### Snake

Visit:<https://www.microbit.co.uk/app/> and then click on `Create Code` and `MicroPython`.

Use <https://create.withcode.uk> to emulate a BBC Microbit

1. Import all functions from module `microbit` and `random`.

from microbit import \*

import random

2. Create a snake which is 2 cells long and facing down

class Snake():

def \_\_init\_\_(self):

self.length = 2

self.direction = "down"

self.head = (2, 2)

self.tail = []

3. To draw the snake we need to define a new function called draw. The parameter self in draw(self) means it is to draw itself.

class Snake():

def \_\_init\_\_(self):

self.length = 2

self.direction = "down"

self.head = (2, 2)

self.tail = []

# Draw snake head

def draw(self):

# draw head

display.set\_pixel(self.head[0], self.head[1], 9)

4. Now let’s create a class called Game that moves the snake, which then creates the snake and displays it.

class Game():

# Create the snake

def \_\_init\_\_(self):

self.player = Snake()

# Draw the snake

def draw(self):

display.clear()

self.player.draw()

5. Let’s create a main game loop which draws the snake and sleeps for 0.5 seconds.

This will create the head of the snake.

game = Game()

# main game loop

while True:

game.draw()

sleep(500)

6. Let’s now add functions in the Snake class to make it move (left or right). We need to know where the current position of the head of the snake is, and then we move left by 1 pixel, so `self.head[0] - 1` to move to left and `self.head[0] + 1` to move to right.

# Snake class

def move(self):

if self.direction == "left":

self.head = ((self.head[0] - 1) % 5, self.head[1])

elif self.direction == "right":

self.head = ((self.head[0] + 1) % 5, self.head[1])

7. To handle the input of “left” or “right” in the game, we need to add a function called handle\_input(self) in the Game class

# Game class

def handle\_input(self):

# change direction? (no reversing)

if button\_a.is\_pressed():

if self.player.direction != "right":

self.player.direction = "left"

elif button\_b.is\_pressed():

if self.player.direction != "left":

self.player.direction = "right"

def update(self):

# move snake

self.player.move()

8. Modify the main game loop so that it looks like this to handle inputs

# main game loop

while True:

game.handle\_input()  
 game.update()

game.draw()

sleep(500)