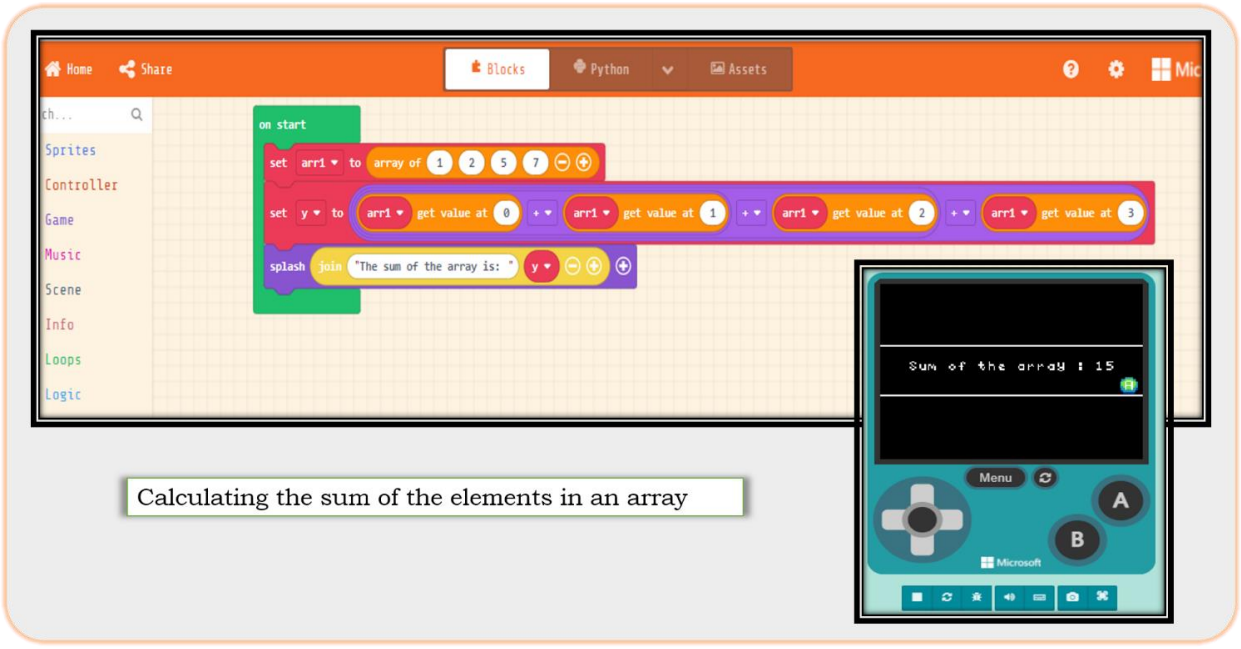
4.6 Examples of arrays using Arcade

Example 1: Printing the first element of the array



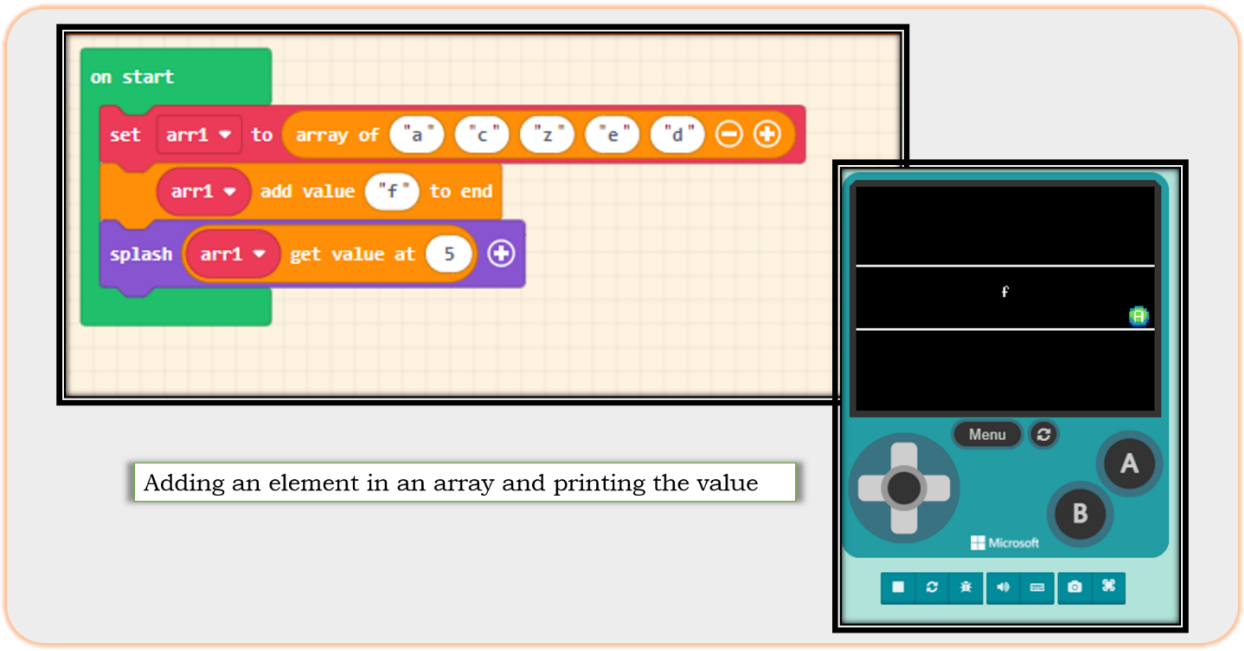
Printing elements from an array

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



4.7 How can we iterate over collections?

As we have understood so far, Collections can consist of lists, sets, or maps. There is a lot of data that we can store in collections. Naturally, we need to use this data or perform modifications on this data depending on the program requirement. To do this, we need to iterate through the data that is stored in these collections. This is where the concept of Iterator comes into the picture.

The iterators are used to provide users a uniform way of accessing collections in a sequential manner. Whatever type of collection we are using, we always need to traverse through the elements of these collections to fetch the data or make any modifications to that data.

However, modifying a collection's elements while iterating through that collection is not allowed and throws an error in the program. We cannot directly add or remove elements while iterating through the collection that includes them.

4.8 Modifying Collections

Now that we have learnt what are collections, how do we store data in collections and iterate over the data, next question that may come to your mind is how do we modify this data that is stored in the collection?

Programming facilitates multiple ways of modifying the data stored in collections. Not every collection can be modified and

there is a limitation on what can be modified in a collection. As we have read above, modifying a collection's elements while iterating through that collection is not allowed and throws an error in the program. We cannot directly add or remove elements while iterating through the collection that includes them.

Below are certain points that we need to note about collections:

- Collections that do not support modification operations (such as add, remove and clear) are known as “unmodifiable”. Collections that are not unmodifiable are modifiable.
- Collections which guarantee that no change in the Collection object will be visible, are known as immutable. Collections that are not immutable are mutable.
- Lists assure that their size remains constant even though the elements can change are known as fixed size. Lists that are not fixed-size are referred to as variable-size.
- We cannot iterate or modify a Collection that is null.

Let us now understand the modifications that we can do on collections.

Adding Elements During Iteration

To add elements while iterating a list, set or map, keep the new elements in a temporary list, set, or map and add them to the original after you finish iterating the collection.

Removing Elements During Iteration

To remove elements from a list, you can create a new list, then insert the elements you wish to keep. Or, add the elements you wish to remove to a different list and remove them after you finish iterating the collection.

4.9 Quiz Time

Objective Type Questions

Question 1	What is the starting index of an array?
Option 1	-1
Option 2	1
Option 3	2
Option 4	0

Question 2	Which one is an incorrect array?
Option 1	Arr[] = [‘a’ , ‘b’ ,’c’ , ’d’]
Option 2	Arr[] = [1 , 2 ,3 ,4]
Option 3	Arr[] = [1, ‘a’, ‘b’, 2]
Option 4	Arr[] = [2.0 , 9.4, 5.6 , 6.7]

Question 3	An array is
Option 1	A group of elements of same data type
Option 2	A type of collection that contains more than one element
Option 3	A type of collection in which elements are stored in memory in continuous or contiguous locations
Option 4	All of the above

Question 4	Select a correct statement about Arrays
Option 1	An array address is the address of first element of array itself
Option 2	An array size must be declared if not initialized immediately
Option 3	Array size is the sum of sizes of all elements of the array
Option 4	All of the above

Microsoft

Standard Questions

1. What are collections?
2. What are different types of collections?
3. How can we iterate over collections?
4. What are the different types of modifications that we can perform on collections?
5. What is an Array Index?
6. Do collections allow backward traversing of data? Support your answer with a proper explanation.
7. Name three real life collections which work like Arrays.

Higher Order Thinking Skills (HOTS)

1. Draw a flowchart for the optimized algorithm for finding the square of numbers that we learnt in this chapter
2. Write a program to create an array of prime numbers from 50 to 100.

Applied Project

Create an exercise in Minecraft to create and Arrays of even numbers from 1 to 20 and then iterate over each element of this Array.

4.10 What have you learnt in this chapter?

By now you:

- Should know what arrays are and their practical implementations
- Should know about collections

HELLO WORLD WITH CODE

5.1 What will you learn this chapter?

At the end of this chapter, you will understand the basics of a programming language.

5.2 What is a Programming Language?

A computer program is a set of instructions that is given to a computer to execute. A programming language is a tool we use to write instructions for computers to follow.

We need a programming language since computers cannot understand the languages we speak. Computers can only understand binary strings of 1s and 0s.

The very earliest computers were programmed by changing ones and zeros manually, alternating the circuit and the wiring. All modern programming languages, however, are made up of a series of syntax and symbols that act as a bridge that allows humans to translate their thoughts into instructions that computers can understand.

Over the years, thousands of programming languages have been created, and more are being created every year. Some of the most popular programming languages are Python, JavaScript, Java and C++. Later in this chapter, we will look at some of the basic syntax of Python.

Low Level vs High Level Programming Language

Programming languages are of two types, high-level languages and low-level languages. Both high-level and low-level languages have their own applications, so it is important to understand the difference between the two types of languages.

A low-level programming language is one that is made to be easily understood by the computer. They include machine code and assembly language, both of which instruct computer hardware components to carry out instructions directly. However, low-level programming languages are difficult to learn and time-consuming to code.

A high-level programming language is made to help human programmers communicate easily to the computer. Good examples of high-level programming languages include C, Java, Python. These languages are usually compiled or converted into low-level programming languages so that they can be executed directly. The person who invented the concept of a compiler was Grace Hopper, sometimes called "Amazing Grace".

5.3 Activity – Sorting the list

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

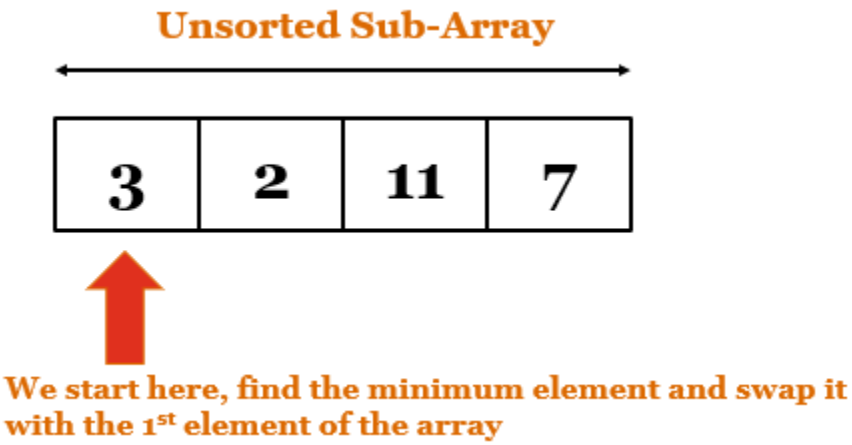
Step 4: Increment MIN to point to next element

Step 5: Repeat until list is sorted

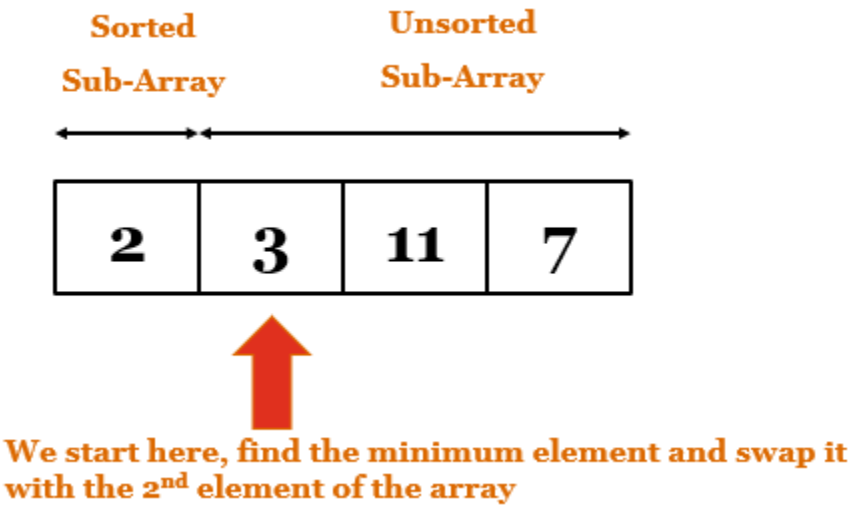
Let us now sort the below array in ascending order using selection sort algorithm:

Array = {3, 2, 11, 7}

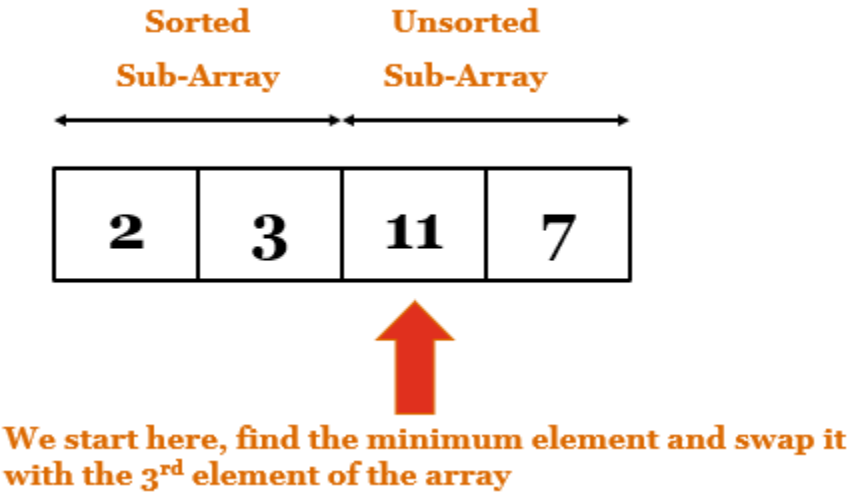
Step 1: For i = 0



Step 2: For i = 1



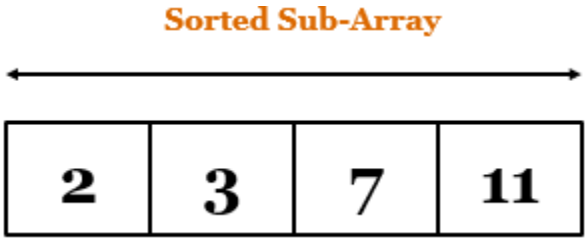
Step 3: For i = 2



Step 4: For i = 3

The loop gets terminated as "I" becomes 2.

The final state of array after loop is terminated looks like below:



5.4 Getting used to syntax

Python is an easy to learn programming language that can be used to build a variety of applications. Python is used in web development, game development and development of desktop-based applications. Python is also widely used for data science, machine learning and artificial intelligence.

You will be surprised to know that some of the world's most famous websites like Google, YouTube and Instagram use Python!

Now let us get started with some basic Python syntax.

5.5 What are Variables and Data types in programming?

Variables are used by programmers to store values. Whenever a variable is created, Python assigns a spot to it so that its value can be stored till the program runs. We can store lots of different types of information in a variable. For example, we can store a number, text, or a list of items. We can also change the value assigned to a variable while the program is running.

Variables are useful for several purposes like counting how many times an event occurs, saving inputs from users, and performing complex calculations.

In any programming language, a variable can only store a value that is in a format. These formats are called data types. Python also has several in-built data types. First, let us understand three basic data types in Python - string, integer and Boolean.

Strings

In Python, a string is defined as a sequence of characters enclosed in either single or double or triple quotes.

The characters can be alphabets of any language or symbols. Strings are also called str in short.

You can perform many different operations on strings. You can access each character in a string using its position in the string. You can also join two or more strings together to create a longer string. To concatenate strings, we use a + symbol between the two strings.

Let's look at the syntax for these operations.

Defining a variable 'Text' and setting its value.

```
Text = "This is a string variable."
```

2. Accessing the first letter of the 'Text' variable. The value printed will be T.

```
print(Text[0])
```

3. Defining a second variable 'Text2' and joining it to 'Text'. The value printed will be "This is a string variable. This is a second variable."

```
Text2 = "This is a second variable."

print (Text+Text2)
```

Integers and Floats

In Python, integer variables are used to store whole numbers, which can be positive or negative. Quotation marks should not be used while defining integers. Integer variables can be treated like numbers for all mathematical operations.

If you need to store decimal numbers in Python, you should use float variables as integer variables ignore the decimal part of a number. When you combine a float variable and an integer variable in a mathematical calculation, the resulting answer will always be a float. You can do several arithmetic operations with floats and integers. Let us see some common operations with floats and integers.

- Addition
- Subtraction
- Multiplication
- Division

- Modulus (Remainder)

Now that we know what Integers and Floats are, let us now understand the syntax for performing different Mathematical operations on them in our Program:

1. Performing Addition of Integers and Floats

```
x = 1
y = 2
sum = x + y
```

2. Performing Subtraction of Integers and Floats

```
x = 1
y = 2
subtraction = x - y
```

3. Performing Multiplication of Integers and Floats

```
x = 1
y = 2
multiplication = x * y
```

4. Performing Division of Integers and Floats

```
x = 1
y = 2
division = x / y
```

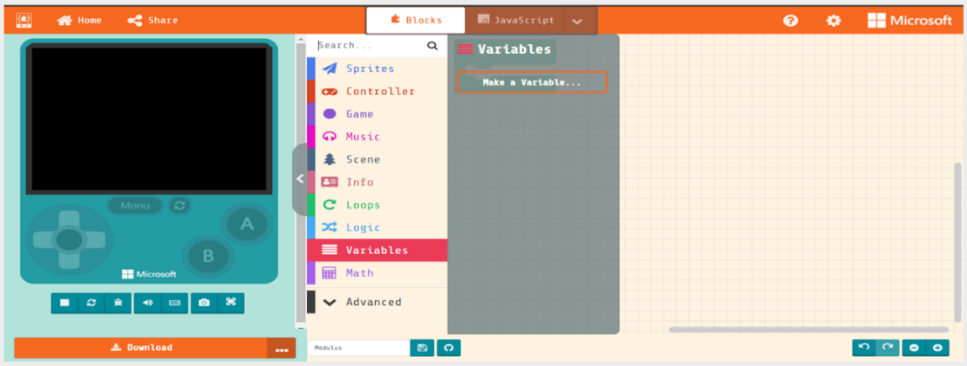
5. Performing Modulus of Integers and Floats

```
x = 1
y = 2
modulus = x % y
```

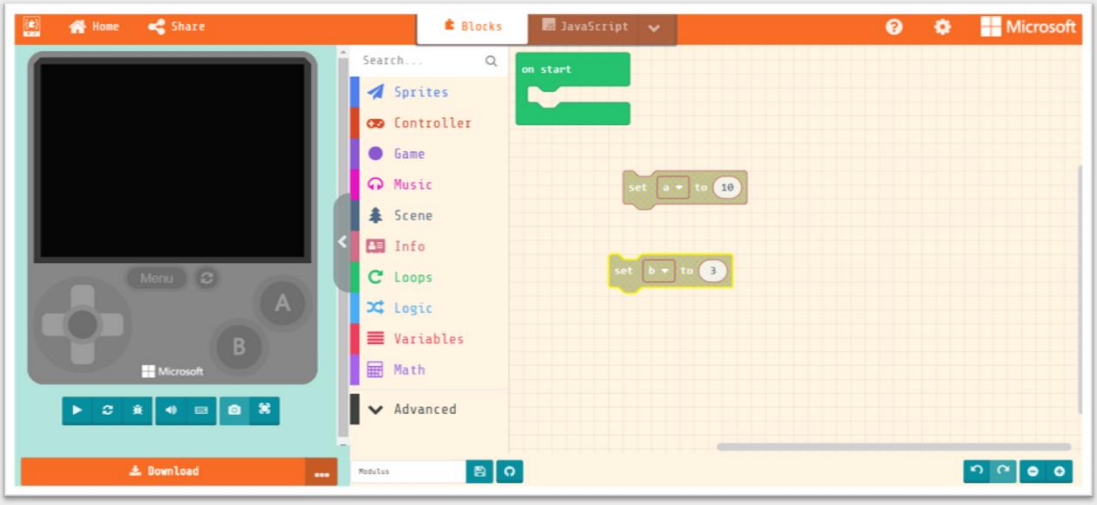
Now that we are familiar with various arithmetic operations, in the next exercise, we will get to understand more on this topic.

5.6 Activity: Building a Calculator

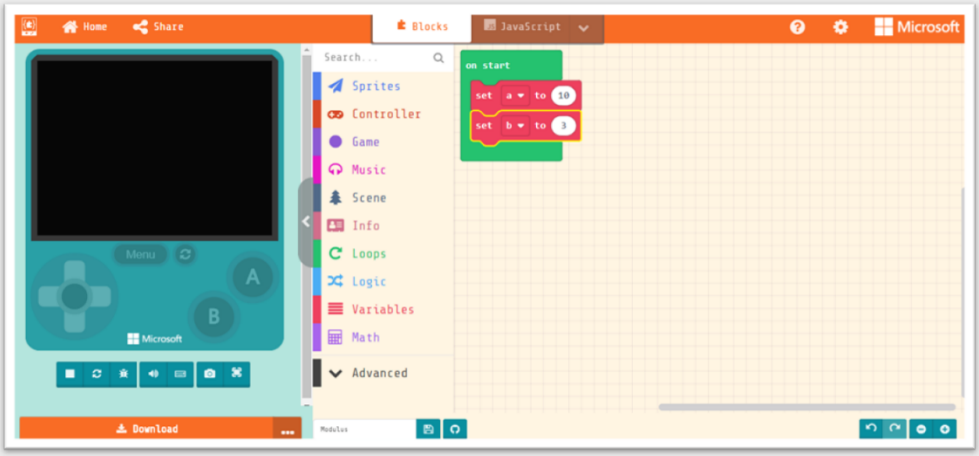
To understand these operations better, let us now perform the below activity. You need to open MakeCode editor for this activity.



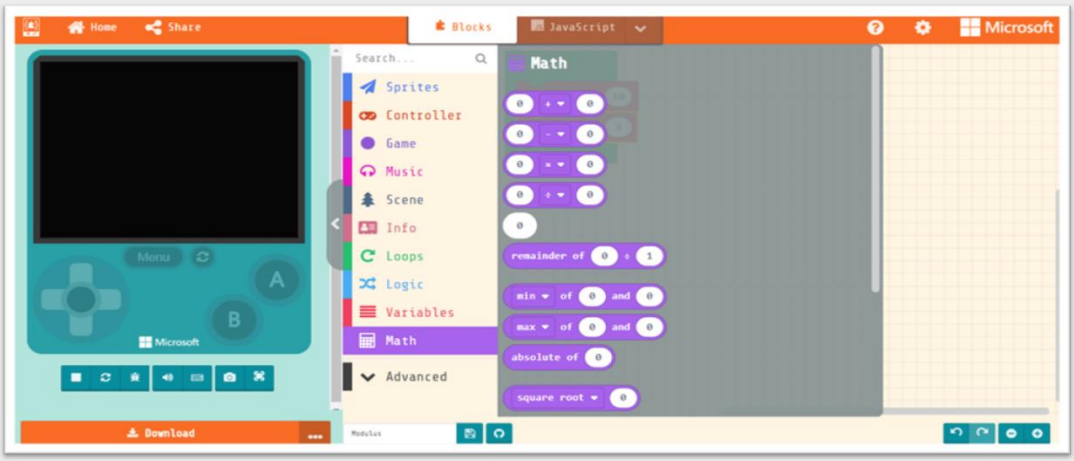
Step 1: In Arcade MakeCode Editor, click on the “Variables” link from the ToolBox and then click on “Make a Variable” link



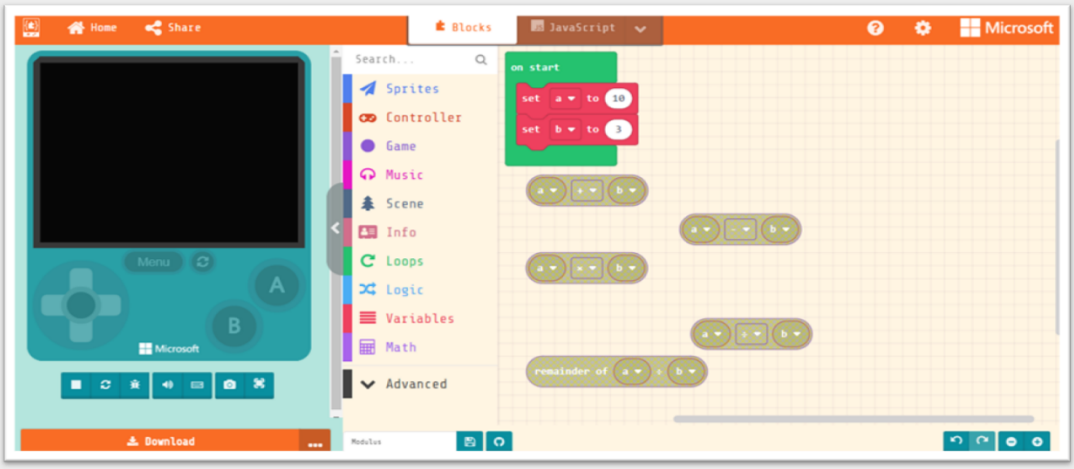
Step 2: Make a Variable “a” and set it value to “10”. Similarly, create another variable “b” and set its value to “3” as shown in the image



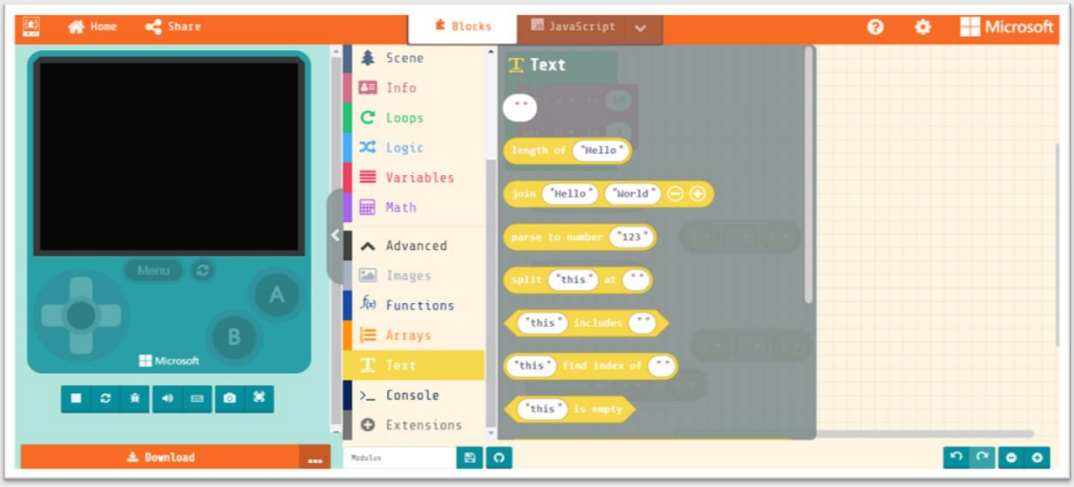
Step 3: Now Drag and drop blocks of both the variables in “on start” block



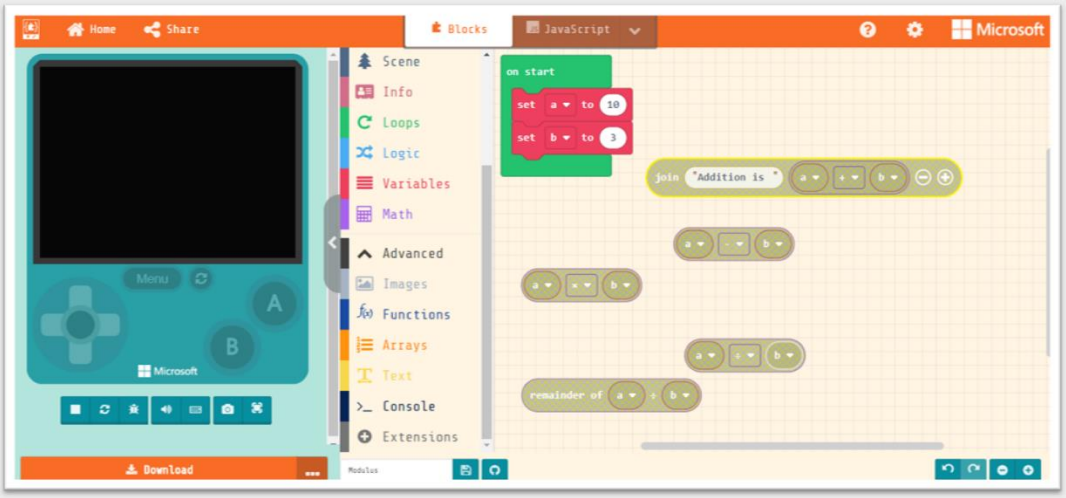
Step 4: Click on “Math” link from the Toolbox and drag and drop “Addition”, “Subtraction”, “Multiplication”, “Division” and “Remainder” Blocks in the Play Area



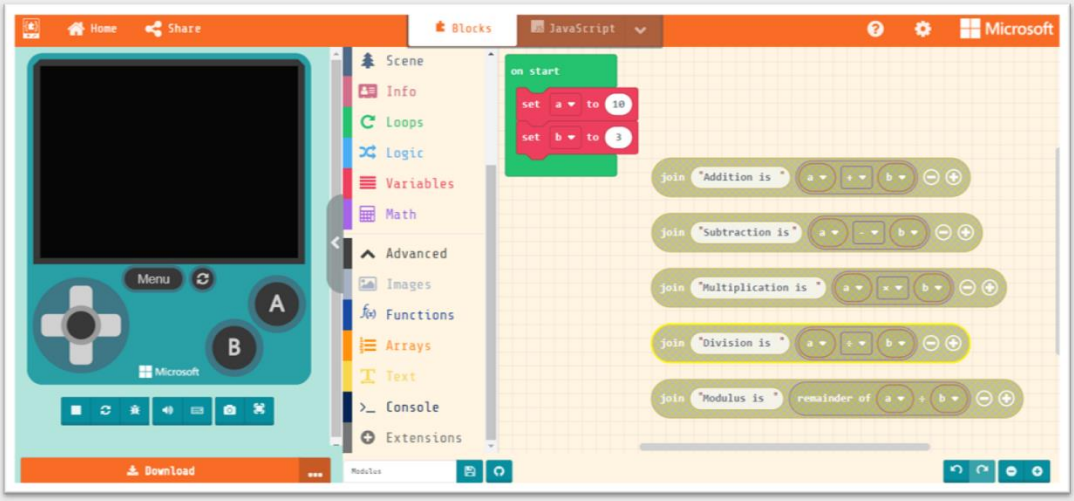
Step 5: Now, fix variables “a” and “b” in the mathematical blocks of the play area



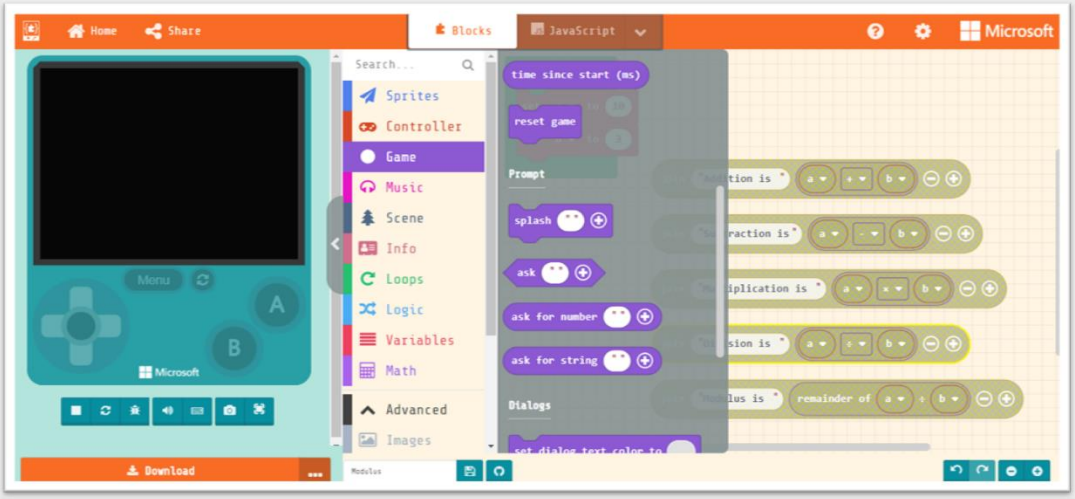
Step 6: Click on “Text” link from the Toolbox and drag and drop “Join” block to the Play area



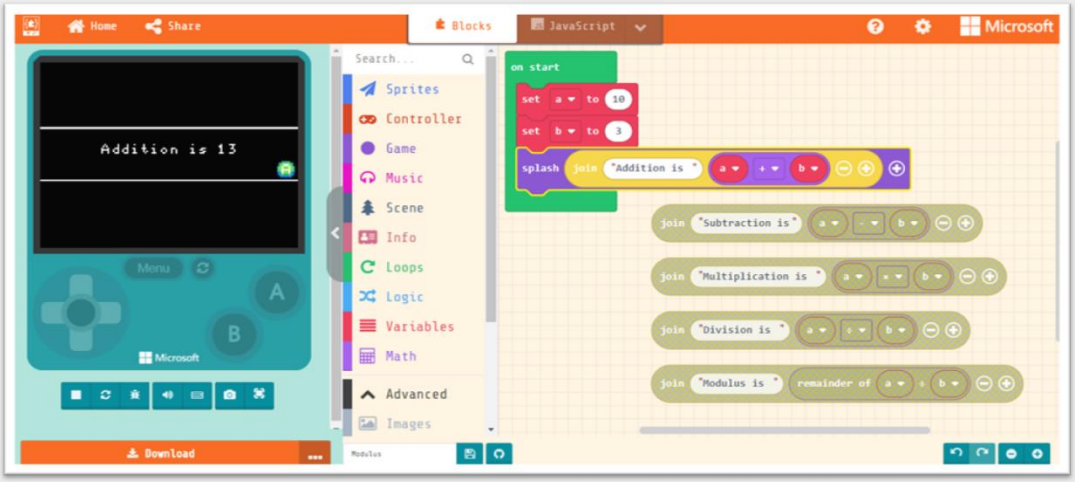
Step 7: In the “Join” block, rename first text box to “Addition is “and attach addition block in the second text box as shown in the image



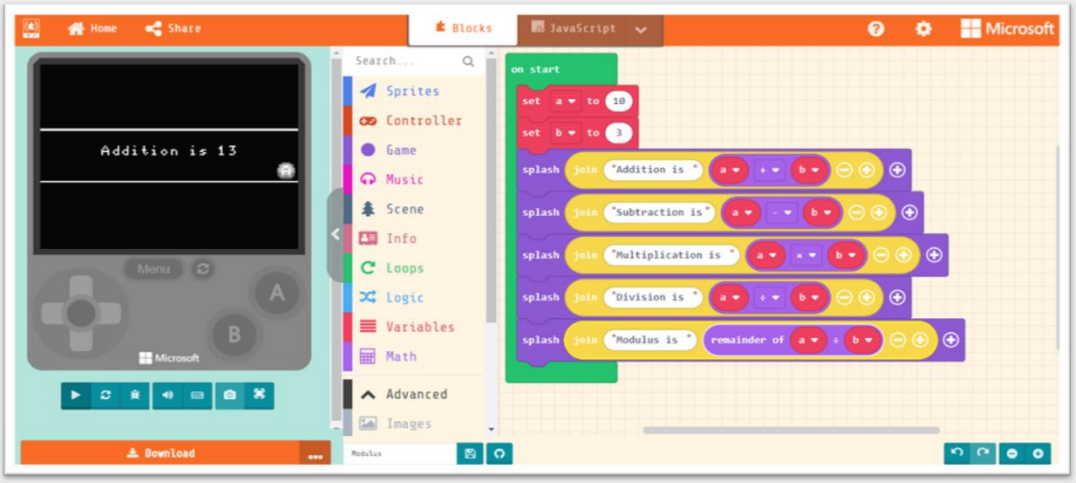
Step 8: Repeat Step 7 for other mathematical operations in the play area



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

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

5.7 Quiz Time

Objective Type Questions

Question 1	Low level languages are user friendly
Option 1	True
Option 2	False

Question 2	Which of the following is a valid data type?
Option 1	string
Option 2	integer
Option 3	float
Option 4	All the above

Question 3	Which of the following is a valid arithmetic operation?
Option 1	Addition
Option 2	Multiplication
Option 3	Modulus
Option 4	All the above

Question 4	Which of the following is NOT a string type?
Option 1	character = “Mr.”
Option 2	vehicle = “Truck”
Option 3	speed = 70
Option 4	lives = “12”

Question 5	In computer science, variables:
Option 1	are number like Pi
Option 2	represent parts of an experiment that are measured or tested
Option 3	are placeholders for storing information
Option 4	are unchangeable

Question 6	Translator which is used to convert codes of assembly language into machine language is termed as
Option 1	Assembler
Option 2	Attempter
Option 3	Compiler
Option 4	Debugger

Question 7	Language in which single statements can be written to accomplish substantial tasks is termed as
Option 1	Machine Language
Option 2	Assembly Language
Option 3	High Level Language
Option 4	Medium Language

Short Answer Questions

1. What are High Level and Low-Level Languages in Computers?
2. What is the Syntax to declare a String variable in Python?
3. What are different Mathematical Operations that we can perform on variables in Python?
4. Explain different data types used in Python with the help of its syntax
5. Explain what are variables and why are they used in programming
6. Research and name five programming languages. Categorize them in the buckets of high-level programming language and low-level programming language
7. Give one example where you can use mathematical operators in Python to build a program that will help you in day to day chores.

Higher Order Thinking Questions(HOTS)

1. Draw a flowchart to explain the flow of selection sort technique in programming.
2. Write an algorithm to sort the below list in descending order using Selection Sort Technique:
   [10, 21, 45, 67, 12]

Applied Project

Create a program in Arcade to perform Modulus operation on two variables.

5.8 What have you learnt in this Chapter?

This chapter gave us a basic overview of creating a program in Python.

- We have learnt about high and low-level programming
- We have also learnt about the syntax in python and variables and data types.
- We also performed a fun activity to create a Calculator in Arcade.

REFERENCES

Microsoft. 2021. Microsoft MakeCode Arcade. [Online]. [25 February 2021]. Available from: https://arcade.makecode.com

Microsoft. 2021. Microsoft MakeCode for Minecraft. [Online]. [25 February 2021]. Available from: https://minecraft.makecode.com

Microsoft. 2021. Computer Science Subject Kit | Minecraft: Education Edition. [Online]. [25 February 2021]. Available from: https://education.minecraft.net/class-resources/computer-science-subject-kit

ACM, Inc. 2021. Code of Ethics. [Online]. [25 February 2021]. Available from: https://www.acm.org/code-of-ethics

Association for computing machinery (acm). 2016. CSPathsala. [Online]. [16 March 2021]. Available from: https://cspathshala.org