**Description**

LED Basket is a simple game built on the Microbit platform. The objective of the game is to catch falling LED using a virtual basket controlled by the player. The game is designed to be played on a Microbit device with buttons for control.

**Features**

- Catch falling LED by moving the basket left and right.

- Increase your score by catching more LED's.

- Game speed increases with points

- Player has 3 lives and the game ends when you miss catching an LED 3 times.

**How to Play**

1. Game Autoload on start of MicroBit.

2. Use the A button to move the basket left and the B button to move it right.

3. Try to catch as many LED's as possible to score points.

4. The game ends when you miss catching 3 LED's.

5. The game can be reset at any time by pressing A and B together.

**Code block**

from microbit import \*

import random

import music

basket\_x = 2 #setting global vaiable for basket x position

#basket movement

def basket\_movement(direction):

global basket\_x

x\_pos = basket\_x + direction

if 0 <= x\_pos <= 4:

display.set\_pixel(basket\_x,4,0) #switch off previous basket position

basket\_x = x\_pos

display.set\_pixel(basket\_x,4,9) #switch on basket position

#functions to show and hide LED drops

def show\_led(led\_x, led\_y):

display.set\_pixel(led\_x, led\_y, 9)

def hide\_led(led\_x, led\_y):

display.set\_pixel(led\_x, led\_y, 0)

#to check if LED is caught by the basket

def check\_catch(led\_x, led\_y):

return led\_x == basket\_x and led\_y == 4

def Main():

display.scroll("HI!")

music.play(music.POWER\_UP)

score = 0

life = 3

while life > 0:

led\_x = random.randint(0,4)

led\_y = 0

show\_led(led\_x, led\_y)

display.set\_pixel(basket\_x,4,9)

while led\_y < 4:

if button\_a.was\_pressed():

basket\_movement(-1)

if button\_b.was\_pressed():

basket\_movement(1)

if button\_a.is\_pressed() and button\_b.is\_pressed():

reset() # game resetter

hide\_led(led\_x,led\_y)

led\_y += 1

show\_led(led\_x,led\_y)

if check\_catch(led\_x, led\_y):

music.play(music.BA\_DING)

score += 1

hide\_led(led\_x, led\_y)

break

if led\_y == 4:

life -= 1

if life > 0:

music.play(music.POWER\_DOWN)

display.scroll("lives: " + str(life))

# Adjust game difficulty (speed) based on the score

sleep(1000 - (score \* 50))

music.play(music.FUNERAL)

display.scroll("GG")

display.scroll("score: " + str(score))

reset()

while True:

Main()