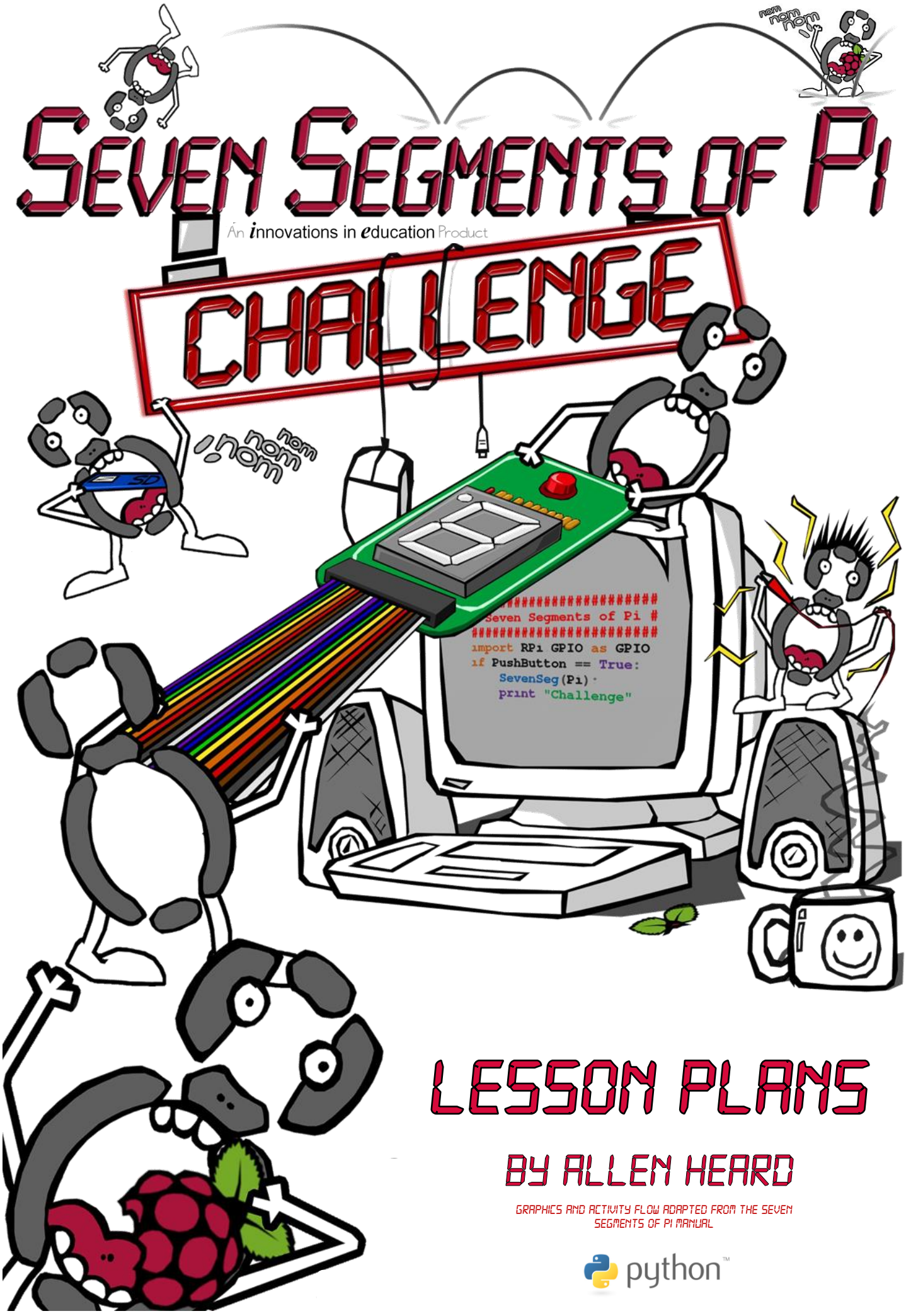
# Seven Segments of Pi

An innovations in education Product

## CHALLENGE



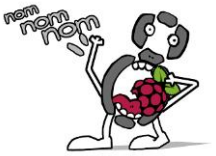

# LESSON PLANS

BY ALLEN HEARD

GRAPHICS AND ACTIVITY FLOW ADAPTED FROM THE SEVEN SEGMENTS OF PI MANUAL

python™

# SEVEN SEGMENTS OF PI
# LESSON 1




## STARTER

Watch the Digital Clock video and think about the following questions:

- What do you notice about how the numbers are made up on the clock?
- As the clock counts upwards, what do you think happens “in the program” to make them change?

## LESSON OBJECTIVES

By the end of this lesson

- Students will understand the make-up of seven segment digits.
- All pupils will be able to communicate a given number in the format of a seven segment display.
- Pupils will successfully code this into Python.

## TASK 1:

Show pupils this image of the Seven Segments of Pi board and explain that in order to make it display numbers we have to code them into Python by using GPIO commands that illuminate individual sections of the device. Pupils should be given this basic file as a starting point, the comments highlight what each piece of code does.

Upon running this program, students will see that it displays the number 1 for a second and then goes blank. The first task is for pupils to code the functions for digits 2 and 3 correctly, test, debug if necessary and fix. Advise pupils to comment any code they add.

**Note: GPIO access needs root privileges, either login as root or run with sudo from LX Terminal (sudo python filename.py)**

## TASK 2:

Next, pupils must add three more functions (four, five, and six) to the code to correctly display the next three.

They must then save and test this code. Did it work? Why not?
(There is no code set up in the ‘SevenSeg’ function to run the display for those numbers, there is also no calling of them in the while loop.)

## TASK 3:

Pupils need to add to the code, refactoring where required, to get the counter successfully counting up to six. Save this version as counter.py

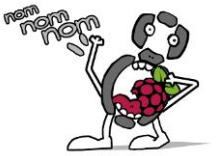
## PLENARY

Now students have had the chance to code a working ‘counter’, show the starter video again and ask pupils to think about the code required to do split seconds.

## HOMEWORK

What line would students need to change in their program to implement split-second timing?
(Obviously not actually able to do this on a Seven Segments of Pi, but the idea here is to get students thinking about sequence and timing)

# SEVEN SEGMENTS OF PI
# LESSON 2




## STARTER

Watch this video of a Space Shuttle Launch. Explaining counting up is great, but counting down is even more exciting.

In lesson 1, code was generated to display from 1 to 6 on the Seven Segments of Pi with a brief delay in between. In this lesson students will create a working countdown timer with a 'lift off' sound played when the counter reaches zero.

## LESSON OBJECTIVES

In this lesson students will:

- Modify their counter.py code to make it count *down* from 6 to 0.
- Add a sound that plays when the counter reaches 0.

## TASK 1:

Students should change the code from counter.py so that instead of counting from 1 up to 6, when the button is pressed it now counts down from 6 to 0. This will require adding another function to correctly display the zero digit and also adding the required code in the SevenSeg function. Save this file as countdown.py

## TASK 2:

Students should now add sound to their countdown so that when it reaches zero, it plays an appropriate sound file. Students can download an appropriate .wav file or use this one. Remember: sound files should be in the same directory as the .py file to play successfully.

The code required to make a sound play in Python is below, obviously the filename will need to be changed if students download their own. They will need to place parts of this code in the right place within their program. Remember those comments! Save it as liftOff.py

```
import pygame # Imports the pygame package to allow sounds to be generated
pygame.mixer.pre_init(44100,-16,2,512) # Sets up correct parameters (Freq,size,channels,buffer)
pygame.init() # Initialises the pygame package
pygame.mixer.music.load("LiftOff.wav") # Load the sound file LiftOff.wav
pygame.mixer.music.play() # Plays the sound file
```

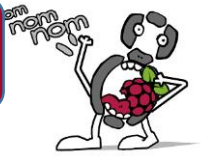
## PLENARY

Give groups of 3 or 4 pupils a dice and ask them to roll it several times. Can they predict what number will appear next? Obviously not as it's completely random and in no order unlike their program. How might we code this in Python to produce a dice using the code we have made already? This will lead to their homework.

## HOMEWORK

Pupils are to research random number generation in Python (and practise).

# SEVEN SEGMENTS OF PI
# LESSON 3




## STARTER

In this lesson, students will modify their liftOff.py program from the last lesson and change it into a working electronic dice.

## LESSON OBJECTIVES

In this lesson students will:

- Code a blank digit that blanks the display when the program is run and to act as a reset for the start of a new game when the button is pressed.
- Add a random number feature. (Using homework set last lesson.)

## TASK 1:

Students need to edit the zero function and change it to a function to 'blank' the seven segments of pi. The display must be blank when the program starts and when the user presses the button.

## TASK 2:

Using the homework from last time, students now need to add a random number feature to countdown.py. Once the countdown reaches one, the program should wait 1 second and display a random number between 1 and 6.

When students use the ***random.randint(1, 6)*** line, you could explain that any whole random number could be generated by changing the lower and upper values. In completing this task students should make sure they create the random number before they need to use it to display a number after the counter reaches 1. They will also need to assign this to a variable. They could also change the delay to speed up the dice. Save this as piDice.py

## YOU NOW HAVE AN ELECTRONIC PI DICE!

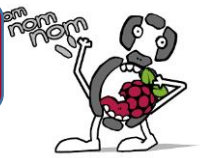
## PLENARY

Ask students to think about when they actually roll a dice. Does the dice show 6, 5, 4, 3, 2, and 1 in that order before flipping to the number it ends up landing on? Of course it doesn't! What needs to change in their piDice program in order to make it more realistic?

## HOMEWORK

Taking a printout/photo of their code, can students annotate what they would change and why to make it more realistic?

# SEVEN SEGMENTS OF PI
# LESSON 4




## STARTER

Show this video of the Seven Segments of Pi running a working piDice program. Ask students what the difference is between what they have coded and this one. Using their homework from the last lesson as a guide, ask students to try and explain what needs to change in their code to make this happen for them.

## LESSON OBJECTIVES

In this lesson students will:

- Make their PiDice program output more realistic by modifying the time delay and displaying random numbers as it 'rolls'.
- Add a sound file to announce the number displayed when the electronic dice finally comes to rest.

## TASK 1:

In this task, students need to edit the code in their piDice program to make it more realistic. They probably have something like this now (minus all the sounds – see next task for that). They now need to remove the section that tells SevenSeg what order to display numbers in before settling on a final number and change it so that it's more random, like the reality of throwing a dice. The starter video shows the sequence they are aiming for. They will also need to modify the delay so that the numbers displayed are in quick succession.

## TASK 2:

Remember that lift off sound? In this task, students should replace that sound with a sound that will speak the number finally 'rolled'. The sound files needed are provided here. Students could code this in two ways, either by using an if statement to test the number selected and play an appropriate sound file, or using 1 function and string concatenation that generates a filename to be played from the random number, as shown in the working file here. Remember: Sound files need to be in the same directory as the main Python file in order to work. This video shows what they are aiming to achieve. Save the file is piDiceV2.py

## PLENARY

Now students have created their very own piDice including sound, give students this flyer and instructions and watch this video.

## HOMEWORK

Students need to complete steps 1 to 3 and complete the project over a number of lessons.