

MIT AI2 204

IoT with MIT App Inventor

Fundamental

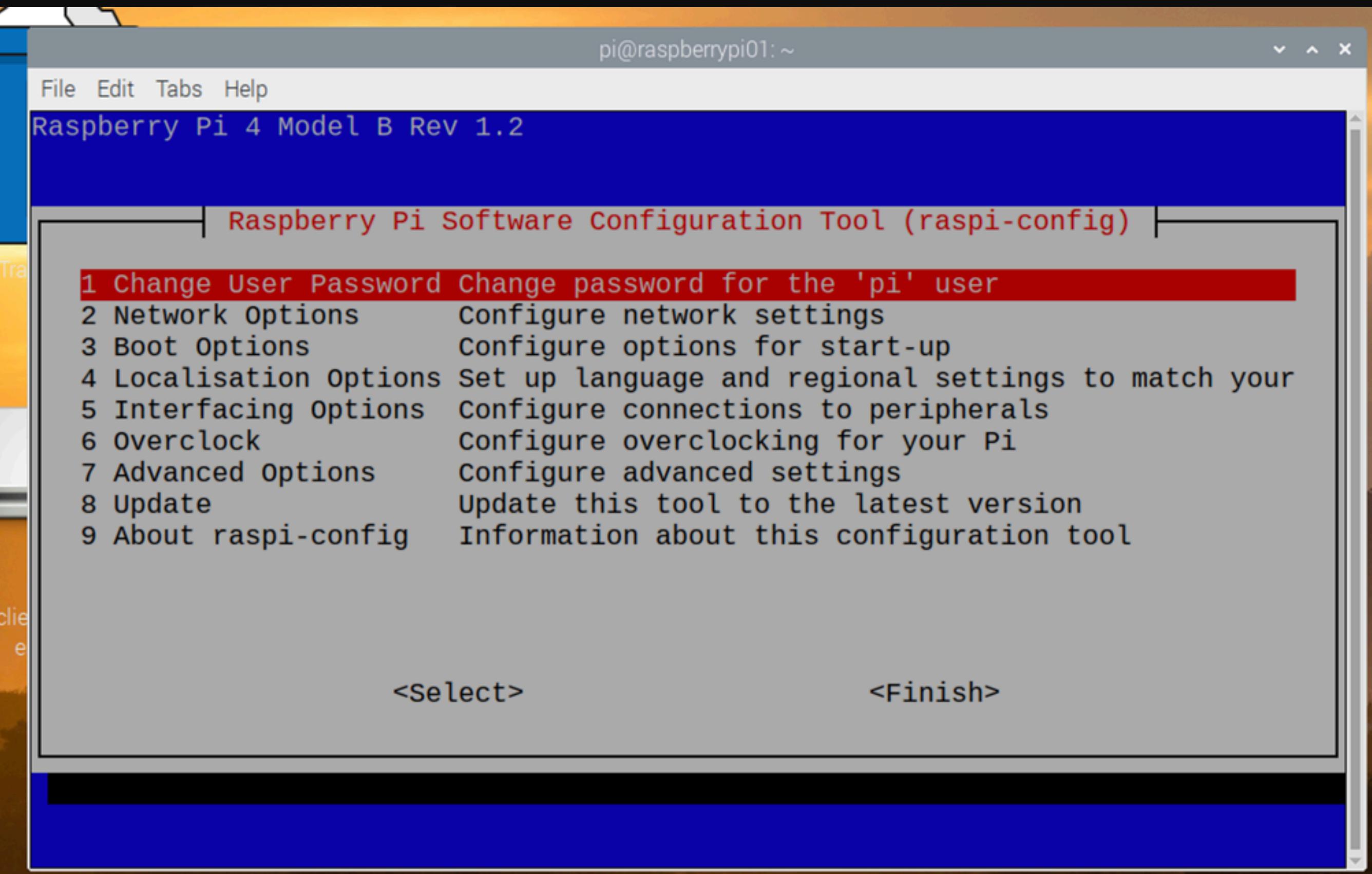
X. Tang

Remote Access -Setup

- \$ sudo raspi-config
- config P2 SSH and P3 VNC to YES under Interface Options
- Reboot
- SSH access. Windows, download Putty. [Https://www.putty.org/](https://www.putty.org/)
- MAC, SSH pi@IP
- Windows, download <https://www.tightvnc.com/download.php>
- MAC, app store, windows remote desktop

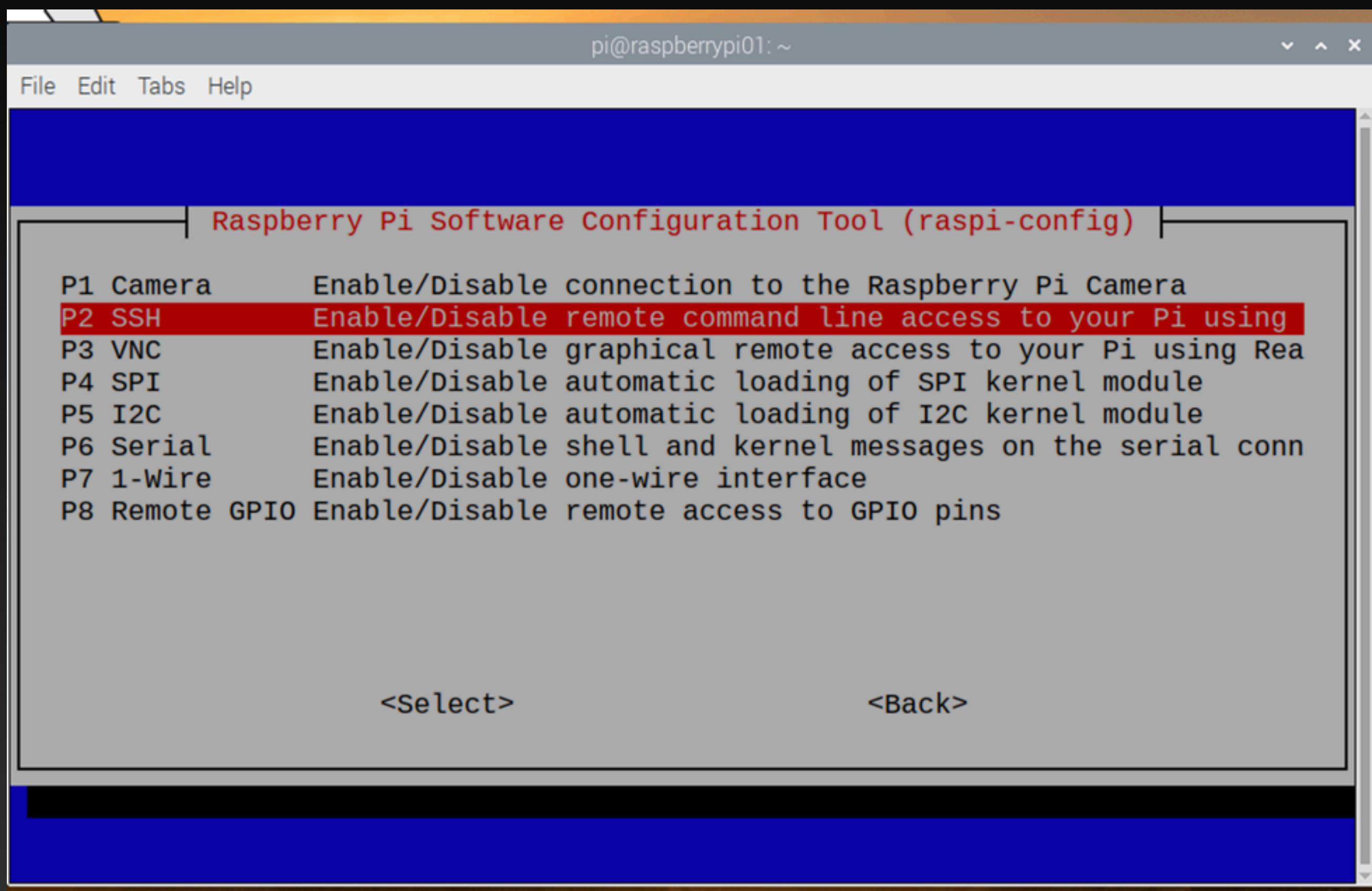
Remote Access -Setup

- \$ sudo raspi-config
- Go to Interfacing Options



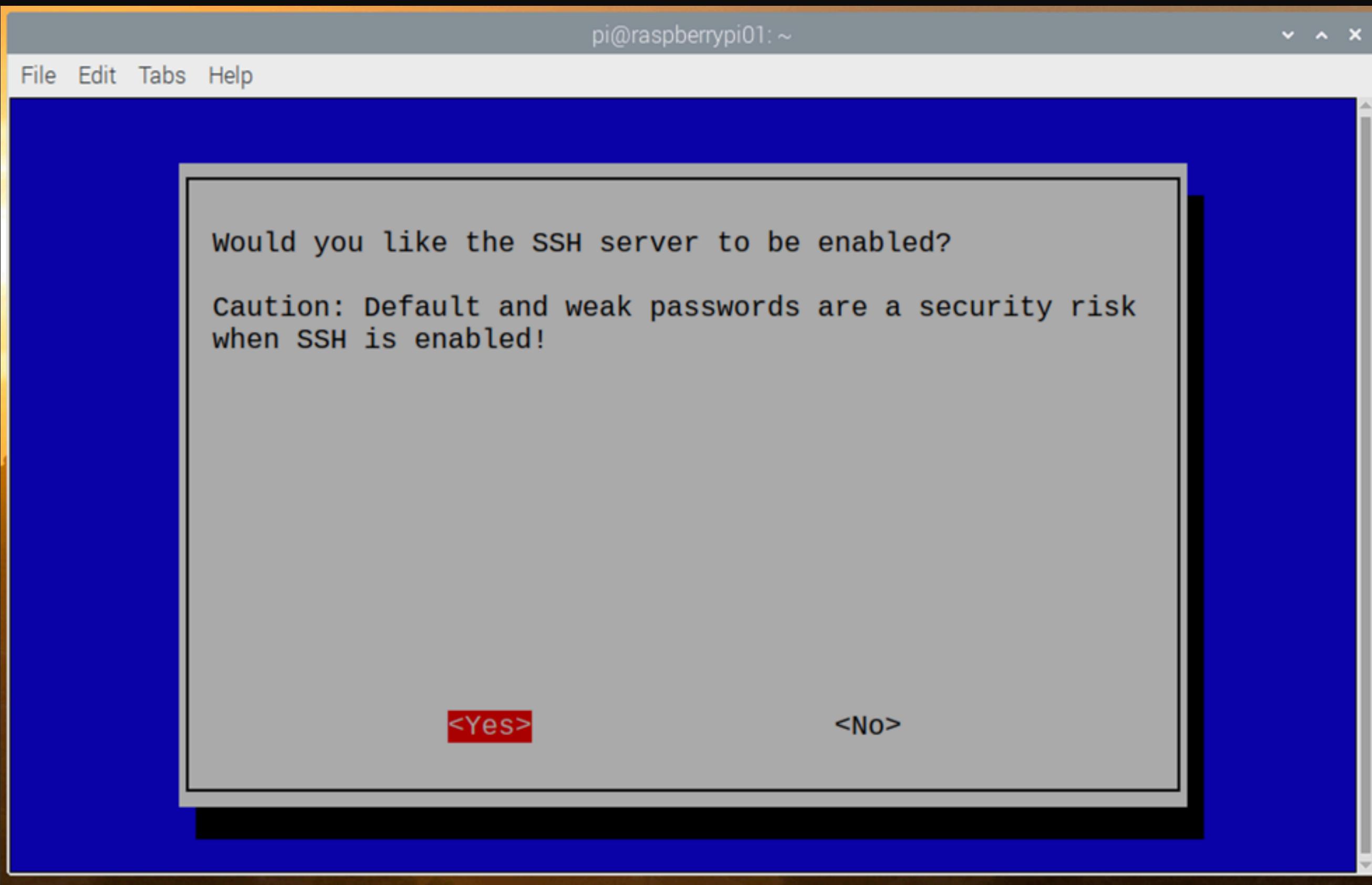
Remote Access -Setup

- Choose P2 SSH, hit enter



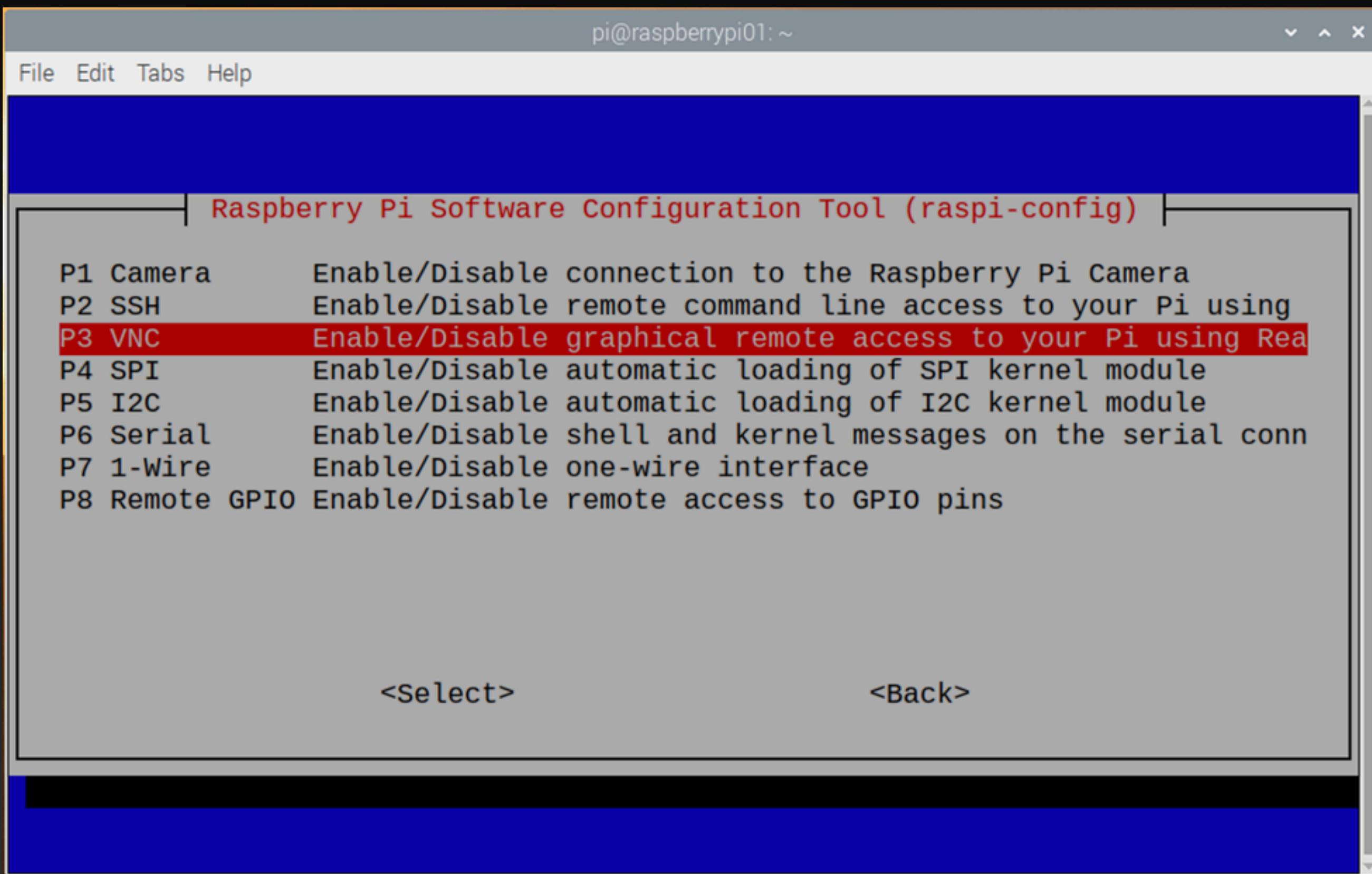
Remote Access -Setup

- Select Yes, hit enter



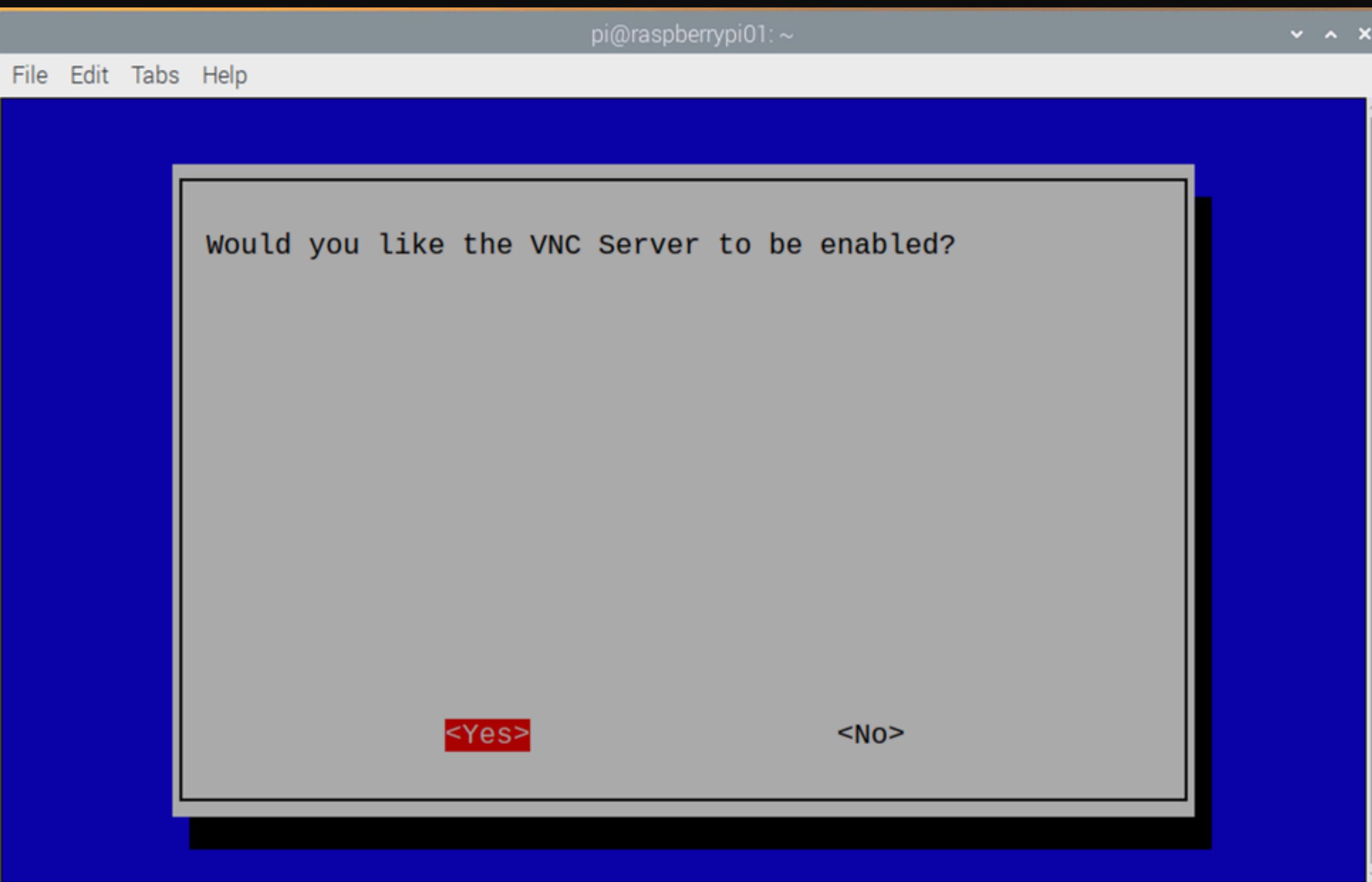
Remote Access -Setup

- Choose P3 VNC, hit enter



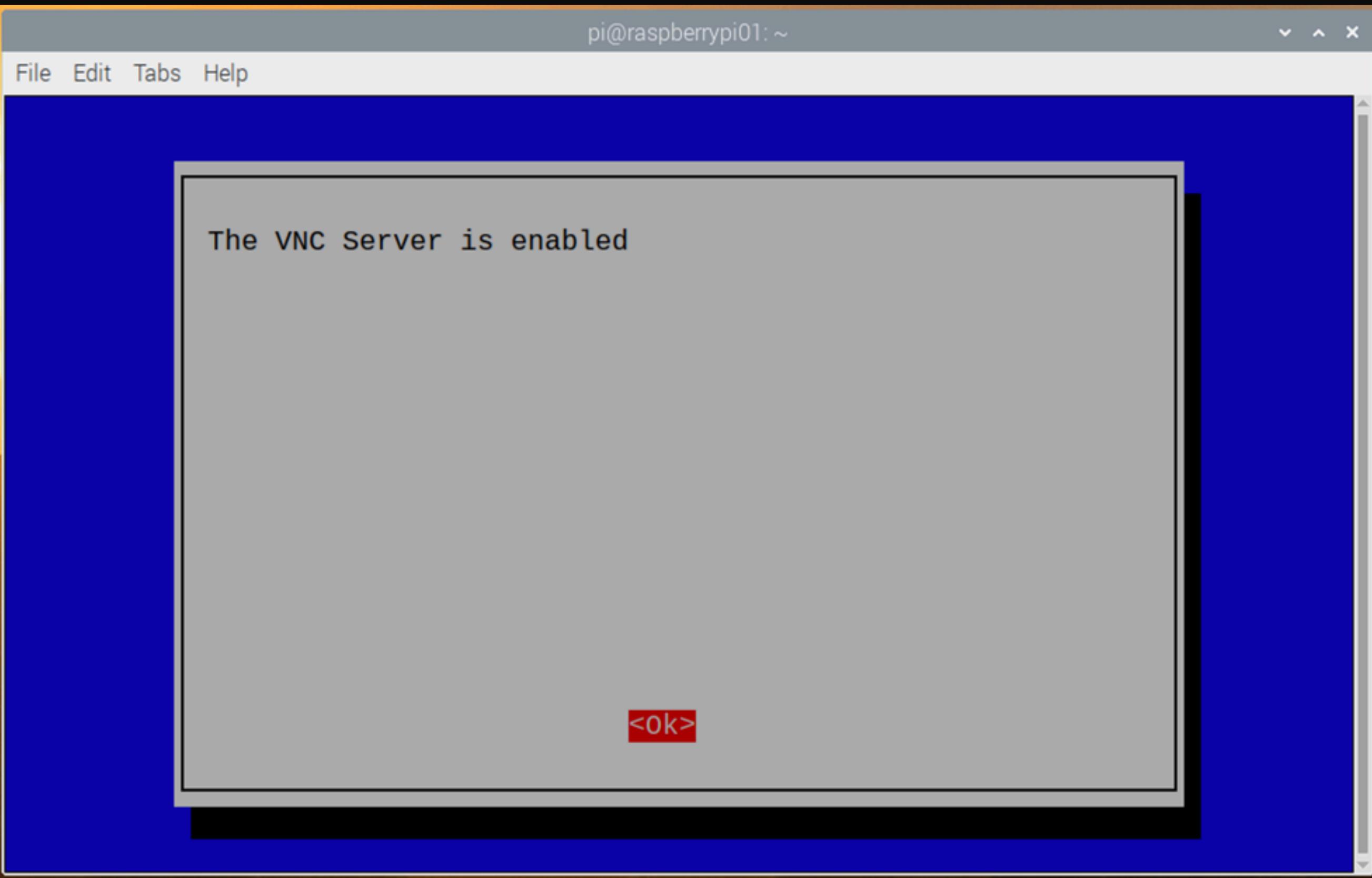
Remote Access -Setup

- Select Yes, hit enter



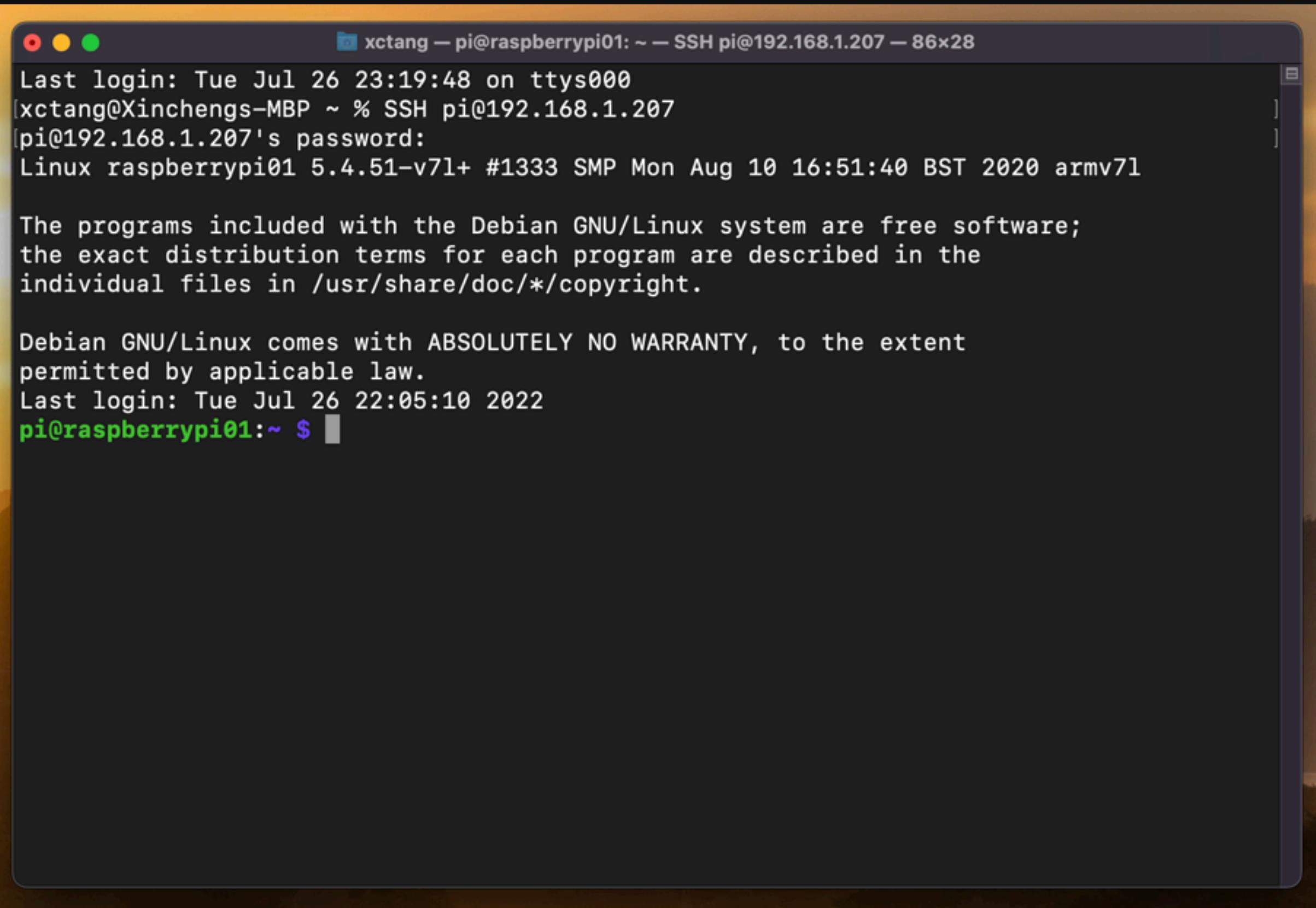
Remote Access -Setup

- You will see screen which confirm VNC is enabled



Remote Access-SSH

- SSH on MAC



A screenshot of a macOS terminal window titled "xctang — pi@raspberrypi01: ~ — SSH pi@192.168.1.207 — 86x28". The window shows the following text:

```
Last login: Tue Jul 26 23:19:48 on ttys000
[xctang@Xinchengs-MBP ~ % SSH pi@192.168.1.207
[pi@192.168.1.207's password:
Linux raspberrypi01 5.4.51-v7l+ #1333 SMP Mon Aug 10 16:51:40 BST 2020 armv7l

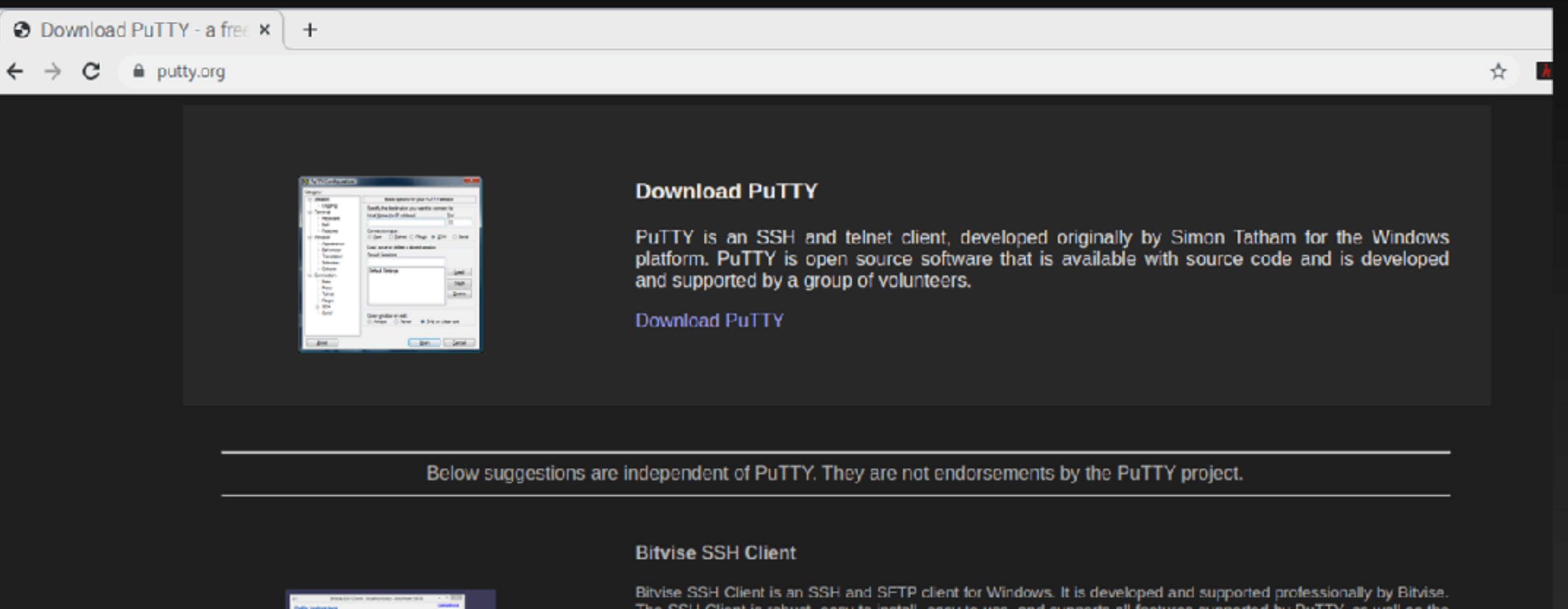
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jul 26 22:05:10 2022
pi@raspberrypi01:~ $
```

Remote Access-SSH

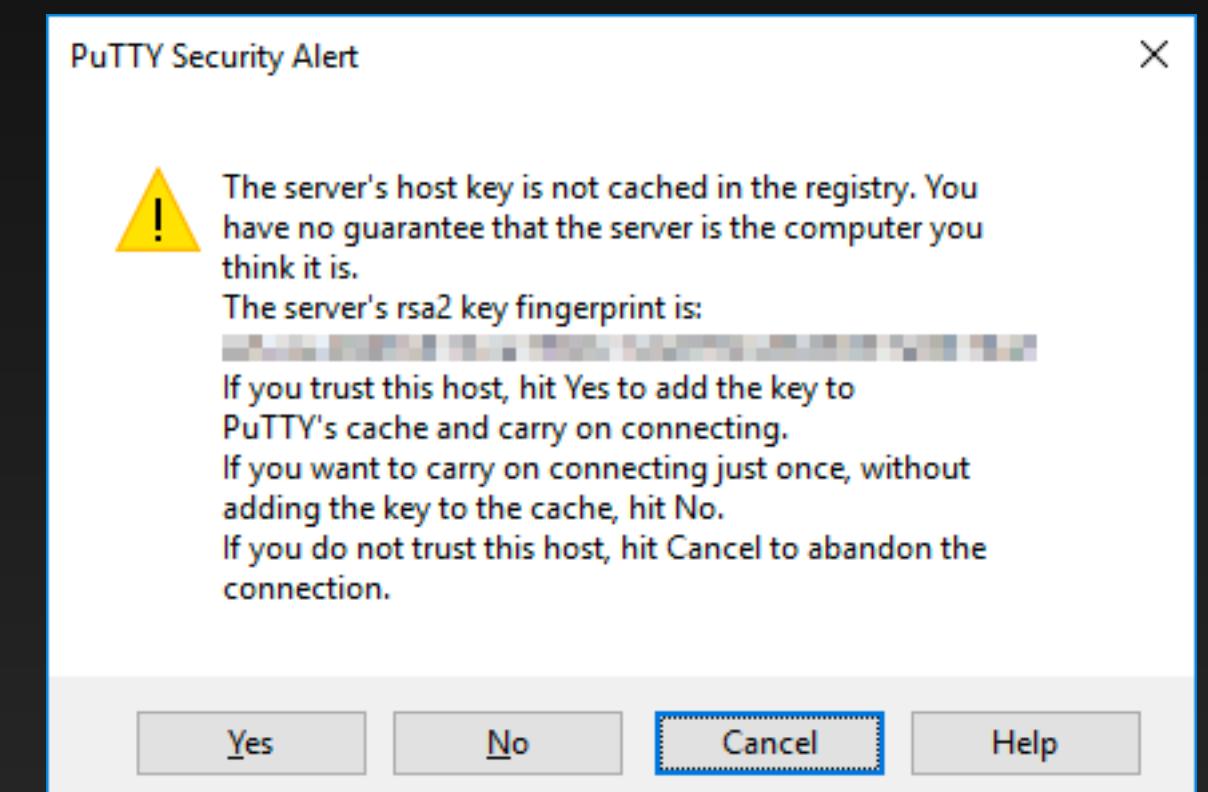
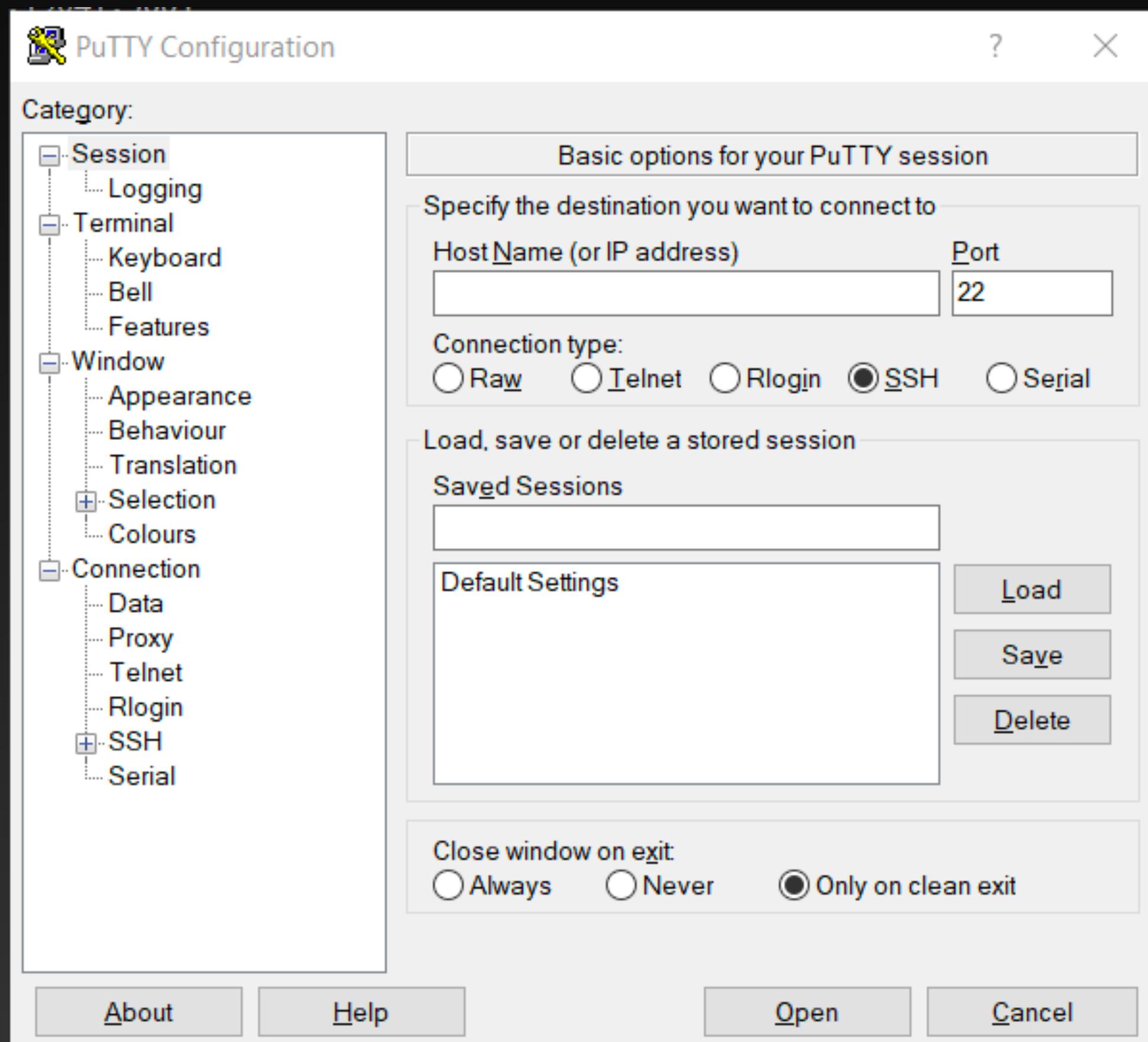
- SSH on Window Machine
- download Putty

[Https://www.putty.org/](https://www.putty.org/)



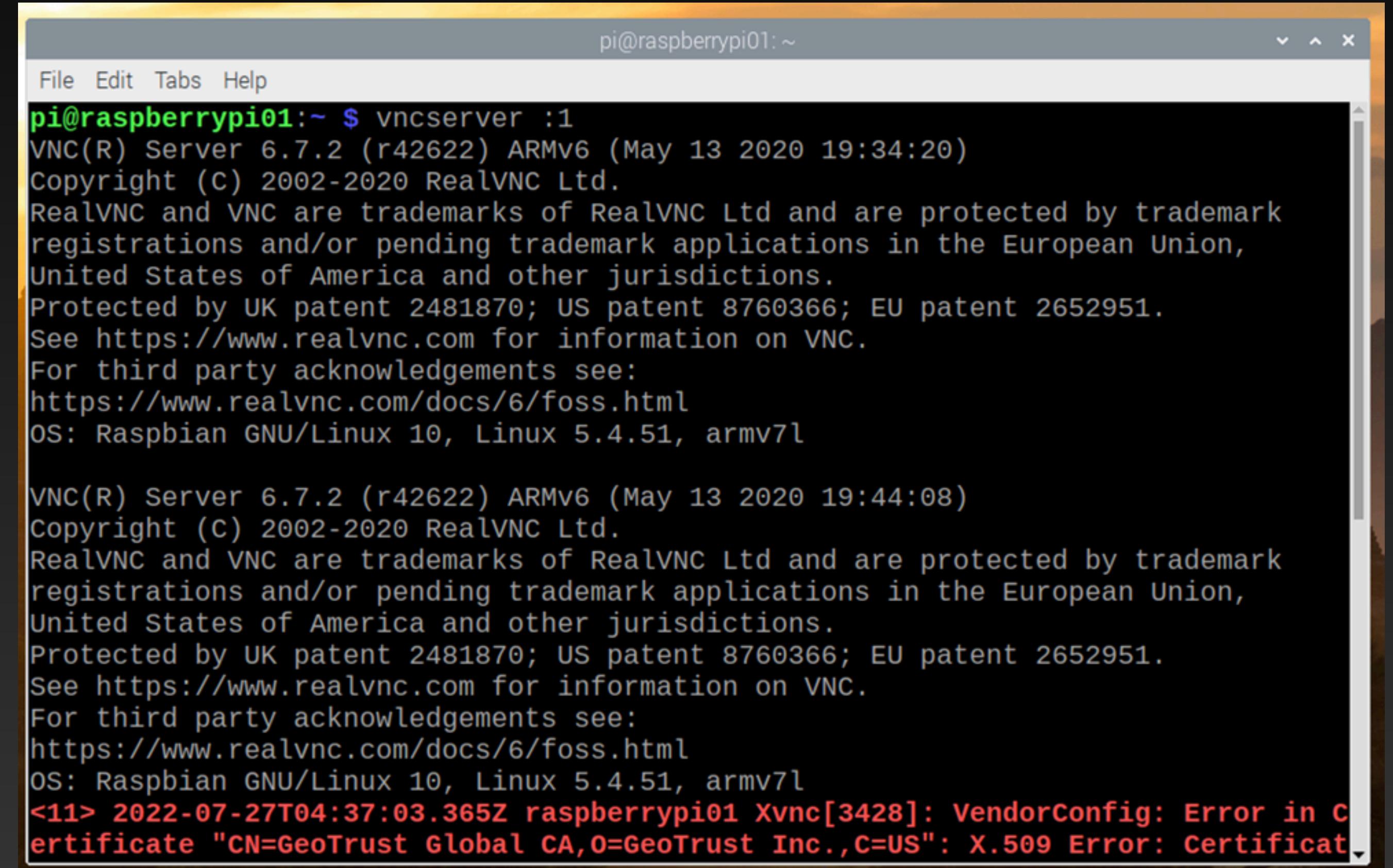
Remote Access-SSH

- Putty config screen
- Enter your IP, click open
- You will see security alert, Click Yes to accept
- You will prompt to enter your Pi username and password



Remote Access-VNC

- VNC consists of two part, VNC server and the VNC Viewer.
- VNC server run on RaspberryPi and VNC viewer run on your Window PC



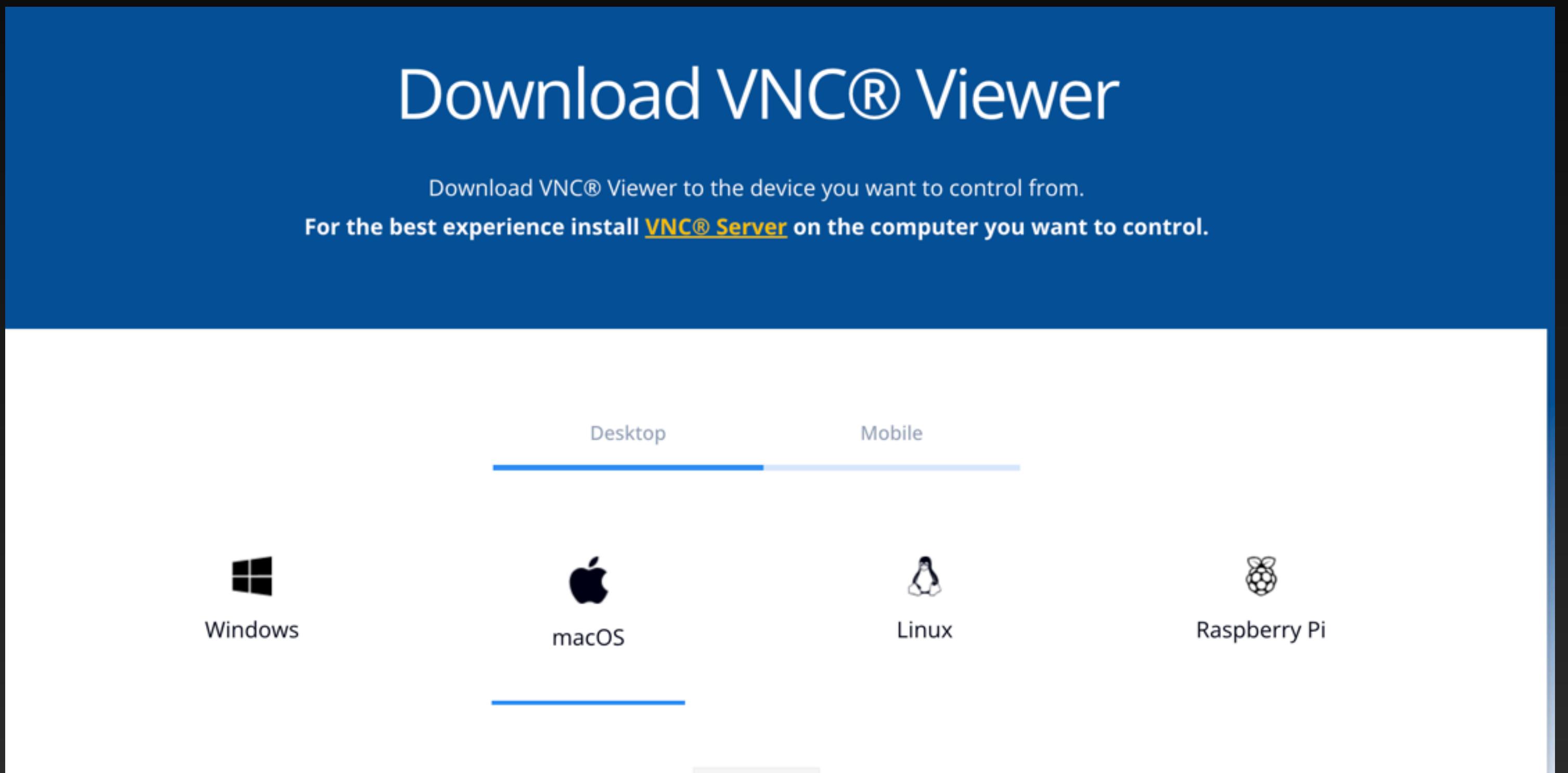
pi@raspberrypi01: ~

```
File Edit Tabs Help
pi@raspberrypi01:~ $ vncserver :1
VNC(R) Server 6.7.2 (r42622) ARMv6 (May 13 2020 19:34:20)
copyright (C) 2002-2020 RealVNC Ltd.
RealVNC and VNC are trademarks of RealVNC Ltd and are protected by trademark
registrations and/or pending trademark applications in the European Union,
United States of America and other jurisdictions.
Protected by UK patent 2481870; US patent 8760366; EU patent 2652951.
See https://www.realvnc.com for information on VNC.
For third party acknowledgements see:
https://www.realvnc.com/docs/6/foss.html
OS: Raspbian GNU/Linux 10, Linux 5.4.51, armv7l

VNC(R) Server 6.7.2 (r42622) ARMv6 (May 13 2020 19:44:08)
Copyright (C) 2002-2020 RealVNC Ltd.
RealVNC and VNC are trademarks of RealVNC Ltd and are protected by trademark
registrations and/or pending trademark applications in the European Union,
United States of America and other jurisdictions.
Protected by UK patent 2481870; US patent 8760366; EU patent 2652951.
See https://www.realvnc.com for information on VNC.
For third party acknowledgements see:
https://www.realvnc.com/docs/6/foss.html
OS: Raspbian GNU/Linux 10, Linux 5.4.51, armv7l
<11> 2022-07-27T04:37:03.365Z raspberrypi01 Xvnc[3428]: VendorConfig: Error in C
ertificate "CN=GeoTrust Global CA,O=GeoTrust Inc.,C=US": X.509 Error: Certificat
```

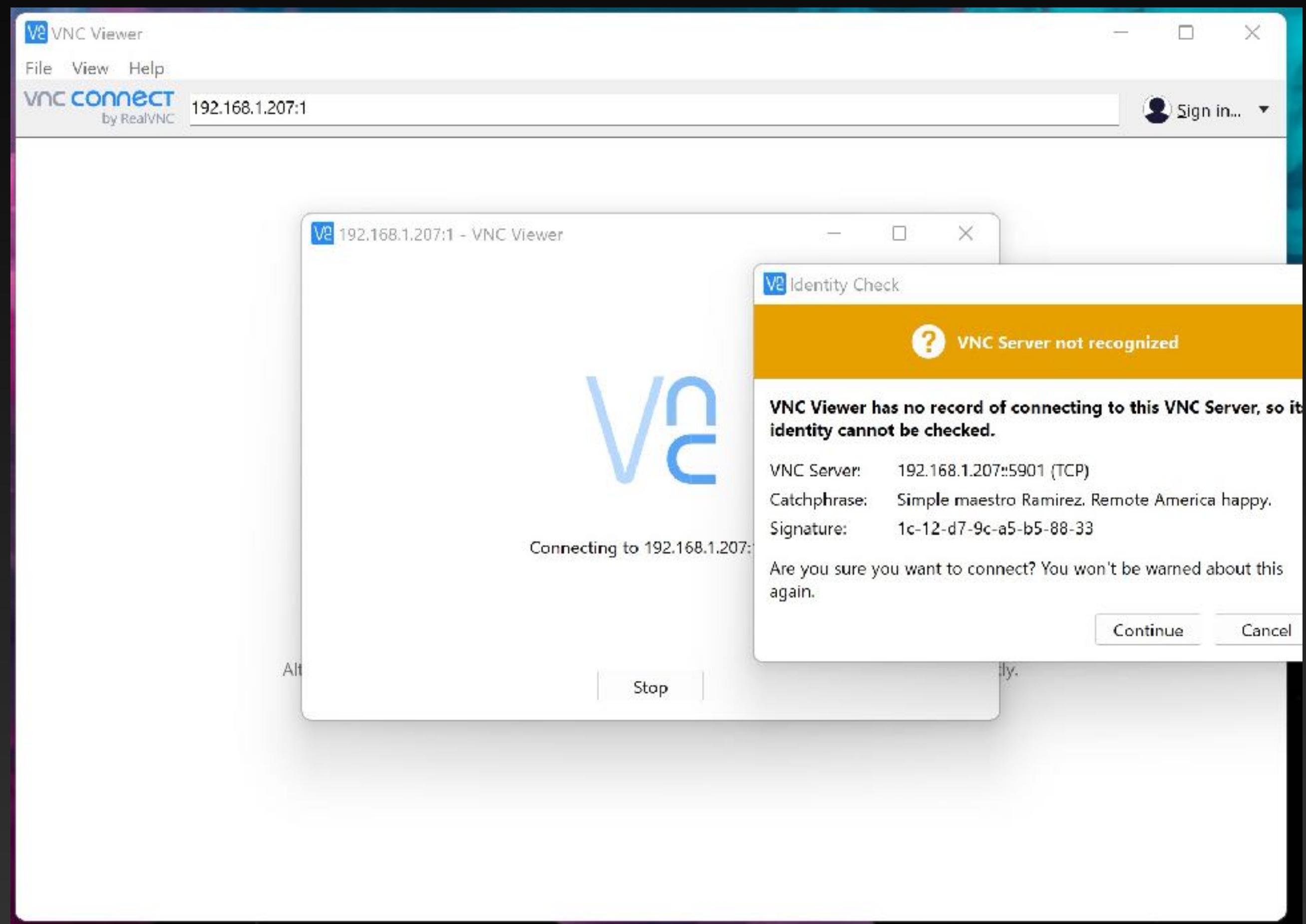
Remote Access-VNC

- [https://www.realvnc.com/
en/connect/download/
viewer/](https://www.realvnc.com/en/connect/download/viewer/)



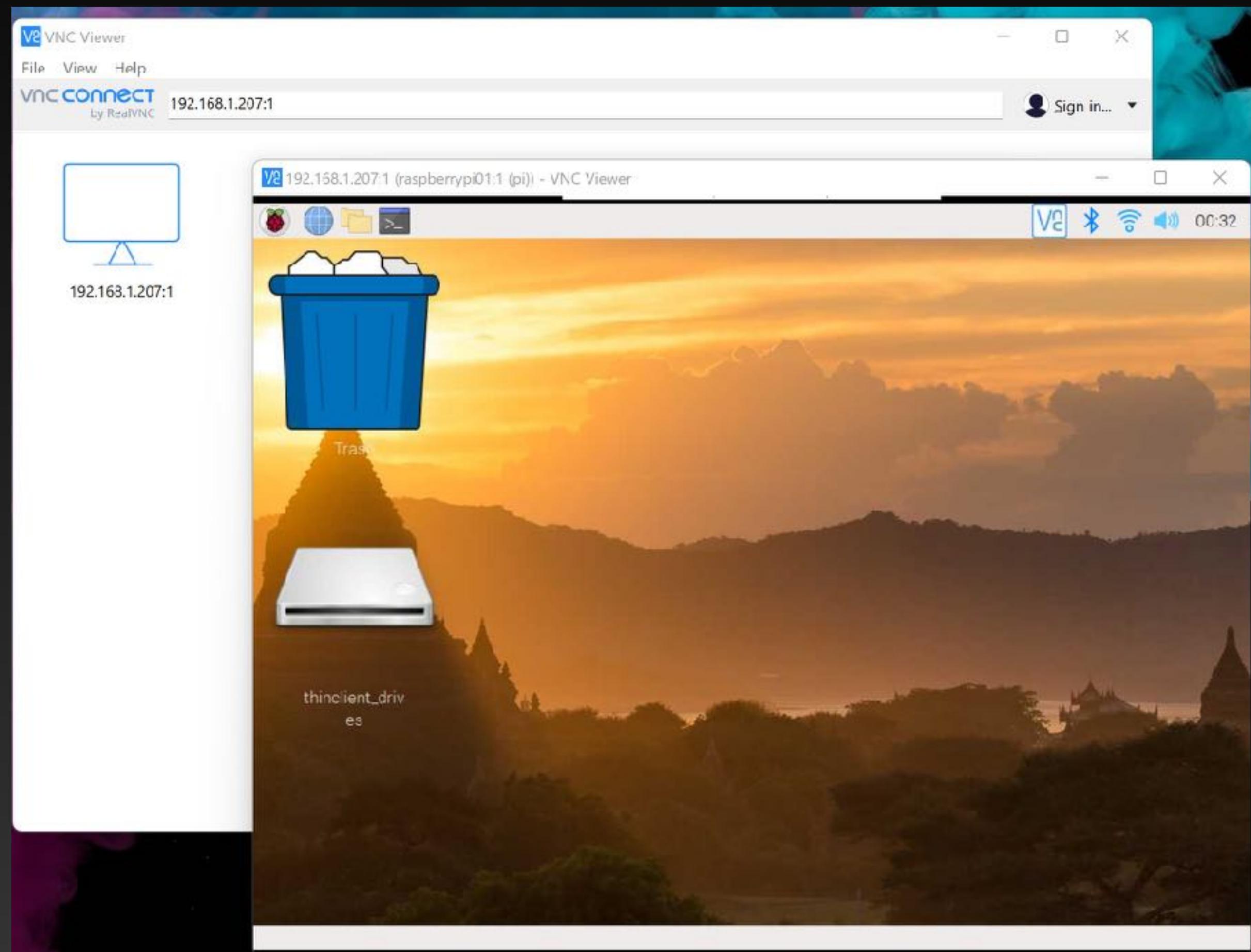
Remote Access-VNC

- Put your Pi IP address in the installed VNC viewer



Remote Access-VNC

- Enter your username and password



GPIO music box

In this project, you will build a button-controlled “music box” that plays different sounds when different buttons are pressed.

https://www.youtube.com/watch?v=2izvSzQWYak&feature=emb_title

What you will learn

Play sounds in Python with `pygame`

- Use the Python `gpiozero` library to connect button presses to function calls
- Use the dictionary data structure in Python

GPIO music box

What you will need

Hardware

- A Raspberry Pi computer
- A breadboard
- Four (4) tactile switches (to make buttons)
- Five (5) pin-to-socket jumper leads
- Four (4) pin-to-pin jumper leads
- Speakers or headphones

GPIO music box

Set up your project

You will need some sample sounds for this project. There are lots of sound files on Raspbian, but it can be a bit difficult to play them using Python. However, you can convert the sound files to a different file format that you can use in Python more easily.

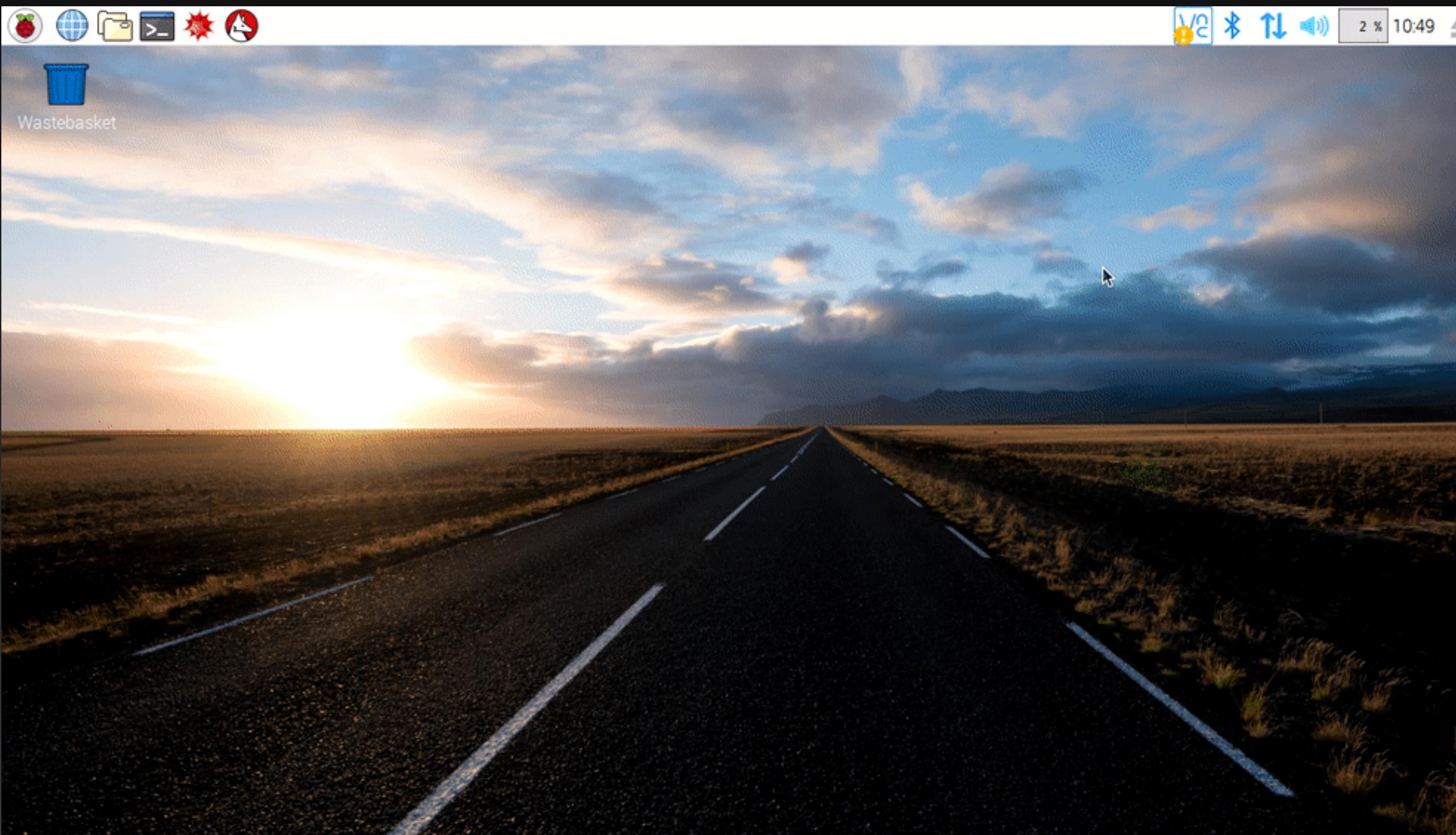
First, in your `home` directory, create a directory called `gpio-music-box`. You will use the new directory to store all your files for the project.

What method you will use to create your directories in Raspberry Pi?

GPIO music box

Set up your project

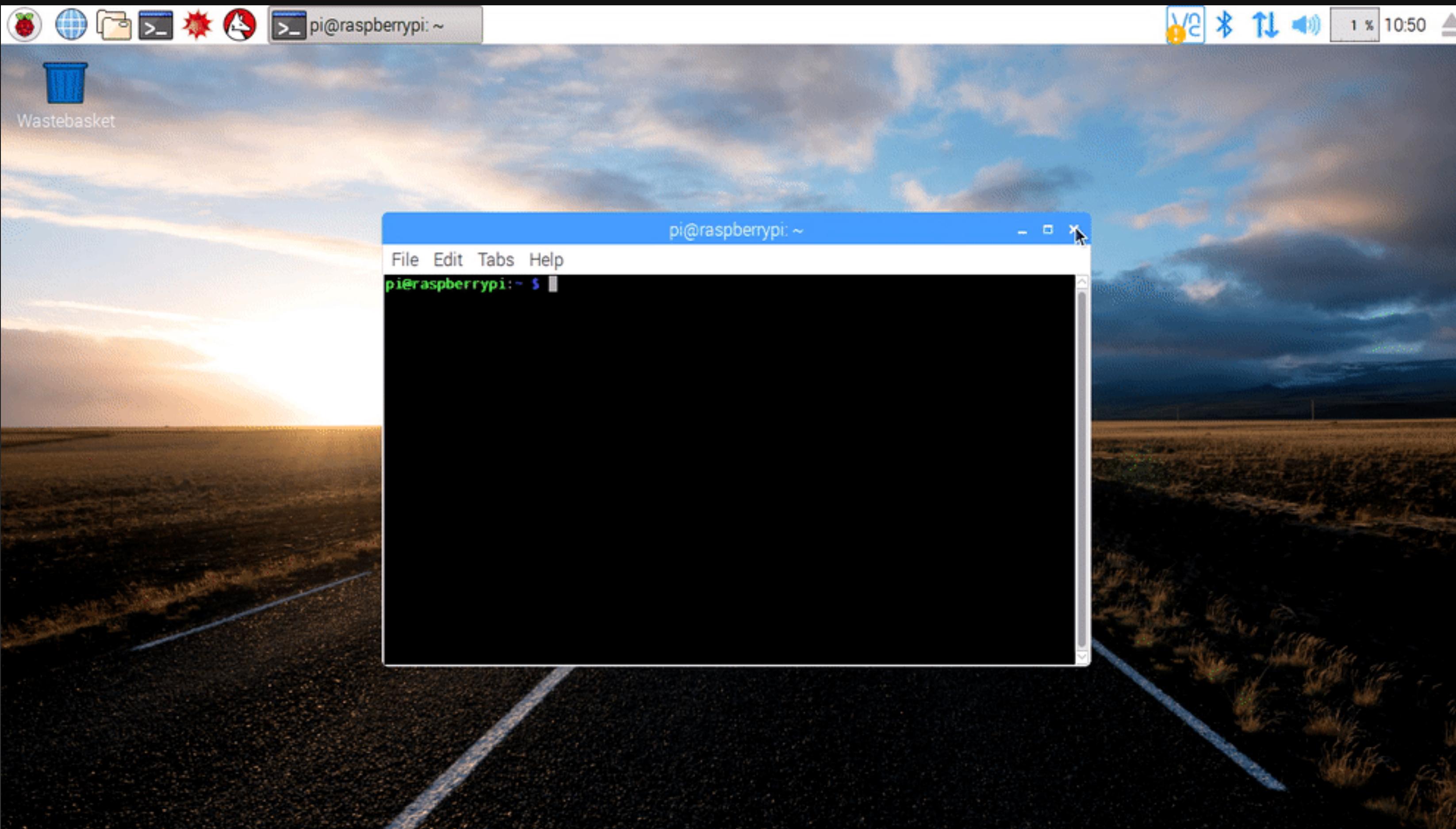
Method 1 - Using the GUI



GPIO music box

Set up your project

Method 1 - Using the Terminal



GPIO music box

Set up your project

Use the same method as before to create a new directory called `samples` in your `gpio-music-box` directory.

There are lots of sample sounds stored in `/usr/share/sonic-pi/samples`. In the next step, you will copy these sounds into the `gpio-music-box/samples` directory.

Type the following lines to copy all the files from one directory to the other:

```
cp /usr/share/sonic-pi/samples/* ~/gpio-music-box/samples/.
```

GPIO music box

Copy the sample sounds

Use the same method as before to create a new directory called `samples` in your `gpio-music-box` directory.

There are lots of sample sounds stored in `/usr/share/sonic-pi/samples`. In the next step, you will copy these sounds into the `gpio-music-box/samples` directory.

Type the following lines to copy all the files from one directory to the other:

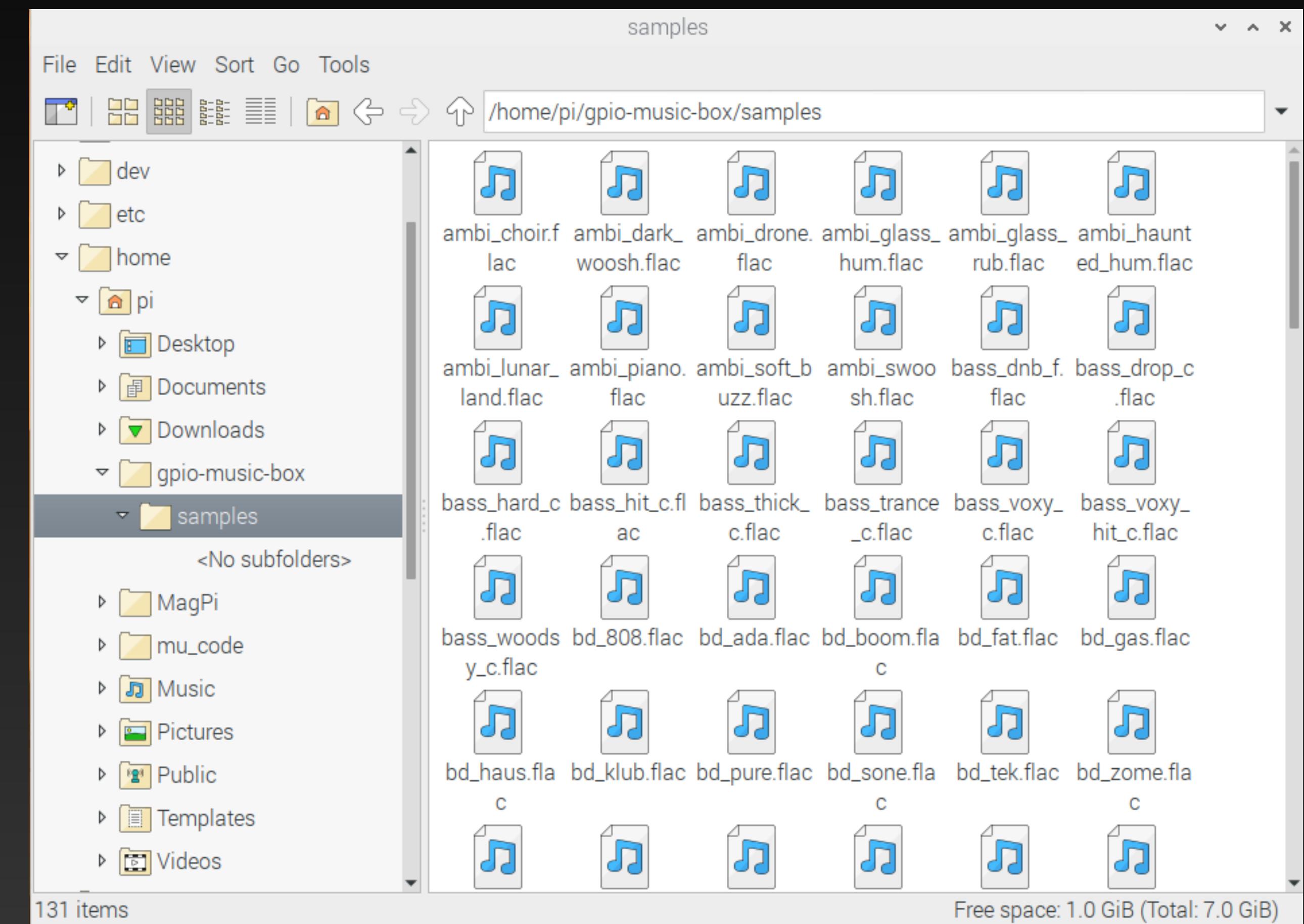
```
cp /usr/share/sonic-pi/samples/* ~/gpio-music-box/samples/.
```

When you have done that, you should be able to see all the `.flac` sound files in the `samples` directory.

GPIO music box

Copy the sample sounds

When you have done that, you should be able to see all the .flac sound files in the samples directory.



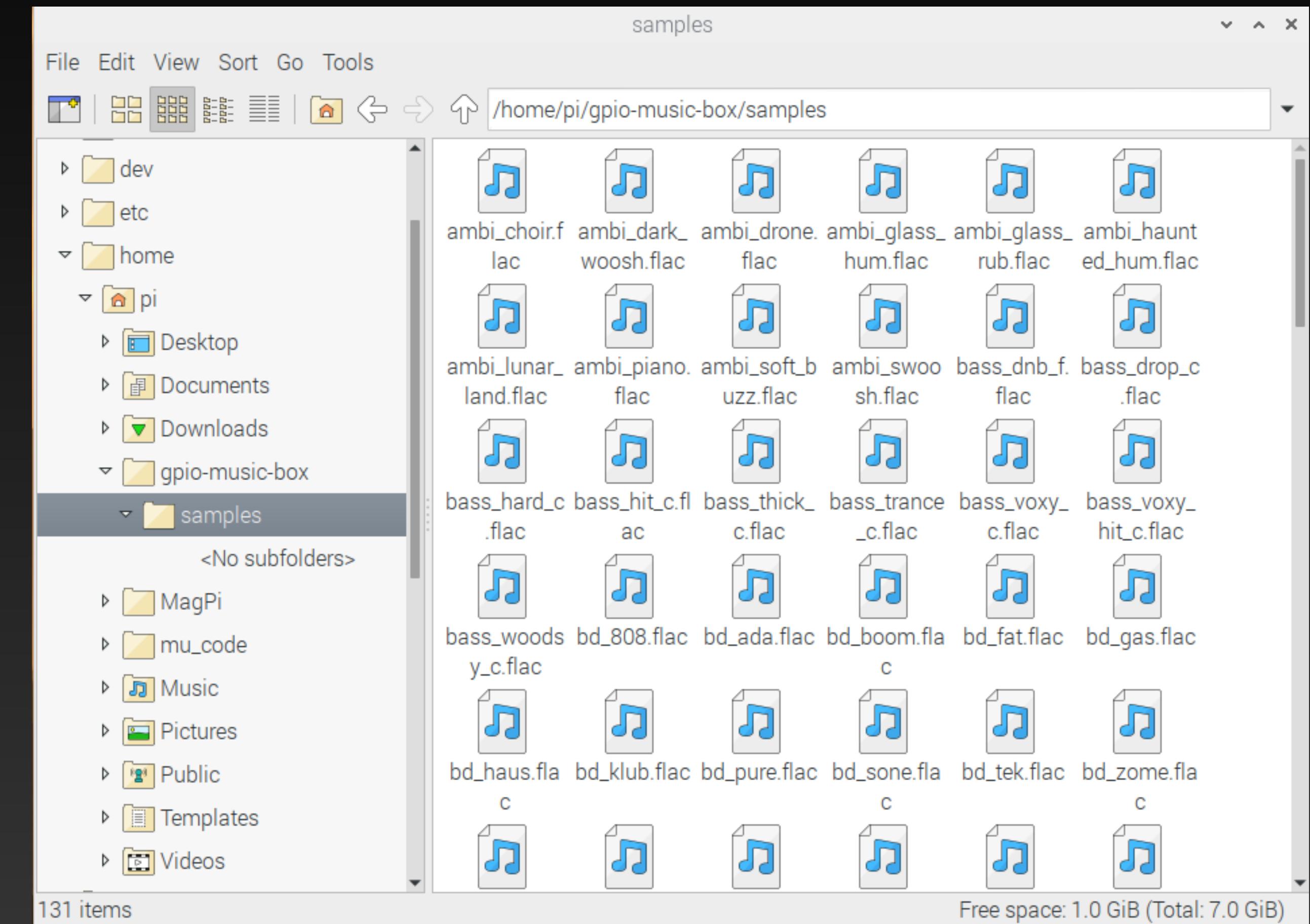
GPIO music box

Convert the sound files

To play the sound files using Python, you need to convert the files from `.flac` files to `.wav` files.

In the terminal, change into your `samples` directory.

```
cd ~/gpio-music-box/samples
```



GPIO music box

Convert the sound files

In your terminal, type the following commands. This will convert all the `.flac` files to `.wav` files, then delete the old files.

```
for f in *.flac; do ffmpeg -i "$f" "${f%.flac}.wav"; done
```

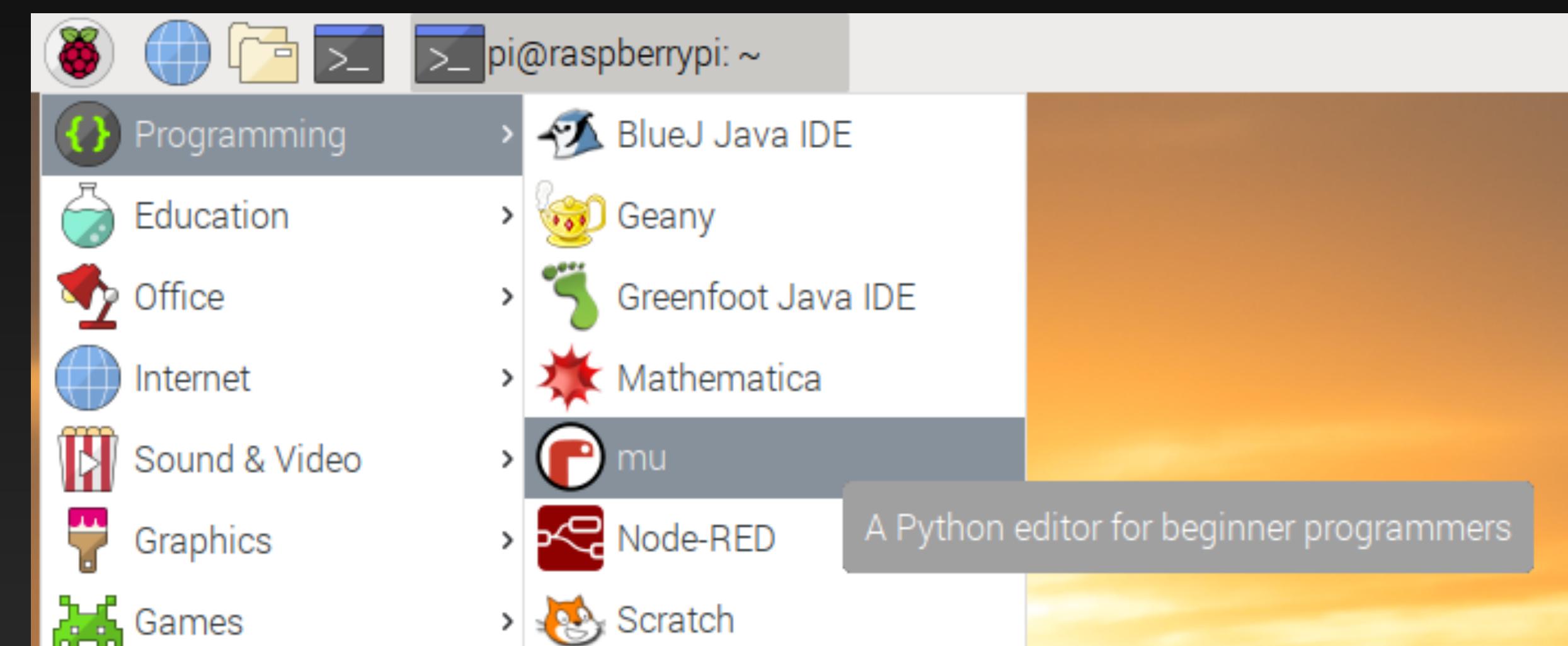
```
rm *.flac
```

GPIO music box

Play sounds

Next, you will start to write your Python code. You can use any text editor or IDE to do this — Mu is always a good choice.

To start to create the instruments of your music box, you need to test whether Python can play some of the samples that you have copied.



GPIO music box

Play sounds

First, import and initialise the `pygame` module for playing sound files.

```
import pygame
```

```
pygame.init()
```

Save this file in your `gpio-music-box` directory.

Choose four sound files that you want to use for your project, for example:

`drum_tom_mid_hard.wav`

`drum_cymbal_hard.wav`

`drum_snare_hard.wav`

`drum_cowbell.wav`

GPIO music box

Play sounds

Then, create a Python object that links to one of these sound files. Give the file its own unique name. For example:

```
drum = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_tom_mid_hard.wav")
```

Create named objects for your remaining three sounds.

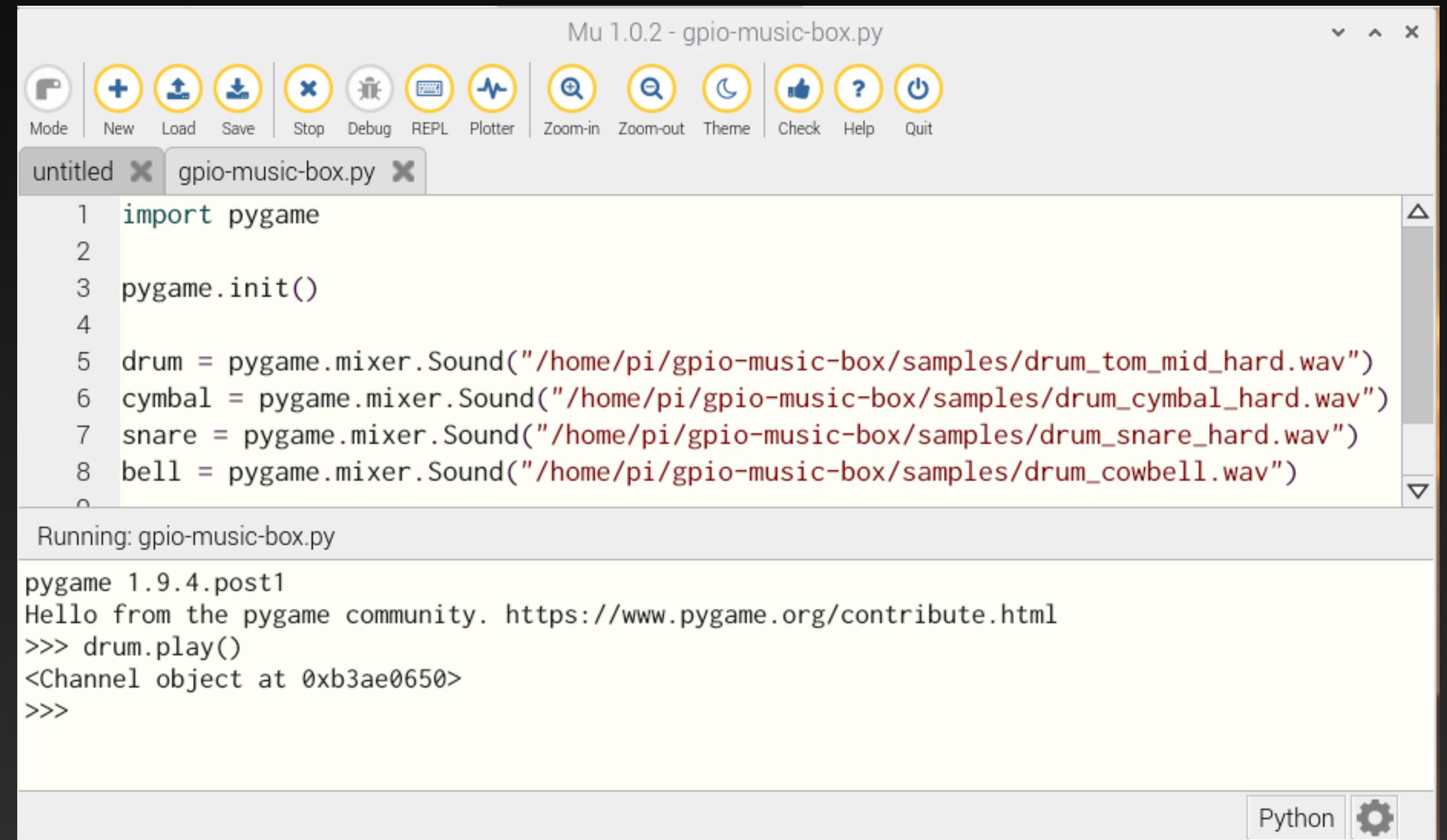
```
cymbal = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_cymbal_hard.wav")
snare = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_snare_hard.wav")
bell = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_cowbell_hard.wav")
```

Save and run your code. Then, in the shell at the bottom of the Mu editor, use `.play()` commands to play the sounds.

GPIO music box

Play sounds

If you don't hear any sound, check that your speakers or headphones are working and that the volume is turned up.



The screenshot shows the Mu 1.0.2 Python editor interface. The title bar reads "Mu 1.0.2 - gpio-music-box.py". The toolbar contains icons for Mode, New, Load, Save, Stop, Debug, REPL, Plotter, Zoom-in, Zoom-out, Theme, Check, Help, and Quit. The code editor window has tabs for "untitled" and "gpio-music-box.py" (which is currently selected). The code itself imports pygame, initializes it, and defines four Sound objects: drum, cymbal, snare, and bell, each loaded from a specific WAV file path. Below the code, the terminal output shows the program running, identifying itself as pygame 1.9.4.post1 and displaying a message from the pygame community. It then shows the command `drum.play()` being run, which returns a Channel object at memory address 0xb3ae0650. The bottom right corner of the editor shows a "Python" tab and a gear icon for settings.

```
1 import pygame
2
3 pygame.init()
4
5 drum = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_tom_mid_hard.wav")
6 cymbal = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_cymbal_hard.wav")
7 snare = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_snare_hard.wav")
8 bell = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_cowbell.wav")
```

Running: gpio-music-box.py

```
pygame 1.9.4.post1
Hello from the pygame community. https://www.pygame.org/contribute.html
>>> drum.play()
<Channel object at 0xb3ae0650>
>>>
```

Python 

GPIO music box

Connect your buttons

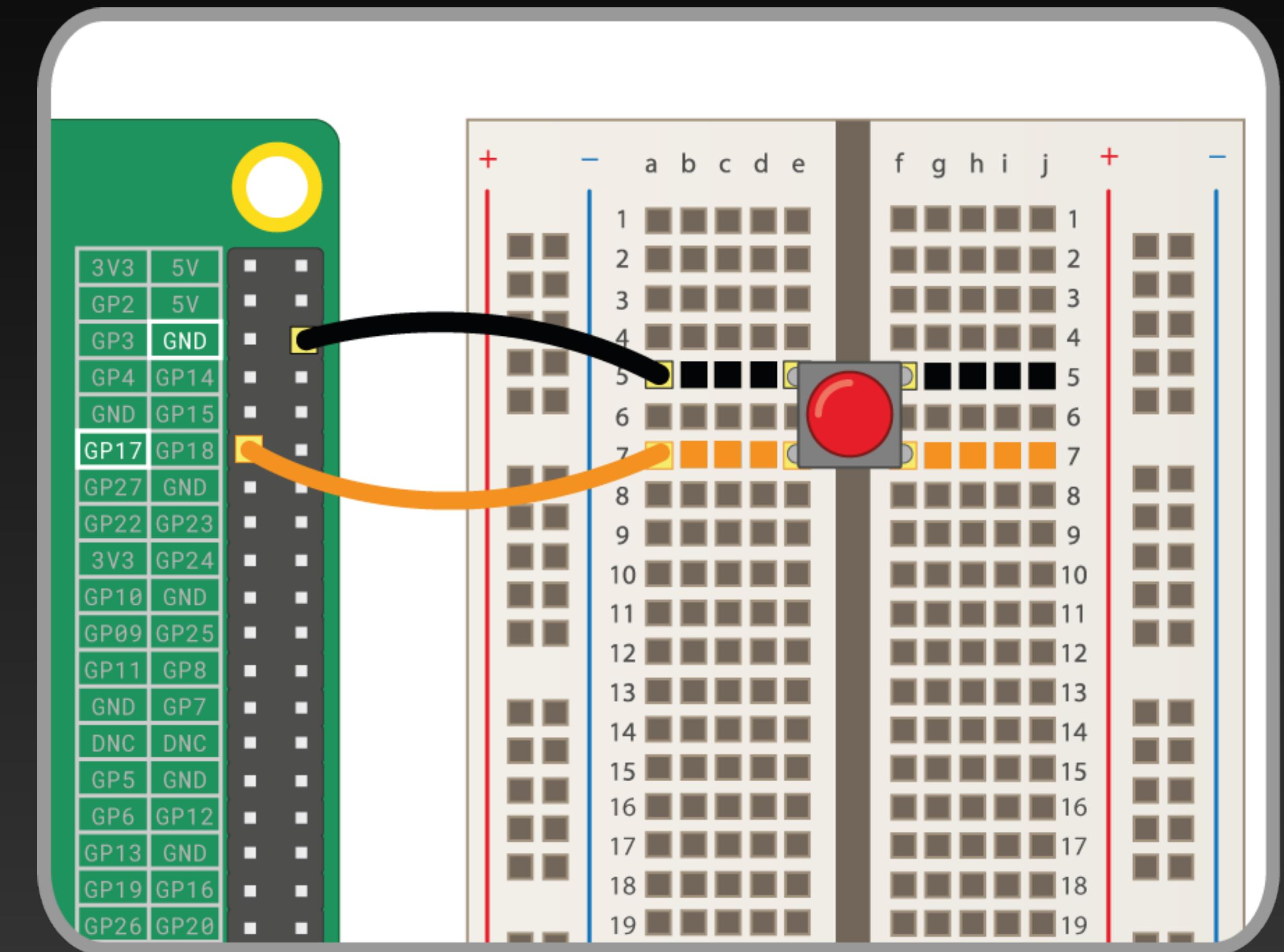
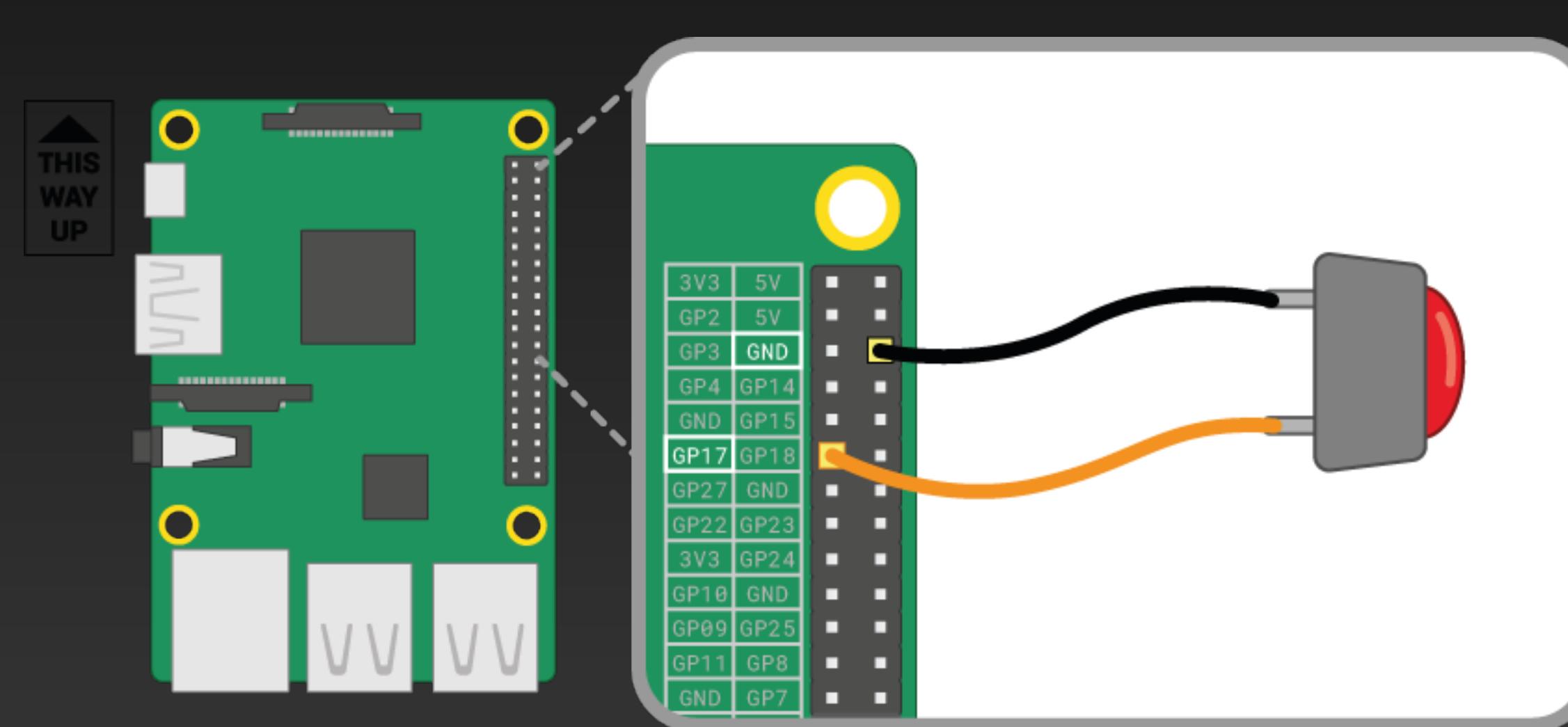
A button is one of the simplest input components you can wire to a Raspberry Pi. It's a non-polarised component, which means you can place it in a circuit either way round and it will work.

There are various types of buttons - they can for example have two or four legs. The two-leg versions are mostly used with flying wire to connect to the control device. Buttons with four legs are generally mounted on a PCB or a breadboard.

GPIO music box

Connect your buttons

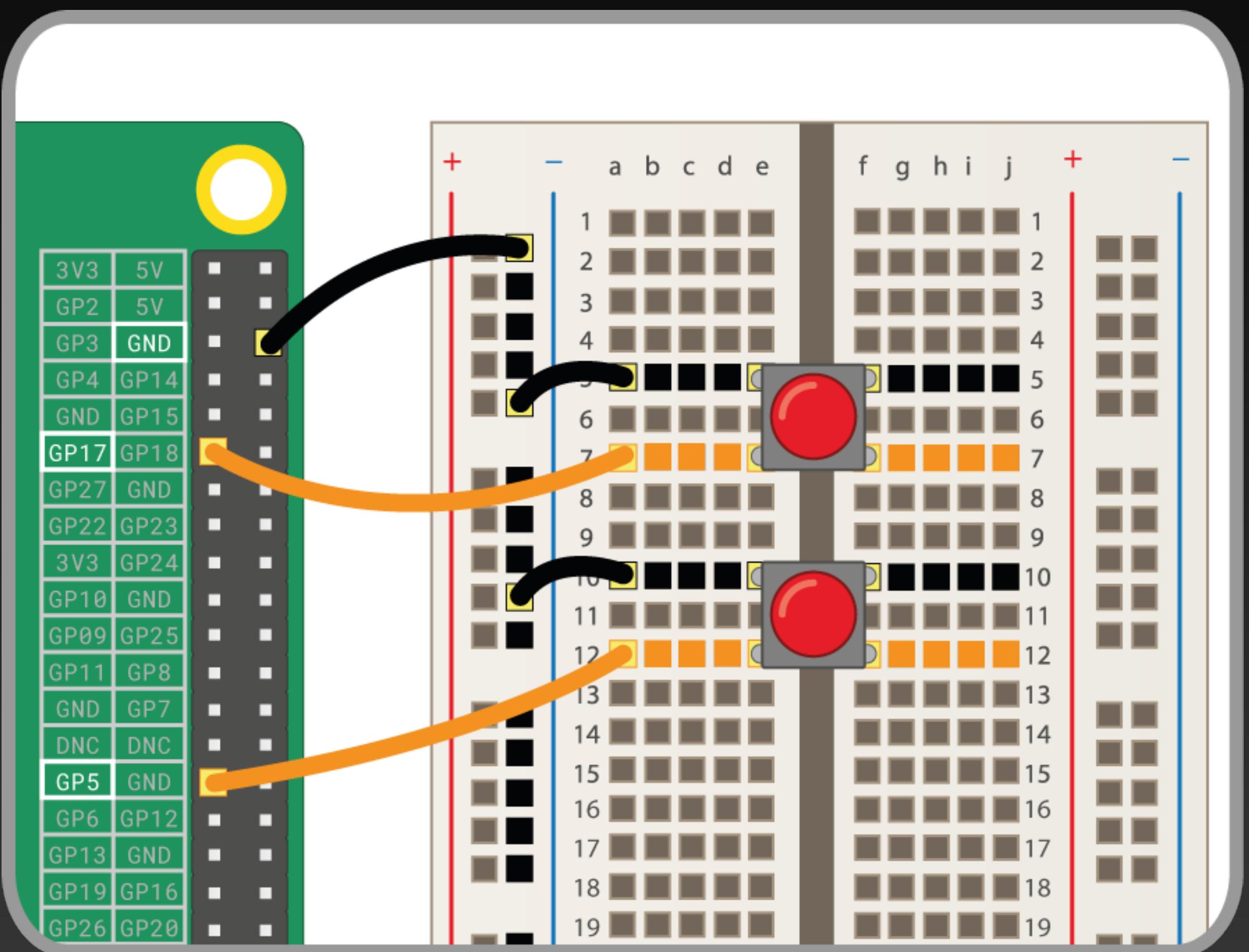
The diagrams below shows how to wire a two-leg or four-leg button to a Raspberry Pi. In both cases, GPIO 17 is the input pin



GPIO music box

Connect your buttons

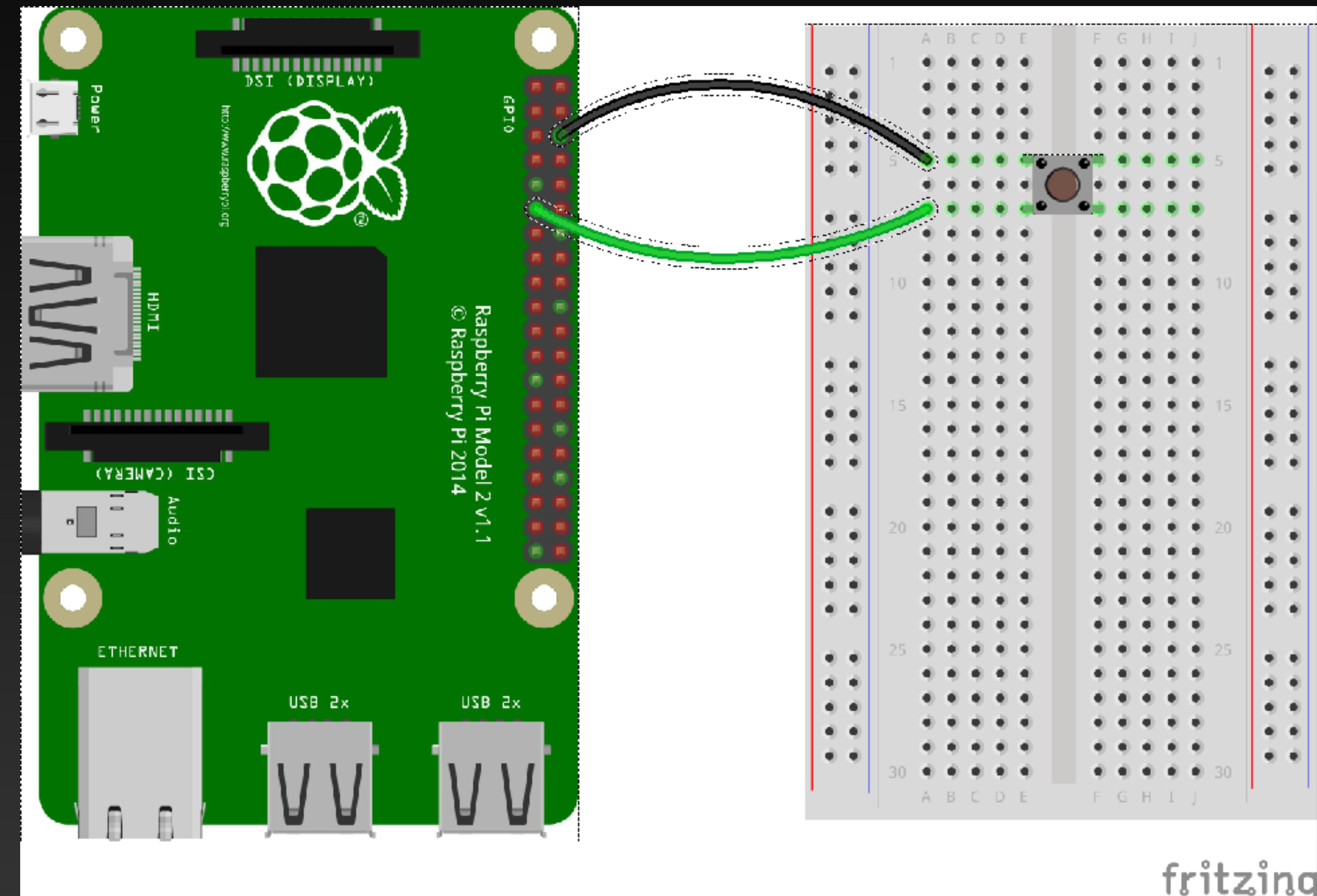
If you are using multiple buttons, then it is often best to use a common ground to avoid connecting too many jumper leads to GND pins. You can wire the negative rail on the breadboard to a single ground pin, which allows all the buttons to use the same ground rail.



GPIO music box

Test the button

```
from gpiozero import Button  
btn = Button(17)  
  
def hello():  
    print('Hello')  
  
btn.when_pressed = hello
```



fritzing

GPIO music box

Test the button

```
import pygame
from gpiozero import Button

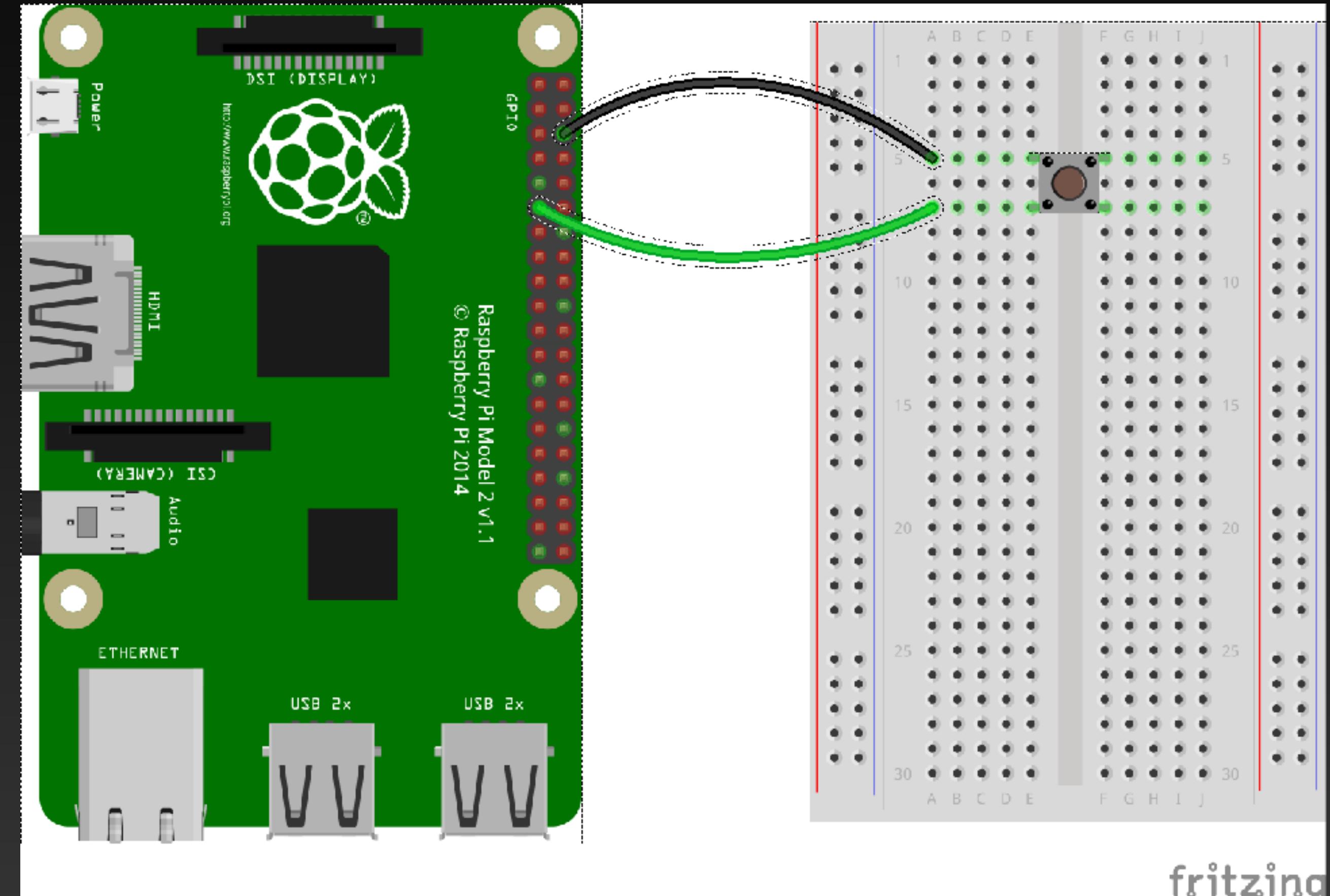
pygame.init()

drum = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_tom_mid_hard.wav")
cymbal = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_cymbal_hard.wav")
snare = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_snare_hard.wav")
bell = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_cowbell.wav")

btn_drum = Button(17)
```

To play the sound when the button is pressed, just add this line of code to the bottom of your file:

```
btn_drum.when_pressed = drum.play
```

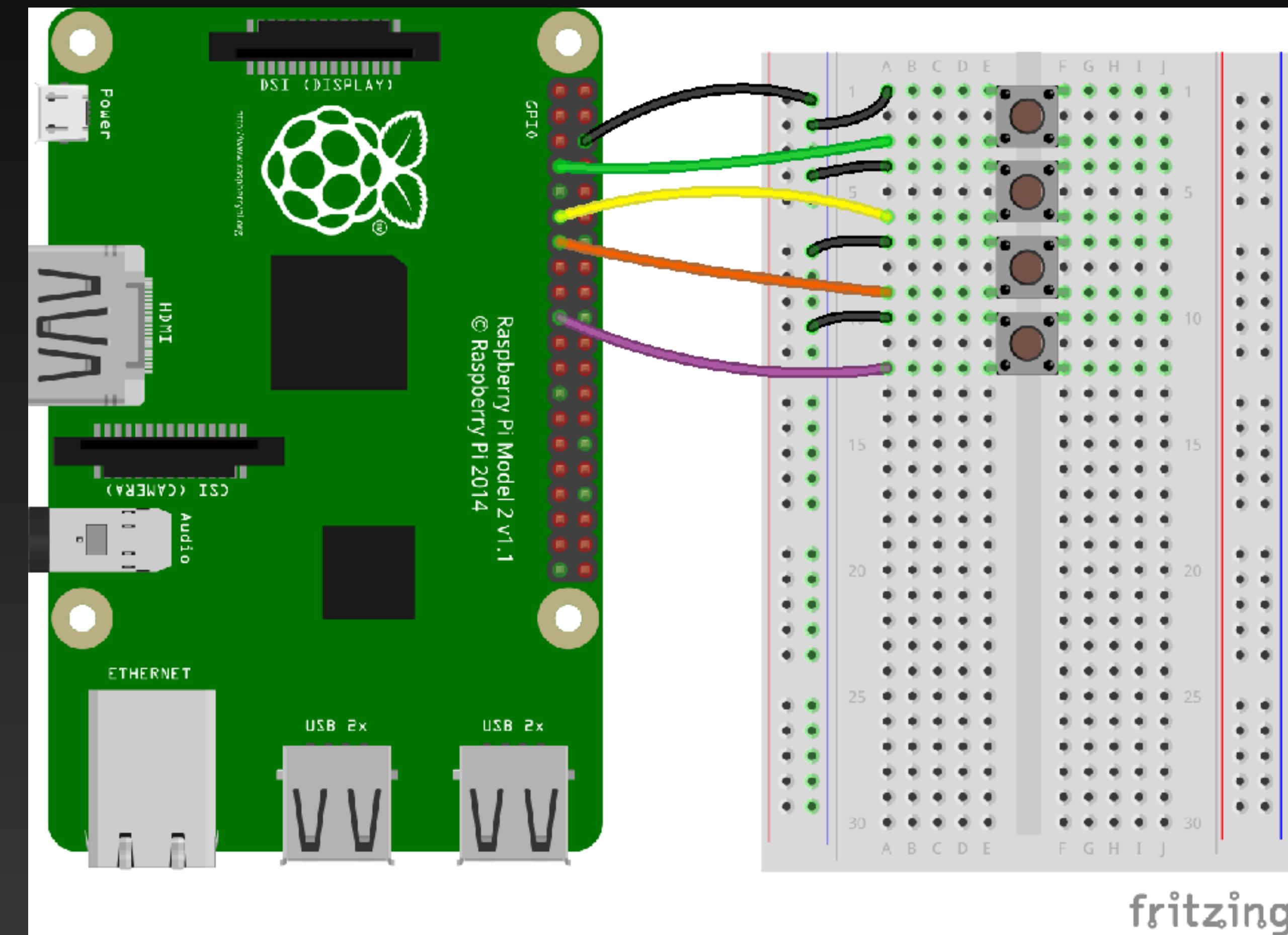


GPIO music box

Connect your buttons

Place the four buttons into your breadboard.

Wire each button to a different numbered GPIO pin. You can choose any pins you like, but you will need to remember the numbers.



fritzing

GPIO music box

Connect your buttons

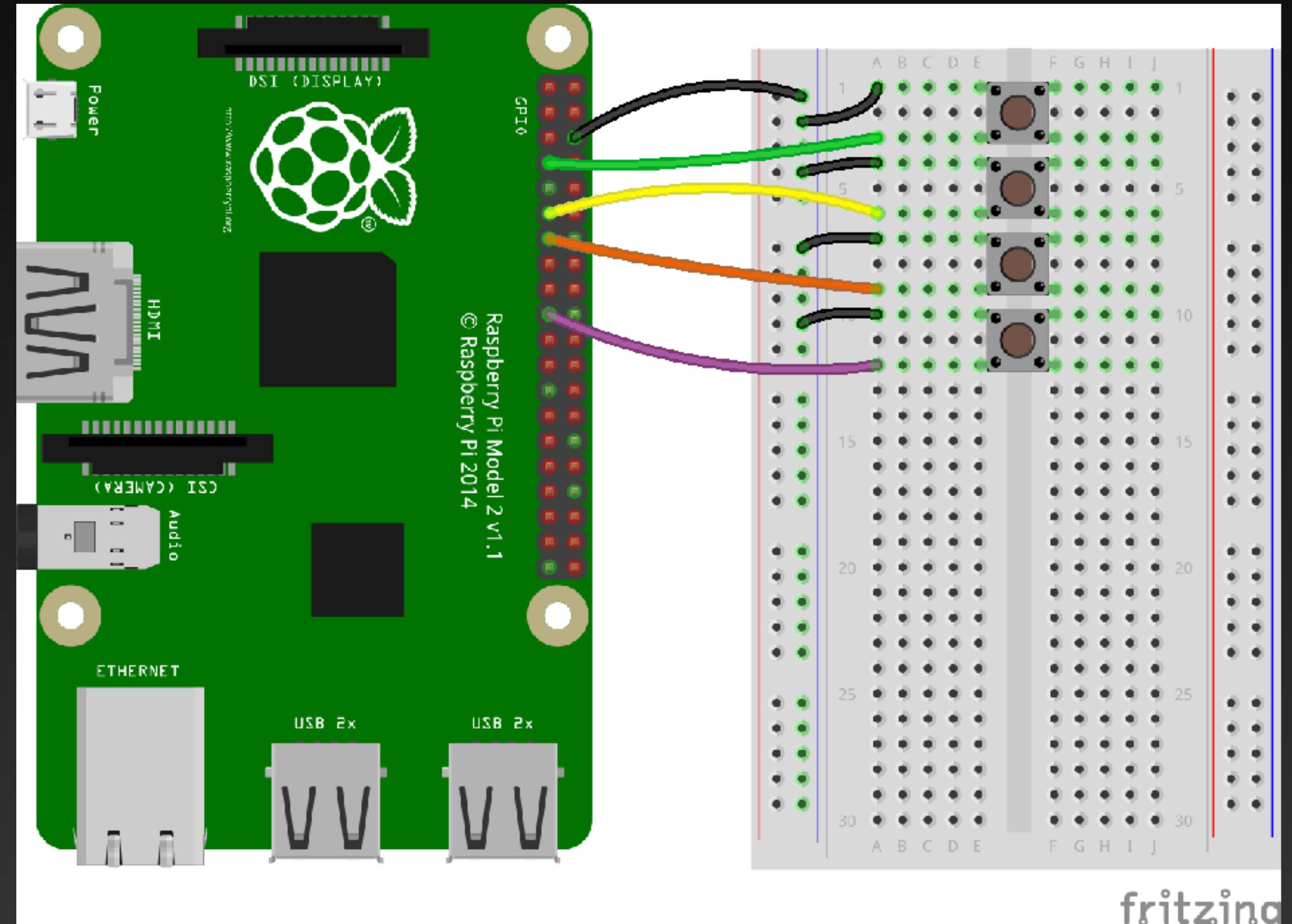
```
import pygame
from gpiozero import Button

pygame.init()

drum = pygame.mixer.Sound("/home/pi/gpio-music-
box/samples/drum_tom_mid_hard.wav")
cymbal = pygame.mixer.Sound("/home/pi/gpio-music-
box/samples/drum_cymbal_hard.wav")
snare = pygame.mixer.Sound("/home/pi/gpio-music-
box/samples/drum_snare_hard.wav")
bell = pygame.mixer.Sound("/home/pi/gpio-music-
box/samples/drum_cowbell.wav")

btn_drum = Button(4)
btn_cymbal = Button(17)
btn_snare= Button(27)
btn_bell = Button(10)

btn_drum.when_pressed = drum.play
btn_cymbal.when_pressed = cymbal.play
btn_snare.when_pressed = snare.play
btn_bell.when_pressed = bell.play
```



fritzing

GPIO music box

Improve your code

```
import pygame
from gpiozero import Button

pygame.init()

button_sounds = {Button(4): pygame.mixer.Sound("/home/pi/gpio-music-box/samples/
drum_tom_mid_hard.wav"),
                 Button(17): pygame.mixer.Sound("/home/pi/gpio-music-box/samples/
drum_cymbal_hard.wav"),
                 Button(27): pygame.mixer.Sound("/home/pi/gpio-music-box/samples/
drum_snare_hard.wav"),
                 Button(10): pygame.mixer.Sound("/home/pi/gpio-music-box/samples/
drum_cowbell.wav")}

for button, sound in button_sounds.items():
    button.when_pressed = sound.play
```

GPIO music box

Improve your code

Homework 1: finish the class project with 4 buttons

Homework 2: add led to class project and modify your code to use button to control LED on/off