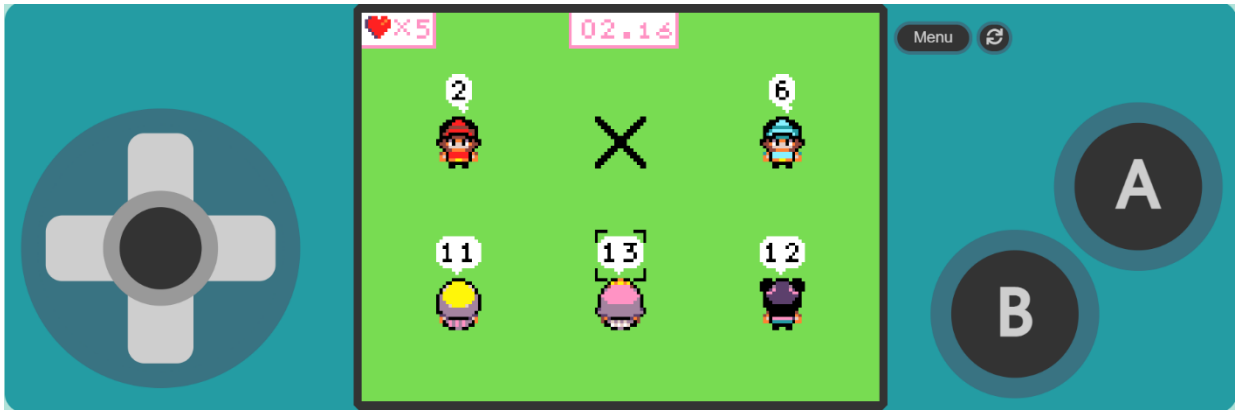
SECOND ARRAYS EXERCISE



DESCRIPTION

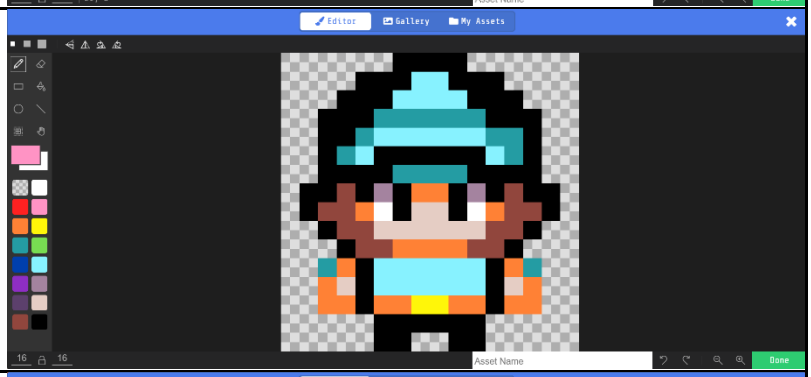
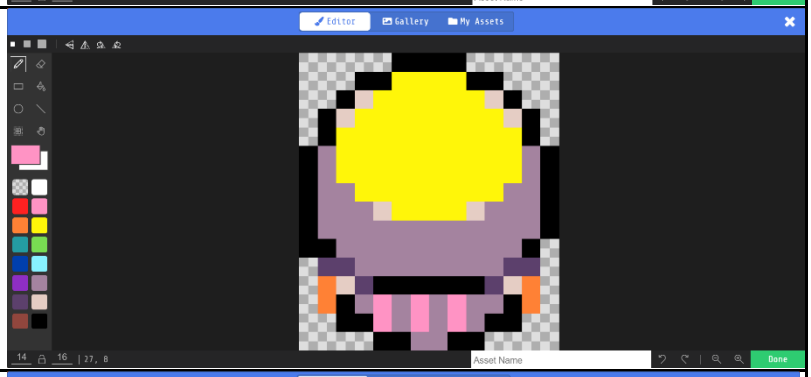
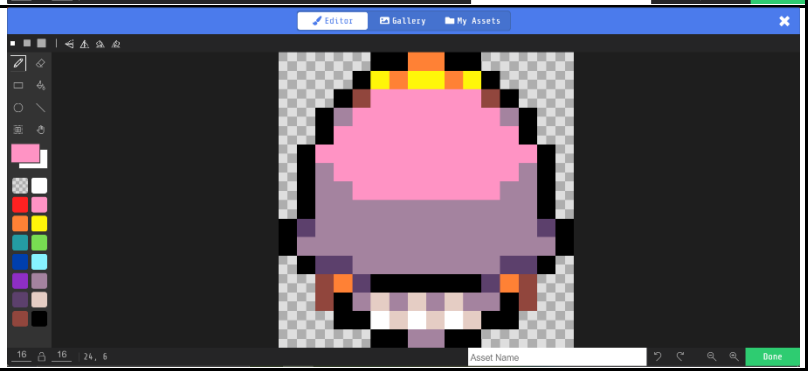
En este ejercicio crearemos un videojuego educativo de multiplicaciones.

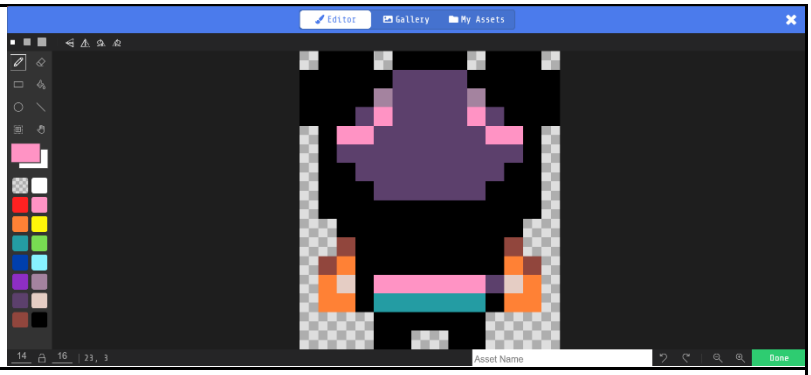
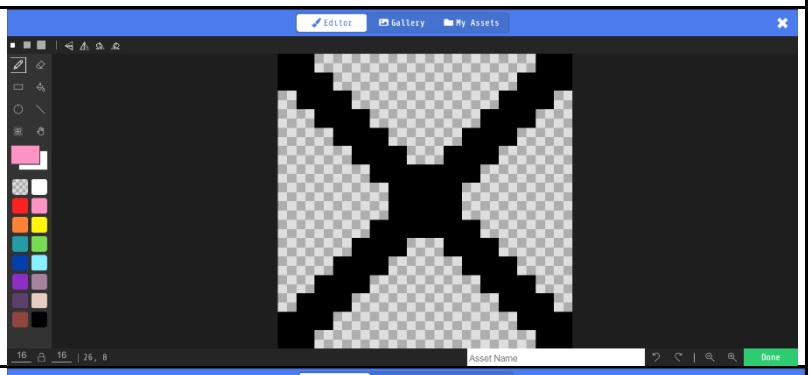
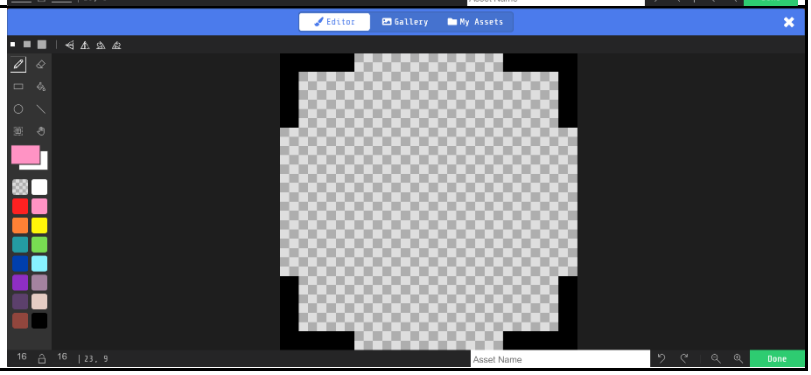
Para ello accederemos a **MakeCode Arcade** y realizaremos las operaciones necesarias.

GOALS

- Working with arrays using game control in MakeCode Arcade.
- Working with and understanding variables in MakeCode Arcade.
- Assigning a position to each game element.
- Using mathematical operations to solve problems.
- Converting numerical values to strings.
- Increasing the difficulty.

Game programming

ASSETS CREATION	
MAIN CHARACTER SPRITES CREATION	
We recommend using a 16x16 pixel grid for the character1 Sprite.	
We recommend using a 16x16 pixel grid for the character2 Sprite.	
We recommend using a 16x16 pixel grid for the Answers1 Sprite.	
We recommend using a 16x16 pixel grid for the Answers2 Sprite.	

We recommend using a 16x16 pixel grid for the Answers3 Sprite.	
ADDITIONAL SPRITES CREATION	
We recommend using a 16x16 pixel grid for the sign Sprite.	
We recommend using a 16x16 pixel grid for the Cursor Sprite.	

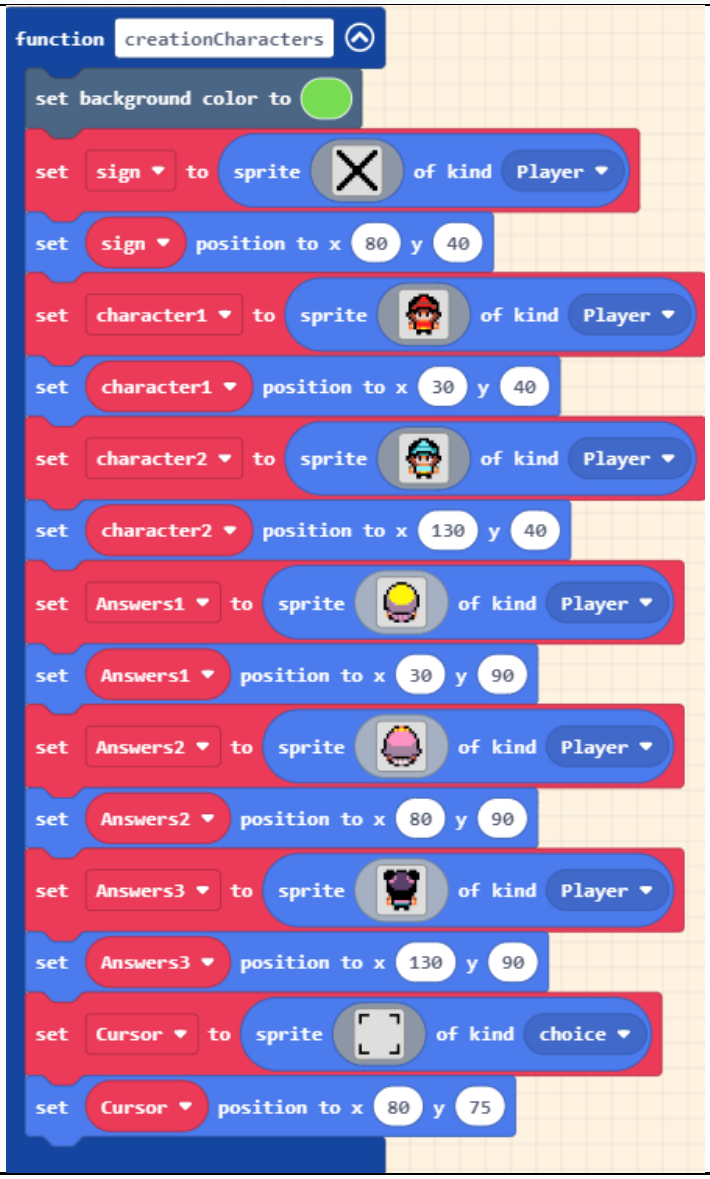
MAIN PROGRAMMING

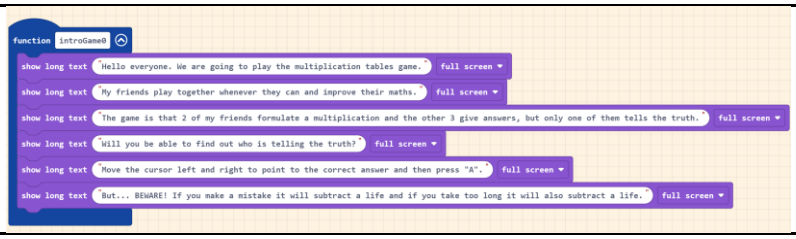
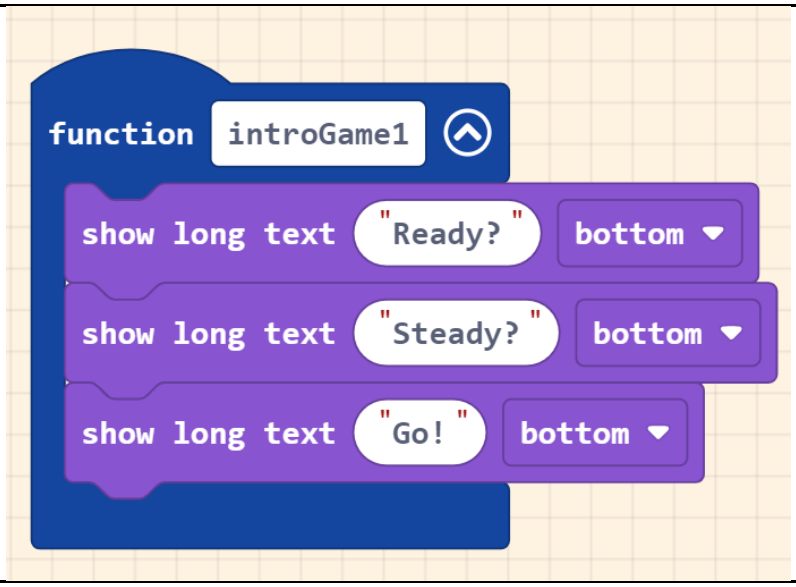
CREATE ON START

CREATE FUNCTION creationCharacters

Let's start by creating a function to set up the game's environment and elements. This function will be responsible for initializing the game, creating the necessary sprites, and setting up the initial state.

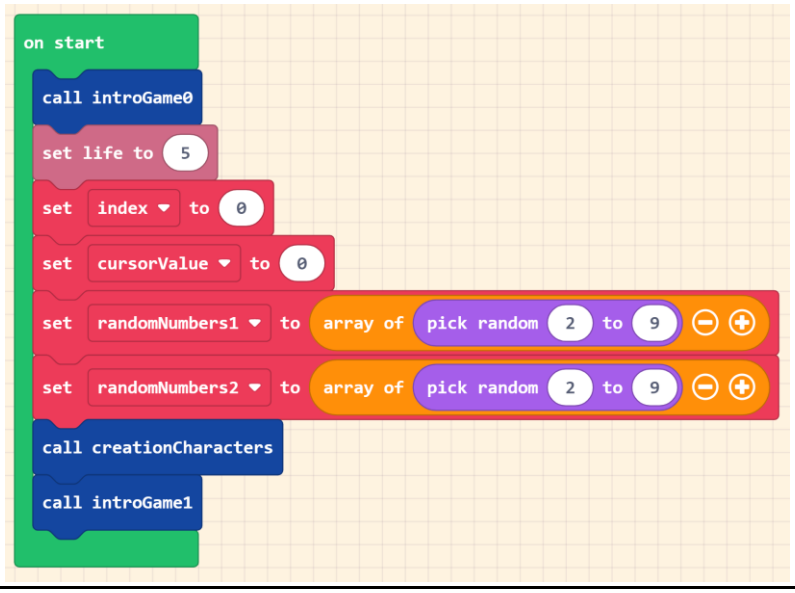
Here we will set the color of our background to green and also place all our player sprites in their positions on the stage. Finally, we will add the cursor sprite and assign it the "choice" type.



CREATE FUNCTION introGAME0	
We start by displaying a message on the screen to provide the player with information on how the game works.	
CREATE FUNCTION introGAME1	
Mostramos más mensajes de inicio de la partida	

Presentation of Elements and Initial Variable Declaration Programming

In the **on start** block, we begin by setting variables and activating functions to display a game introduction. Both **introGame** functions show messages to the player, and **creationCharacters** creates and positions all the sprites in the game. The **index** variable will be used to look up certain positions in the arrays later on. The **cursorValue** variable will control the cursor. The **randomNumbers** variables will hold random values that the player needs to guess the result of multiplying both variables.

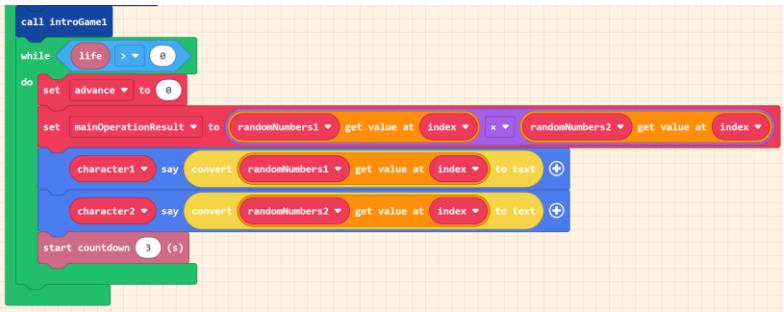


PROGRAMMING LOOP TO RESET SCREEN FOR EACH ATTEMPT

We will create a loop that runs as long as the player has remaining lives. Inside the loop, we will set certain variables, display a message for the current operation using characters, and start a countdown.

The **advance** variable will be used to check if we have selected a result.

The **mainOperationResult** variable will hold the correct result for the operation that the player needs to guess.



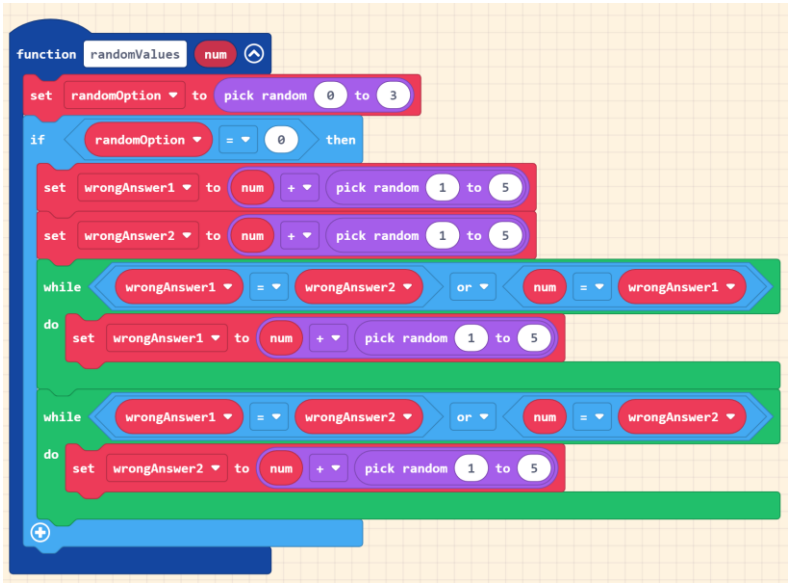
CREATE FUNCTION RANDOMVALUES

We will create the **randomValues** function with a numeric parameter. This function will generate variations in the answers, adding a certain level of randomness.

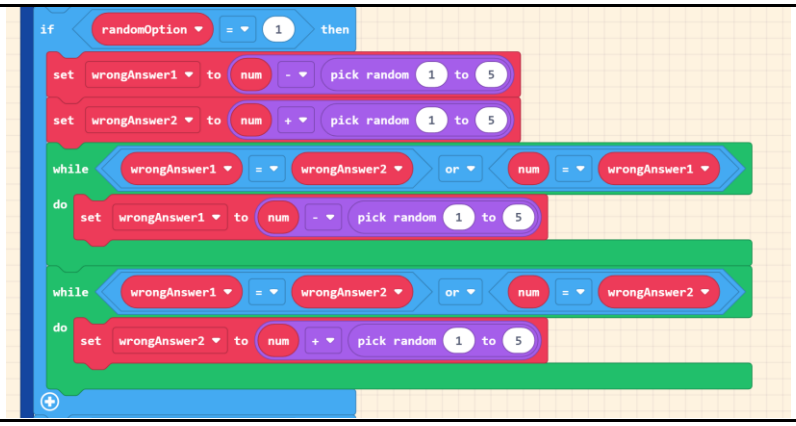
We will start by generating a random value for **randomOption**. The number of possible values for this variable will determine the patterns for the incorrect results.

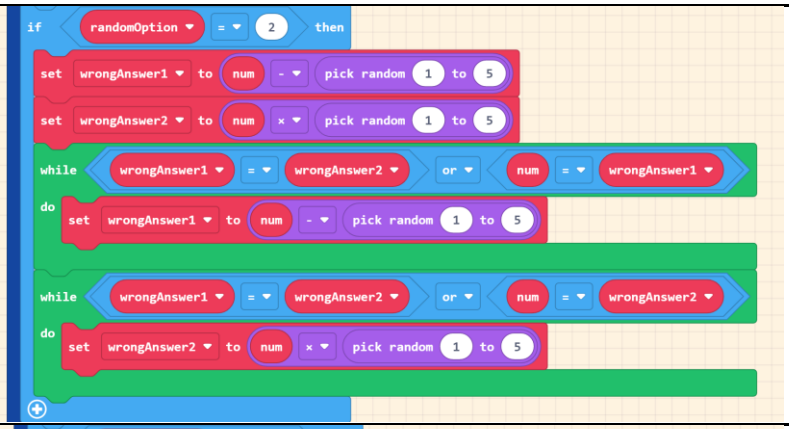
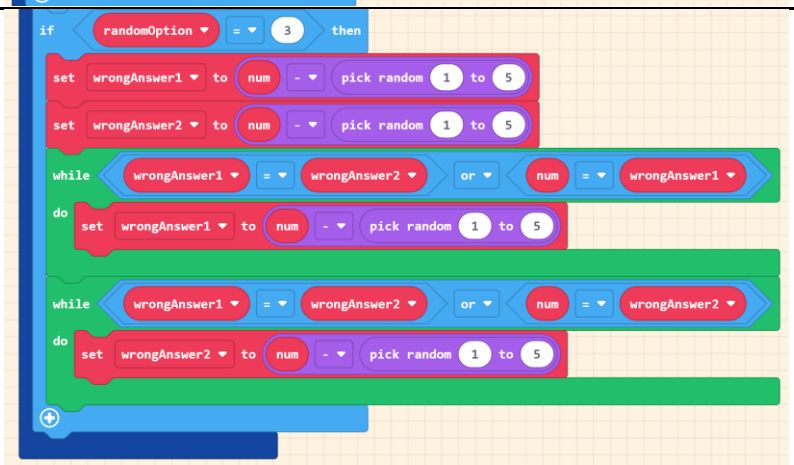
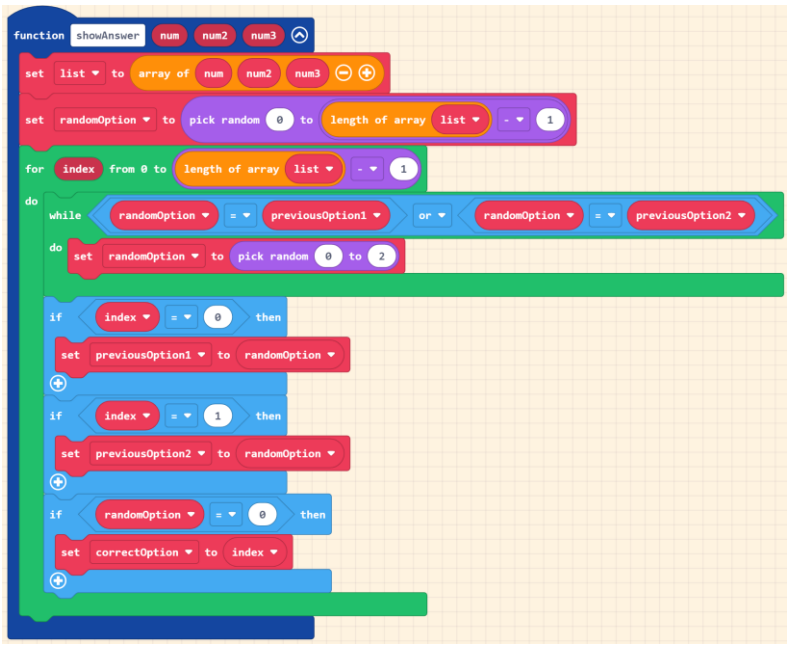
Starting with **randomOption** equal to 0, we will assign values to the **wrongAnswer** variables. These values will be the parameter value plus a random number between 1 and 5.

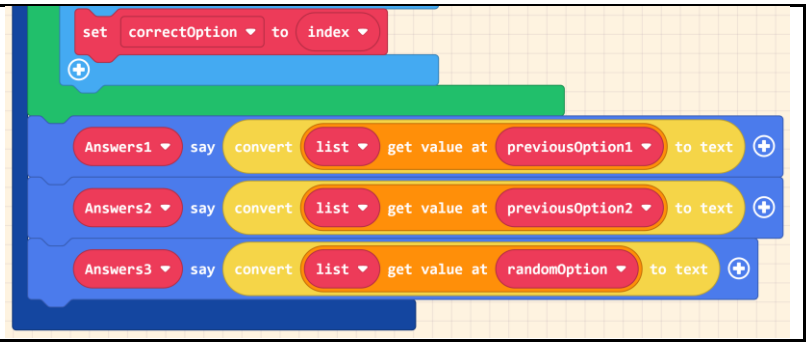
Next, we will use while loops to ensure that the options for the player to choose from do not have any duplicate numbers.



Here's an updated version where we duplicate the set of blocks under the **randomOption = 0** condition and change the values to create another possible result. In this case, we will subtract from **wrongAnswer1**.



Here's an updated version where we modify the previous group of blocks and use multiplication for **wrongAnswer2**.	
In the last variation of possible incorrect results, we will use subtraction for both **wrongAnswer1** and **wrongAnswer2**.	
CREATE FUNCTION SHOWANSWER	
We will create a function with 3 numerical parameters called **shortAnswer**. The parameters will contain the results to be chosen by the player. Within this function, we will place the correct result in different positions, filling the other spaces with incorrect results. We will create an array with the different values entered in the parameters. Then, we will create a variable that has a random value between 0 and the size of the **list** array minus one. In this case, that operation results in 2. We will use a loop to iterate through all the spaces of the **list** array. We will assign the positions to the variables, which will be displayed later.	

To conclude the function, we will program certain characters to say the options that the player can choose from.	

TO ACTIVATE THE FUNCTIONS "RANDOMVALUES" AND "SHOWANSWER"	
In "randomValues", we introduce the variable "mainOperationResult" so that the alternative options are related to it. In "showAnswer", we will introduce the variables "mainOperationResult", "wrongAnswer1", and "wrongAnswer2" in the parameters, where the position of the options that the player can choose will change.	

With this programming, characters will be created that will serve as graphical elements to display information. The values and positions of the different randomly chosen options will be set.