

Segundo ejercicio Arrays



Descripción

En este ejercicio crearemos un videojuego educativo de multiplicaciones.

Para ello accederemos a [MakeCode Arcade](#) y realizaremos las operaciones necesarias.

Objetivos de programación y diseño

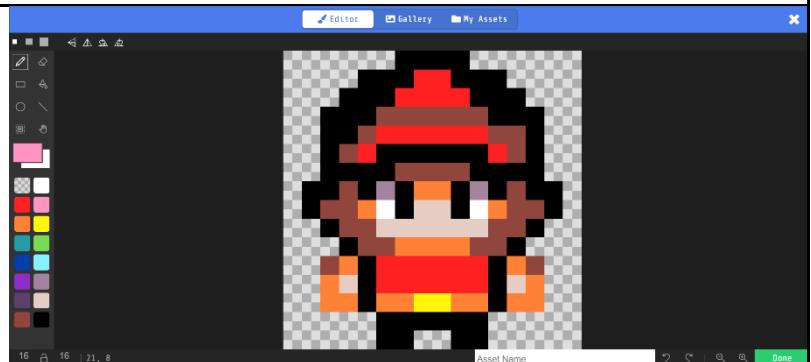
- Trabajar arrays mediante un control del juego en MakeCode Arcade.
- Trabajar y entender las variables en MakeCode Arcade.
- Asignar una posición a cada elemento del juego.
- Utilizar operaciones matemáticas para resolver problemas.
- Convertir valores numéricos en textos (string).
- Aumentar la dificultad.

Programación del juego

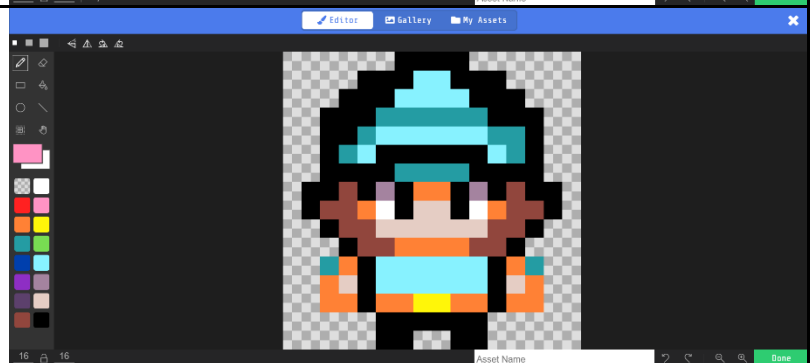
CREACIÓN DE ASSETS

CREACIÓN SPRITE PROTAGONISTAS

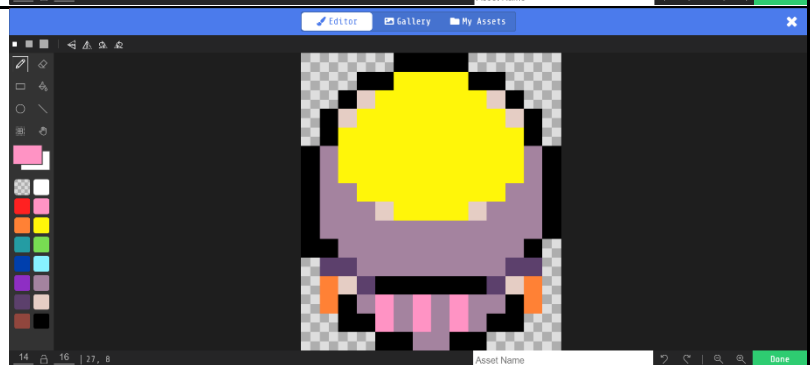
Te recomendamos utilizar una matriz de 16x16 px para el **Sprite** de **character1**.



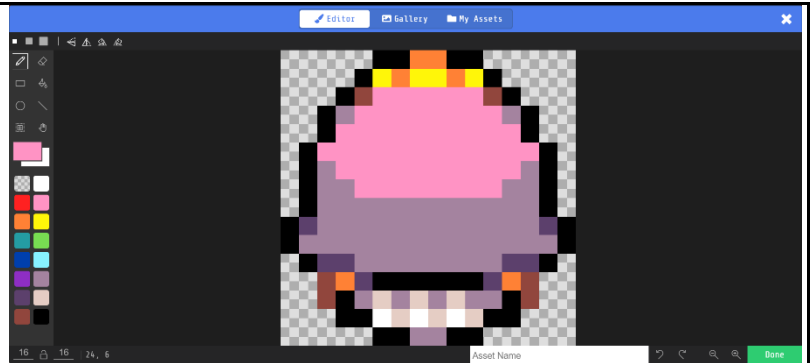
Te recomendamos utilizar una matriz de 16x16 px para el **Sprite** de **character2**.



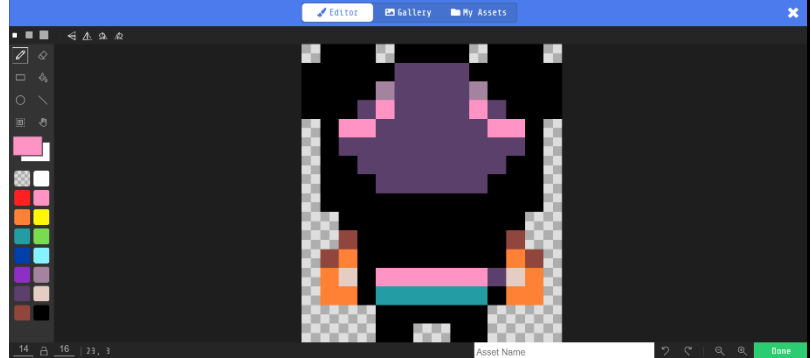
Te recomendamos utilizar una matriz de 16x16 px para el **Sprite** de **Answers1**.



Te recomendamos utilizar una matriz de 16x16 px para el **Sprite** de **Answers2**.

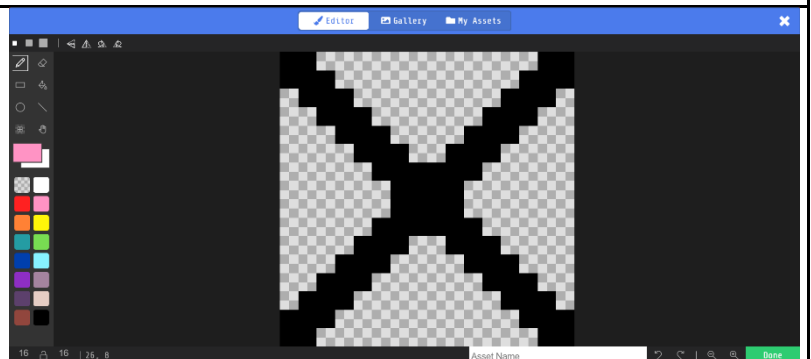


Te recomendamos utilizar una matriz de 16x16 px para el **Sprite** de **Answers3**.

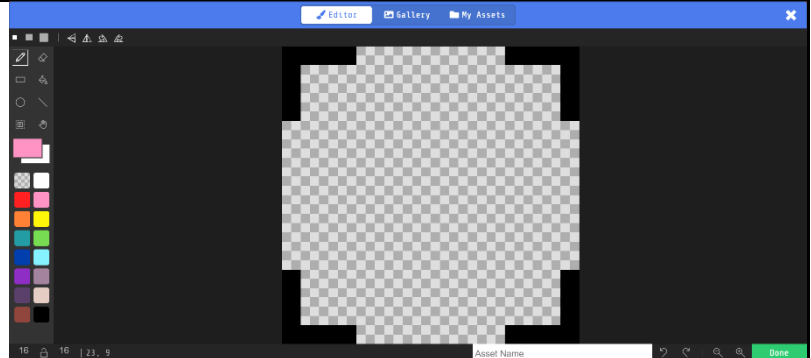


CREACIÓN SPRITES ADICIONALES

Te recomendamos utilizar una matriz de 16x16 px para el **Sprite** de **sign**.



Te recomendamos utilizar una matriz de 16x16 px para el **Sprite** de **Cursor**.

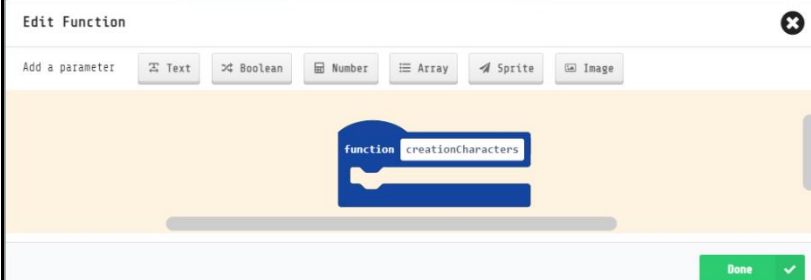


PROGRAMACIÓN PRINCIPAL

CREACIÓN INICIO DEL JUEGO

CREACIÓN FUNCIÓN creationCharacters

Este juego empieza ya a complicarse a la hora de programar. Por eso tenemos que empezar por crear una función para crear el escenario y los elementos del juego.



Aquí introduciremos que el color de nuestro escenario sea verde y también pondremos todos nuestros sprite de tipo player y los colocaremos en su posición en el escenario, por último, colocaremos el sprite cursor y lo asignaremos tipo choice.

```
function creationCharacters
  set background color to green
  set sign to sprite of kind Player
  set sign position to x 80 y 40
  set character1 to sprite of kind Player
  set character1 position to x 30 y 40
  set character2 to sprite of kind Player
  set character2 position to x 130 y 40
  set Answers1 to sprite of kind Player
  set Answers1 position to x 30 y 90
  set Answers2 to sprite of kind Player
  set Answers2 position to x 80 y 90
  set Answers3 to sprite of kind Player
  set Answers3 position to x 130 y 90
  set Cursor to sprite of kind choice
  set Cursor position to x 80 y 75
```

CREACIÓN FUNCIÓN introGAME0

Empezamos mostrando mensaje en pantalla para que el jugador tenga información de cómo funciona el juego

```
function introGame0
  show long text "Hello everyone. We are going to play the multiplication tables game." full screen
  show long text "My friends play together whenever they can and improve their maths." full screen
  show long text "The game is that 2 of my friends formulate a multiplication and the other 3 give answers, but only one of them tells the truth." full screen
  show long text "Will you be able to find out who is telling the truth?" full screen
  show long text "Move the cursor left and right to point to the correct answer and then press 'A'." full screen
  show long text "But... BEWARE! If you make a mistake it will subtract a life and if you take too long it will also subtract a life." full screen
```

CREACIÓN FUNCIÓN introGAME1

Mostramos más mensajes de inicio de la partida

```
function introGame1
  show long text "Ready?" bottom
  show long text "Steady?" bottom
  show long text "Go!" bottom
```

PROGRAMACIÓN PRESENTACIÓN DE ELEMENTOS Y DECLARACIÓN VARIABLES INICIALES

En el on start empezamos a establecer variables y activamos las funciones para mostrar una presentación del juego. En ambos introGame muestran mensajes al jugador y creationCharacters crea y coloca todos los sprites que tendrá el juego. La variable index servirá para mirar en ciertas posiciones en los arrays más adelante. La variable cursorValue servirá para tener un control sobre el cursor. Las variables randomNumbers tendrán un valor aleatorio que serán las que se muestre al jugador y adivine el resultado de multiplicar ambas variables

```

on start
  call introGame0
  set life to 5
  set index to 0
  set cursorValue to 0
  set randomNumbers1 to array of pick random 2 to 9
  set randomNumbers2 to array of pick random 2 to 9
  call creationCharacters
  call introGame1
  
```

PROGRAMAMOS BUCLE DE REINICIO DE PANTALLA PARA CADA INTENTO

Hacemos un bucle que se ejecute mientras el jugador tiene alguna vida. Dentro estableceremos ciertas variables, mostraremos un mensaje de la operación actual usando personajes e iniciamos una cuenta atrás.

La variable advance servirá para comprobar si hemos seleccionado algún resultado.

La variable mainOperationResult tendrá de valor el resultado de la operación que debe acertar el jugador.

```

call introGame1
while life > 0
do
  set advance to 0
  set mainOperationResult to randomNumbers1 get value at index * randomNumbers2 get value at index
  character1 say convert randomNumbers1 get value at index to text
  character2 say convert randomNumbers2 get value at index to text
  start countdown 3 (s)
  
```

PROGRAMACIÓN FUNCIÓN RANDOMVALUES

Crearemos la función randomValues con un parámetro numérico, esta sirve para generar las variaciones en las respuestas, que tendrán cierta aleatoriedad.

Estableceremos un valor aleatorio en randomOption. La cantidad de números posibles que tendrá esta variable servirán para establecer pautas en los distintos resultados incorrectos.

Empezando por randomOption igual a 0, hacemos que las variables wrongAnswer tengan el valor introducido en el parámetro de la función más una suma entre 1 y 5 de manera aleatoria.

Después usaremos bucles while para comprobar que no hay números iguales en las distintas opciones a escoger por el jugador.

```
function randomValues num
  set randomOption to pick random 0 to 3
  if randomOption = 0 then
    set wrongAnswer1 to num + pick random 1 to 5
    set wrongAnswer2 to num + pick random 1 to 5
    while wrongAnswer1 = wrongAnswer2 or num = wrongAnswer1
      do set wrongAnswer1 to num + pick random 1 to 5
    while wrongAnswer1 = wrongAnswer2 or num = wrongAnswer2
      do set wrongAnswer2 to num + pick random 1 to 5
```

Podremos duplicar el conjunto de bloques recogidos en “randomOption = 0” y cambiar valores para adecuarlo a otro posible resultado.

En este caso se cambia wrongAnswer1 con una resta

```
if randomOption = 1 then
  set wrongAnswer1 to num - pick random 1 to 5
  set wrongAnswer2 to num + pick random 1 to 5
  while wrongAnswer1 = wrongAnswer2 or num = wrongAnswer1
    do set wrongAnswer1 to num - pick random 1 to 5
  while wrongAnswer1 = wrongAnswer2 or num = wrongAnswer2
    do set wrongAnswer2 to num + pick random 1 to 5
```

Lo haremos de nuevo, pero ahora, respecto al anterior grupo de bloques, pondremos una multiplicación para wrongAnswer2

```

if randomOption = 2 then
  set wrongAnswer1 to num - pick random 1 to 5
  set wrongAnswer2 to num x pick random 1 to 5
  while wrongAnswer1 = wrongAnswer2 or num = wrongAnswer1
  do set wrongAnswer1 to num - pick random 1 to 5
  while wrongAnswer1 = wrongAnswer2 or num = wrongAnswer2
  do set wrongAnswer2 to num x pick random 1 to 5
  
```

En la última variación de posibles resultados incorrectos pondremos en ambas una resta

```

if randomOption = 3 then
  set wrongAnswer1 to num - pick random 1 to 5
  set wrongAnswer2 to num - pick random 1 to 5
  while wrongAnswer1 = wrongAnswer2 or num = wrongAnswer1
  do set wrongAnswer1 to num - pick random 1 to 5
  while wrongAnswer1 = wrongAnswer2 or num = wrongAnswer2
  do set wrongAnswer2 to num - pick random 1 to 5
  
```

PROGRAMACIÓN FUNCIÓN SHOWANSWER

Crearemos una función con 3 parámetros numéricos llamado showAnswer.

En los parámetros se introducirán los resultados a escoger por el jugador. Dentro de esta función se irá ubicando las distintas posiciones el resultado correcta, rellenado los otros espacios con los incorrectos.

Crearemos un array con los distintos valores introducidos en los parámetros.

Se creará una variable que tenga un valor aleatorio entre 0 y el tamaño del array list restando uno, en este caso, esa operación da de resultado 2.

Usaremos un bucle para que recorra todos los espacios del array list.

Se asignará en las variables las posiciones que se mostrarán más adelante

```
function showAnswer: num num2 num3
  set list to array of num num2 num3
  set randomOption to pick random 0 to length of array list - 1
  for index from 0 to length of array list - 1
  do
    while randomOption = previousOption1 or randomOption = previousOption2
    do
      set randomOption to pick random 0 to 2
    end while
    if index = 0 then
      set previousOption1 to randomOption
    end if
    if index = 1 then
      set previousOption2 to randomOption
    end if
    if randomOption = 0 then
      set correctOption to index
    end if
  end for
```

Para terminar con la función, programaremos a ciertos personajes para que digan las opciones que podrá escoger el jugador

```
set correctOption to index
Answers1 say convert list get value at previousOption1 to text
Answers2 say convert list get value at previousOption2 to text
Answers3 say convert list get value at randomOption to text
```

ACTIVAR FUNCIONES RANDOMVALUES Y SHOWANSWER

Llamaremos a las funciones “randomValues” y “showAnswer”.
 En “randomValues” introducimos la variables “mainOperationResult” para que las opciones alternativas tengan relación a esta.
 En “showAnswer” introduciremos en los parámetros las variables “mainOperationResult”, “wrongAnswer1” y “wrongAnswer2” donde cambiarán la posición de las opciones que podrá escoger el jugador



Con esta programación, se crearán los personajes que servirán como elementos gráficos para mostrar información, se establecerán los valores y posición de las distintas opciones a elegir de manera aleatoria.