

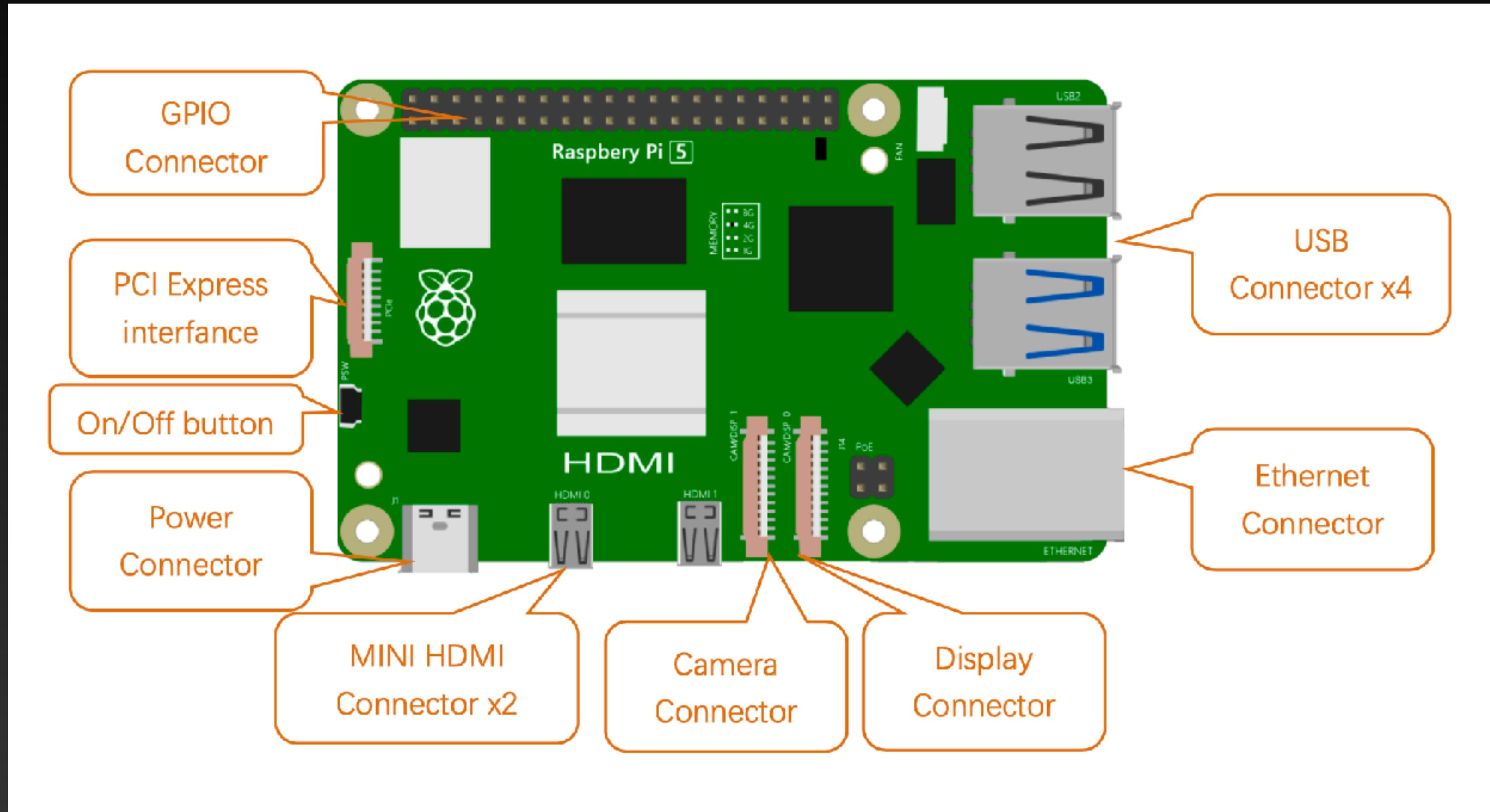
# MIT AI2 204

# IoT with MIT App Inventor

Fundamental

X. Tang

# GPIO control in python on Raspberry Pi

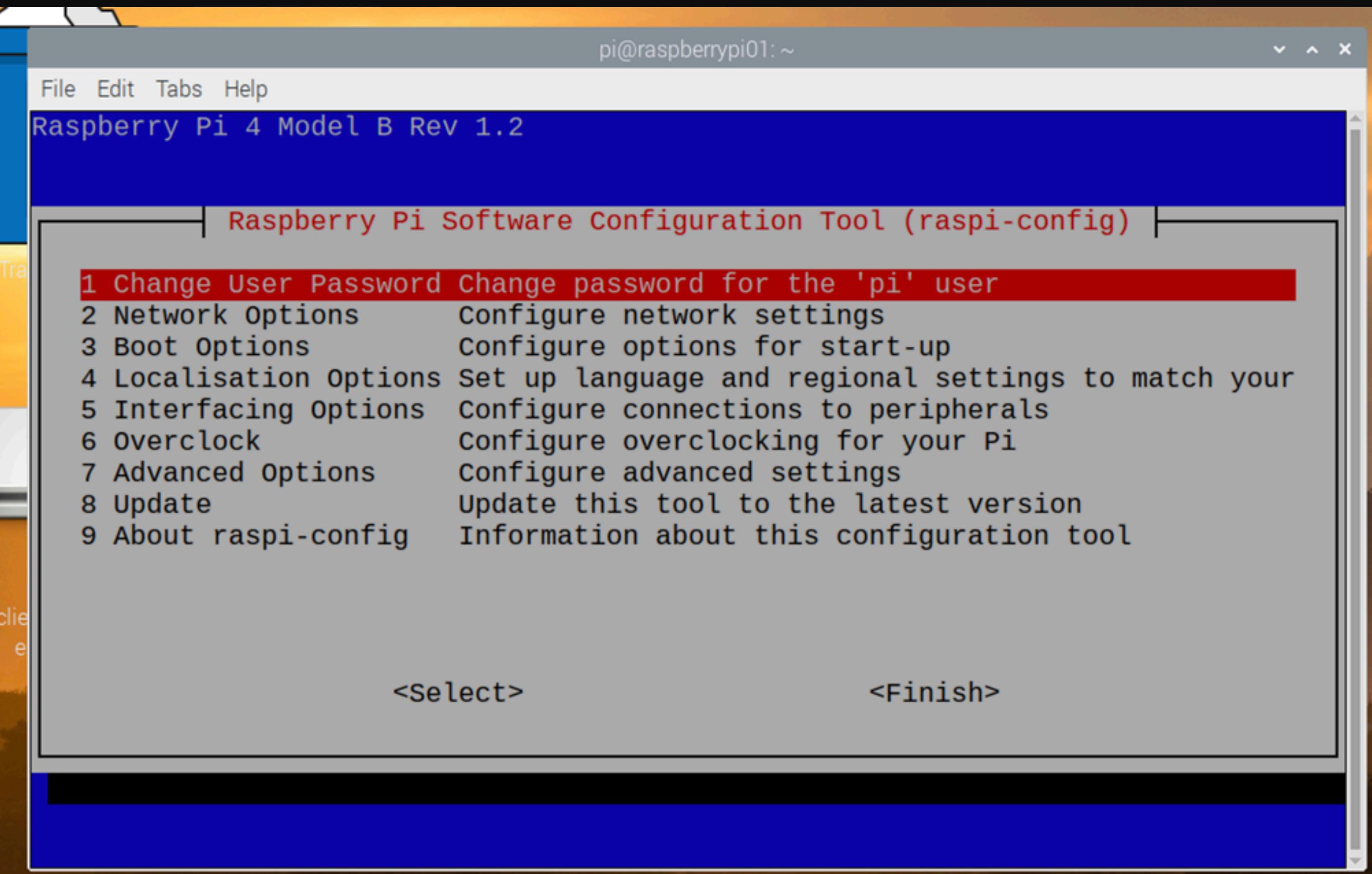


# Remote Access -Setup

- \$ 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.realvnc.com/en/connect/download/viewer/>
- MAC, app store, windows remote desktop

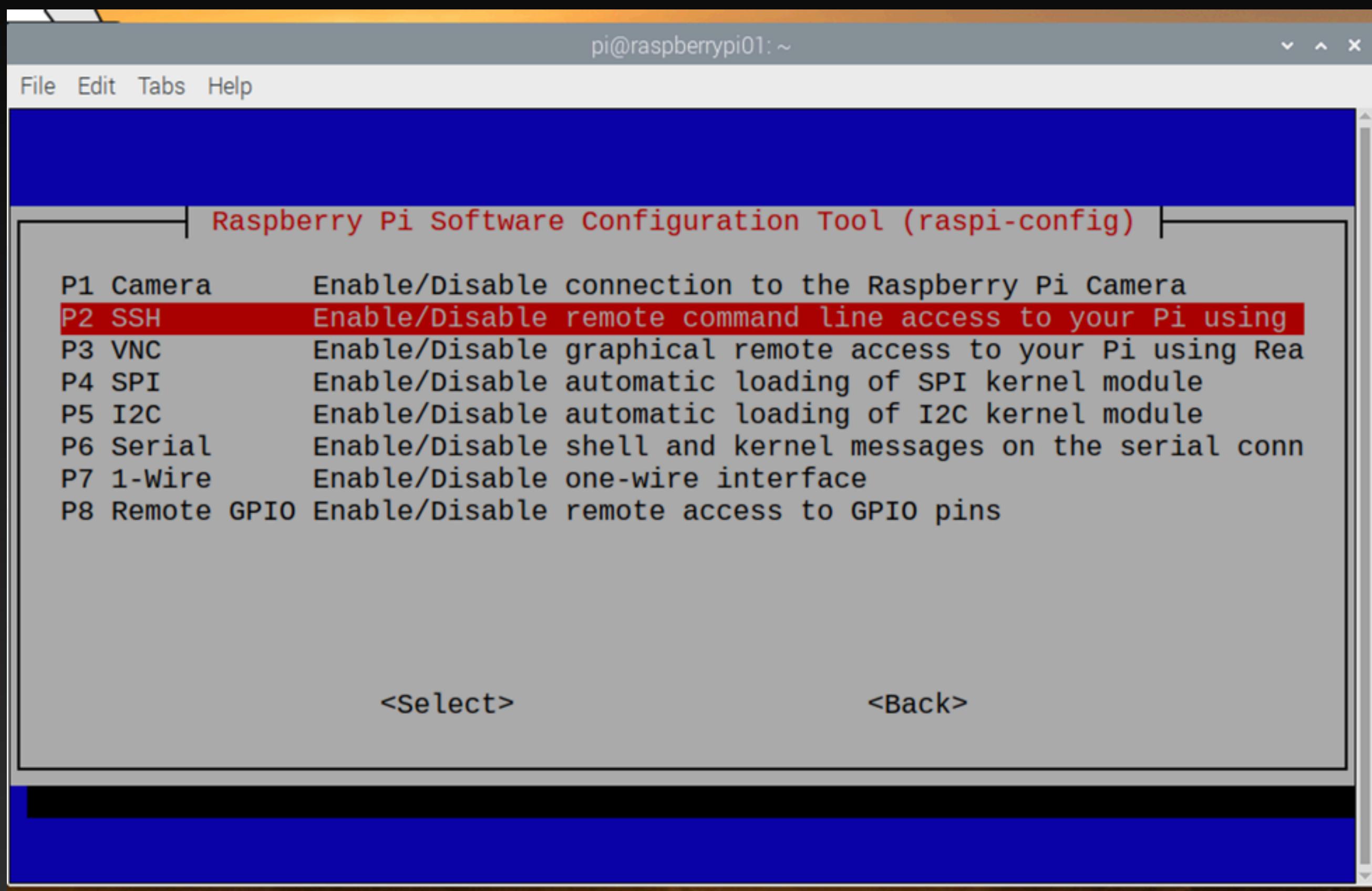
# Remote Access -Setup

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



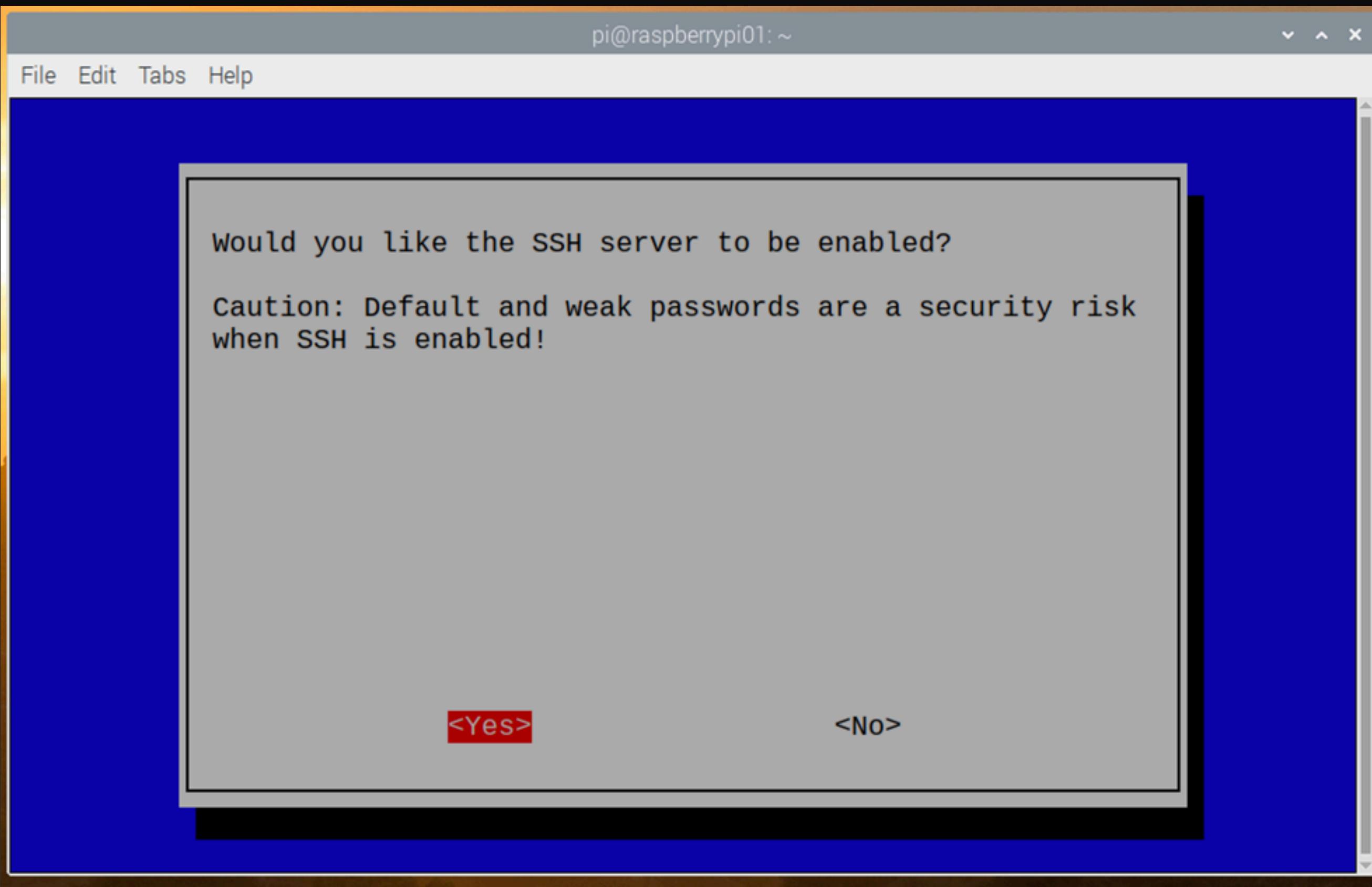
# Remote Access -Setup

- Choose P2 SSH, hit enter



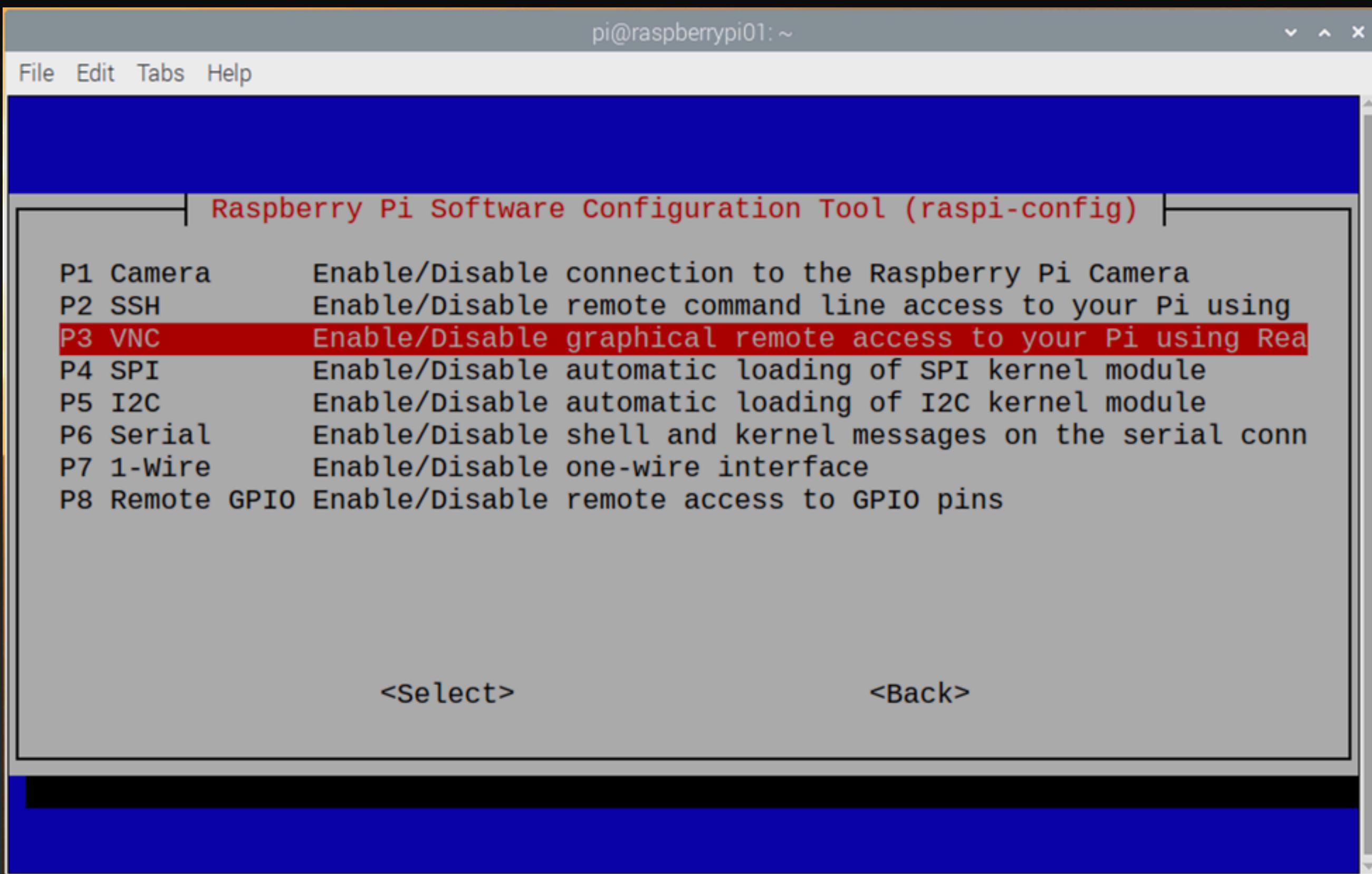
# Remote Access -Setup

- Select Yes, hit enter



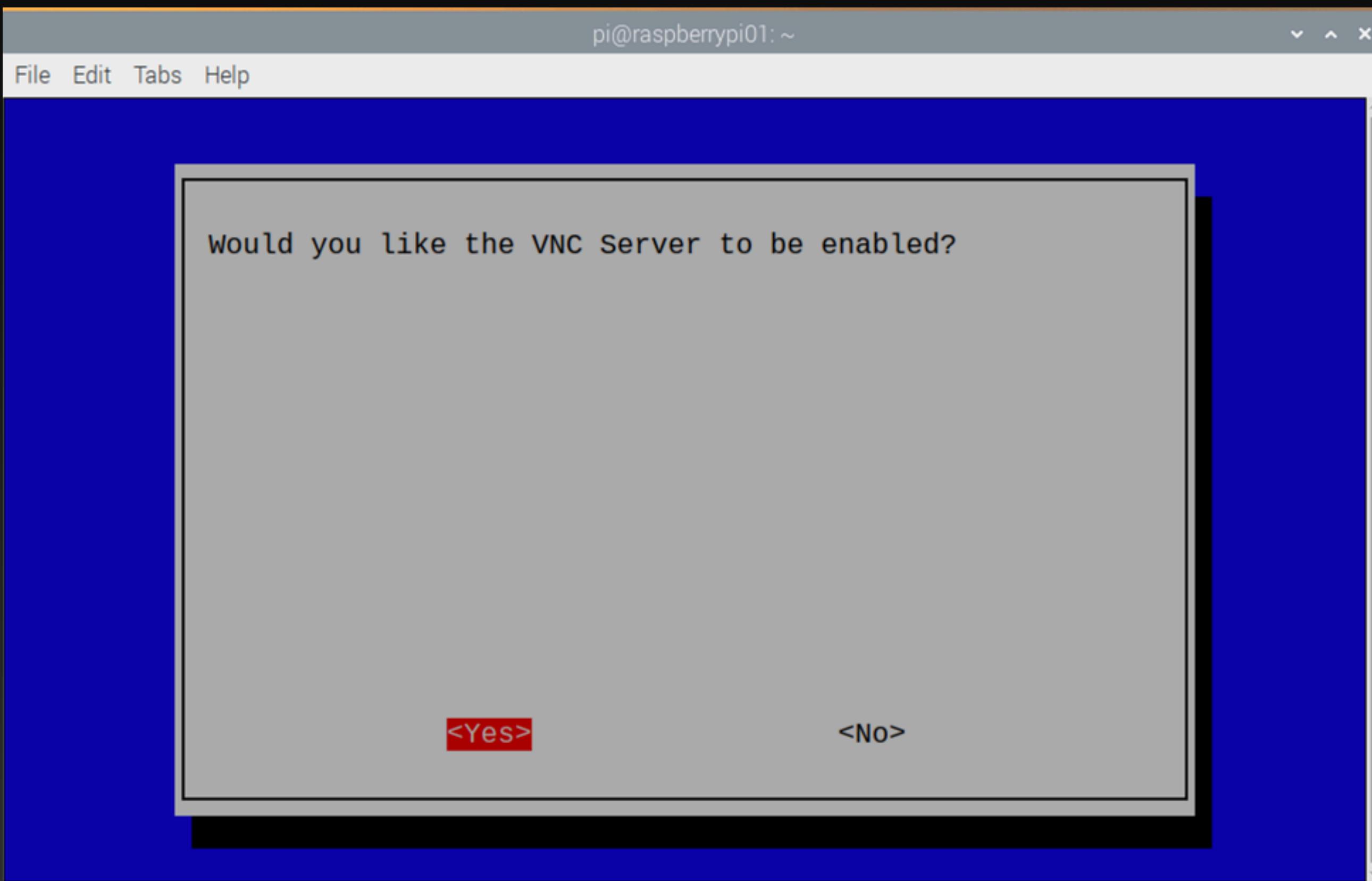
# Remote Access -Setup

- Choose P3 VNC, hit enter



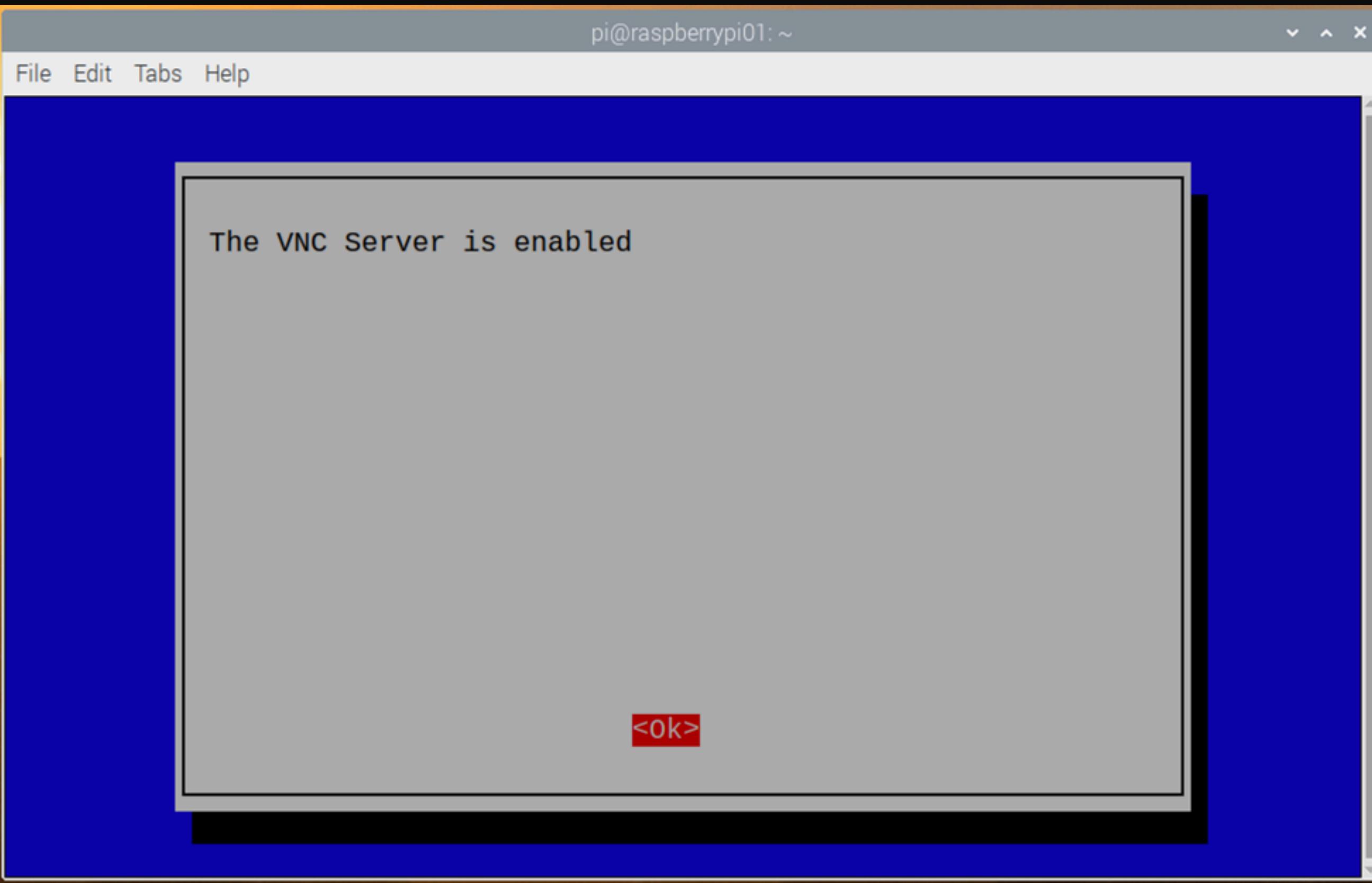
# Remote Access -Setup

- Select Yes, hit enter



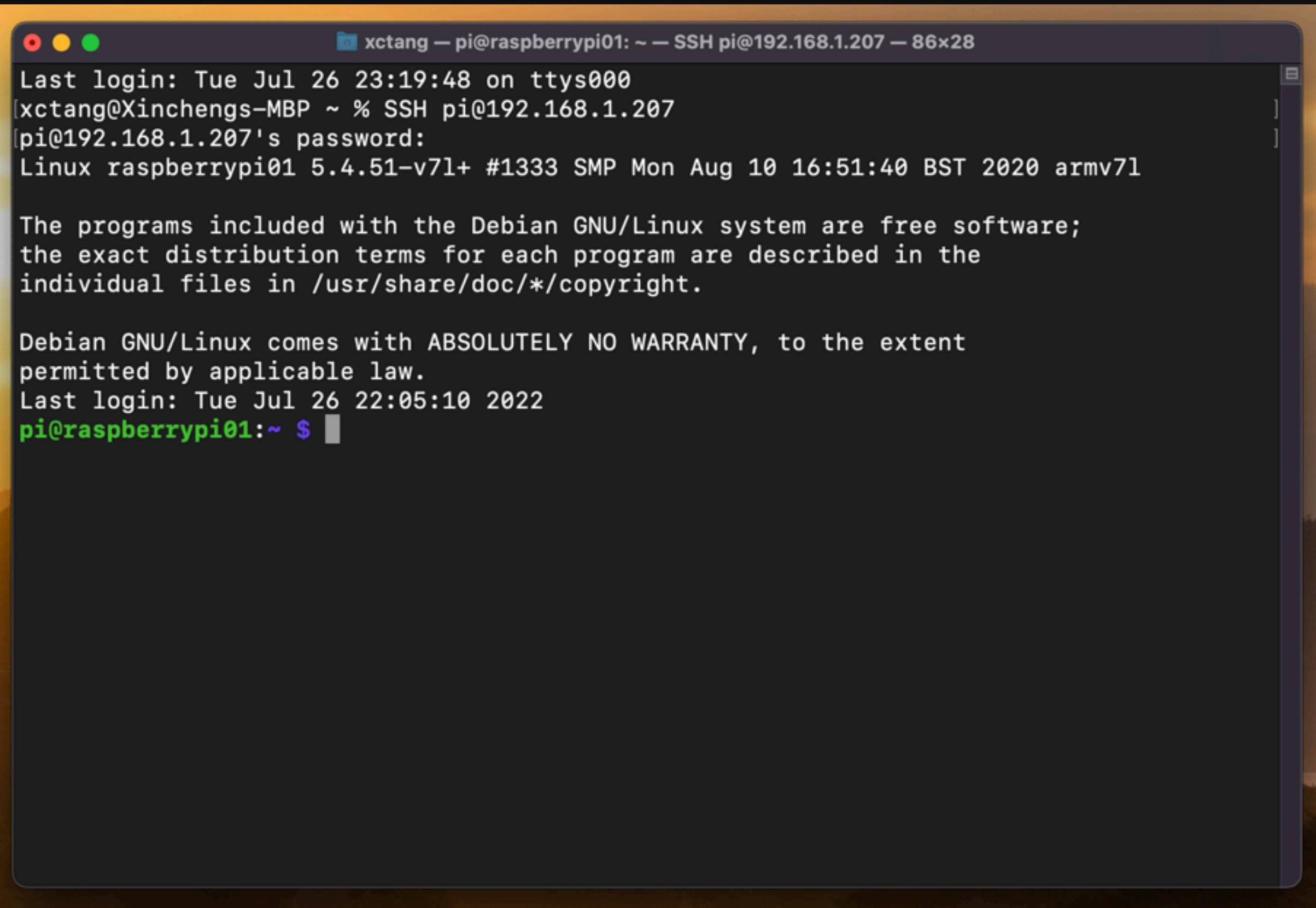
# Remote Access -Setup

- 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

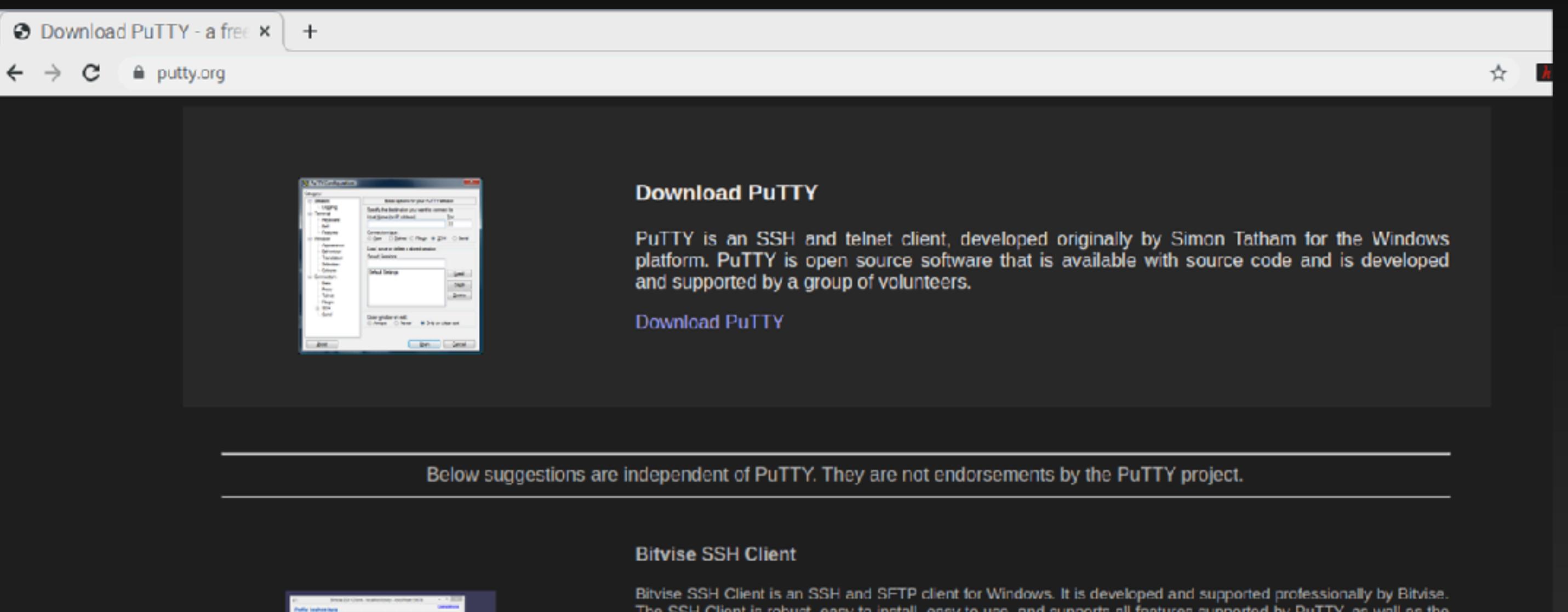
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

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

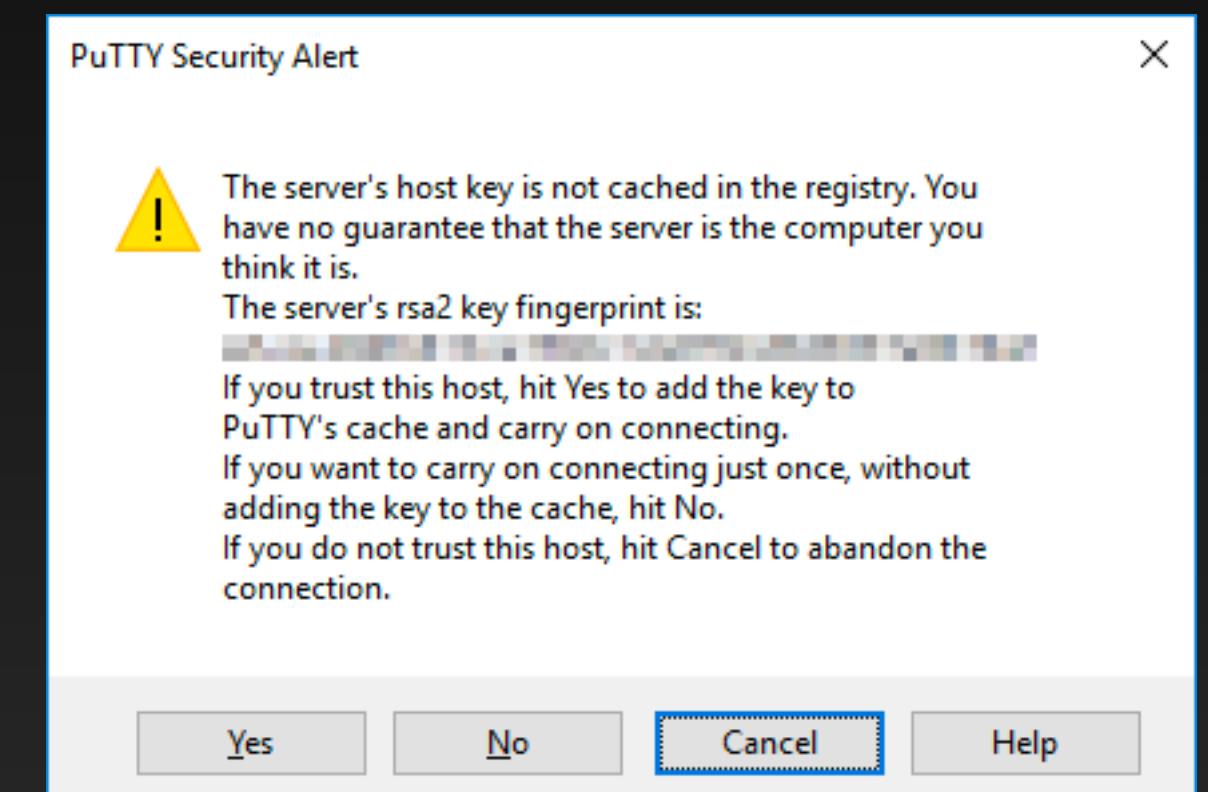
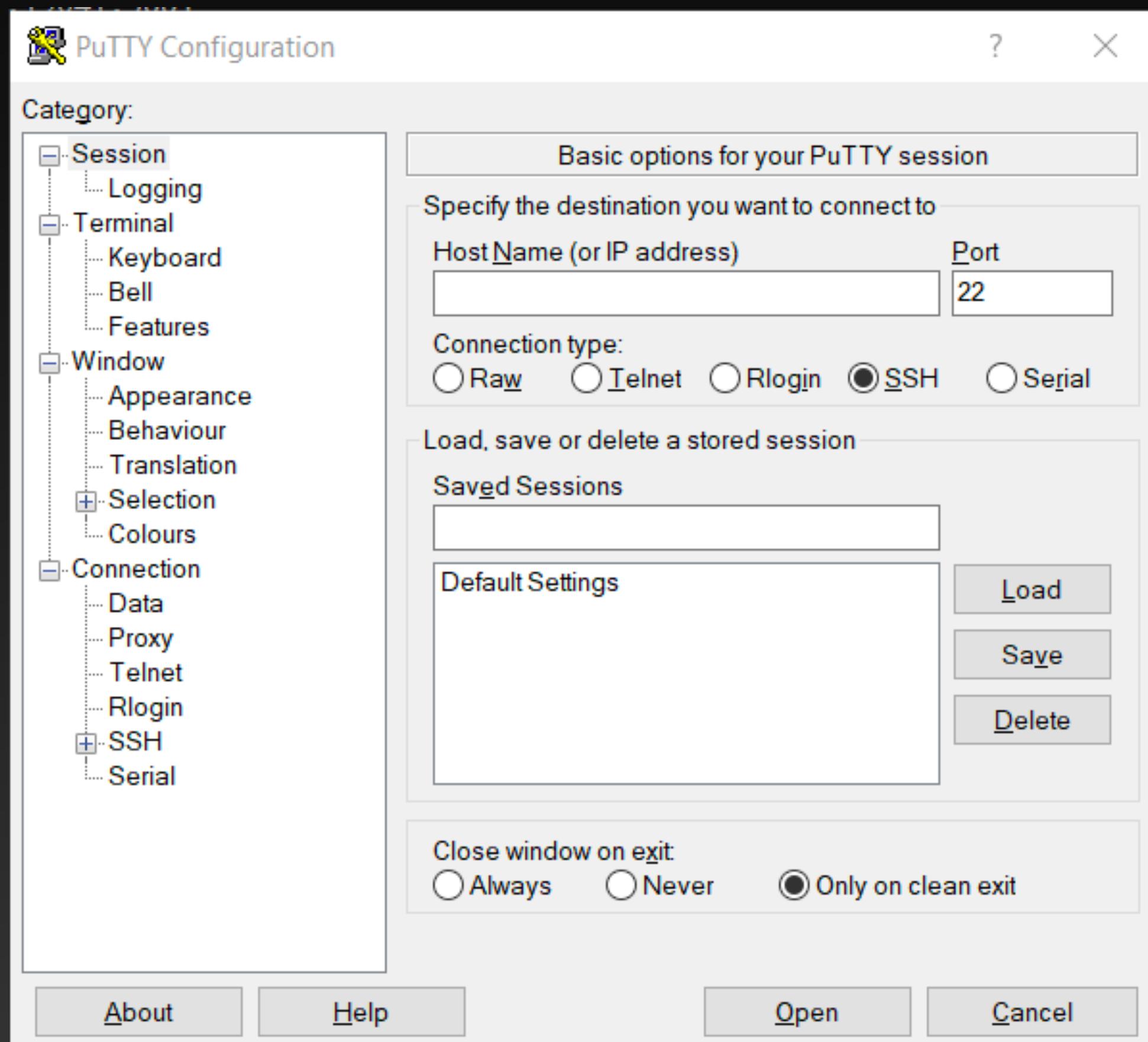
- SSH on Window Machine
- download Putty

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



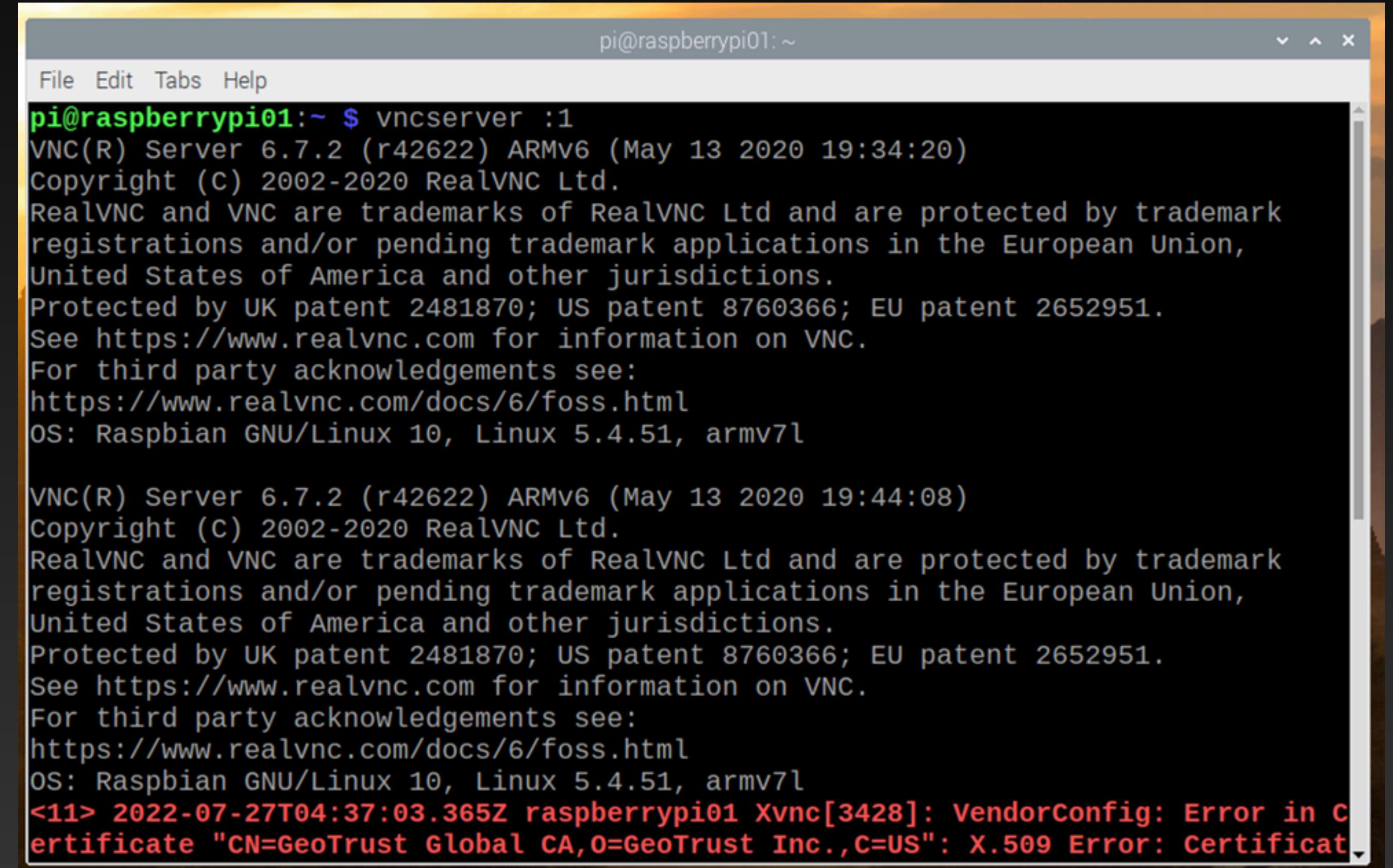
# Remote Access-SSH

- 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



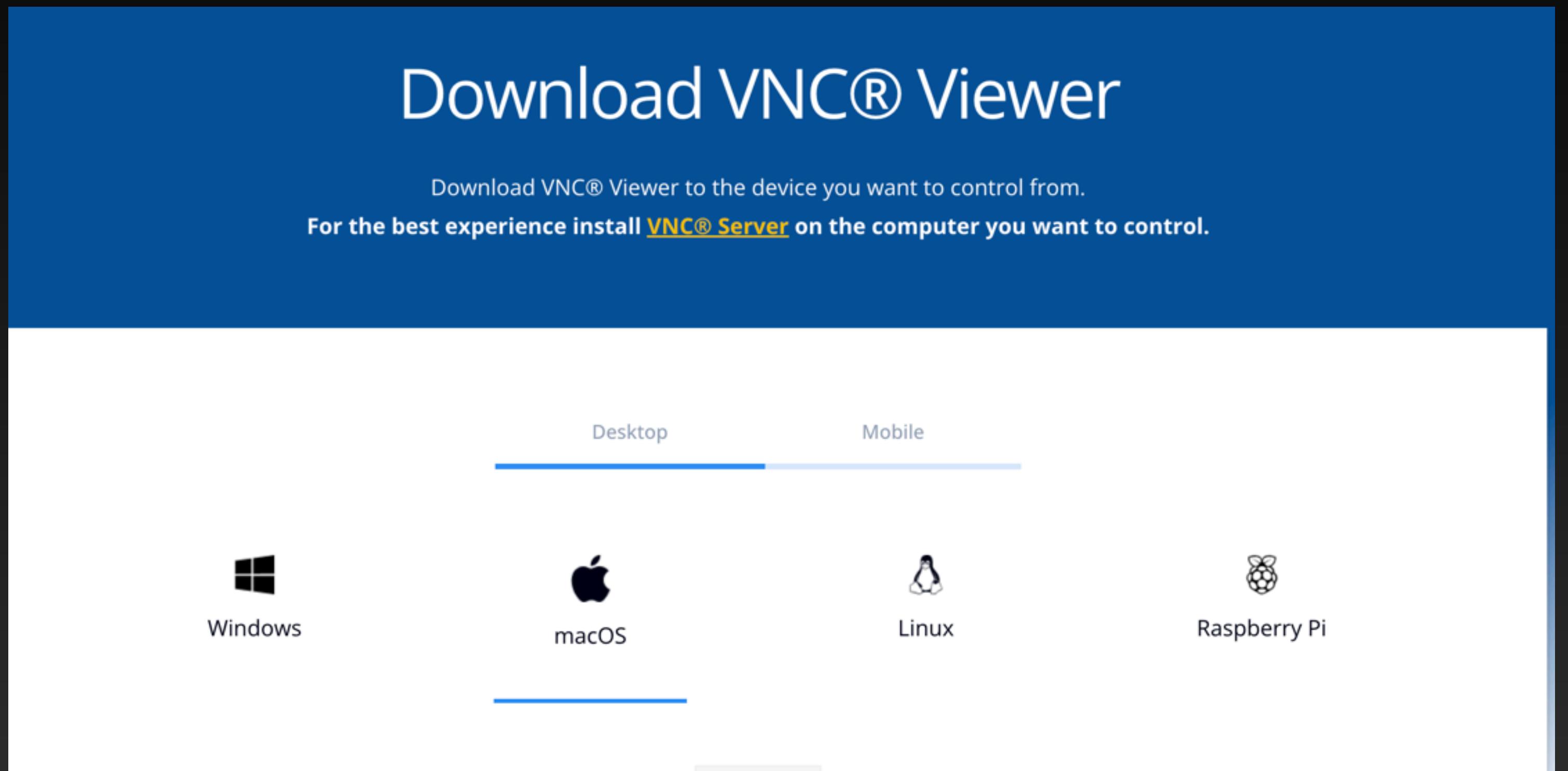
pi@raspberrypi01: ~

```
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
```

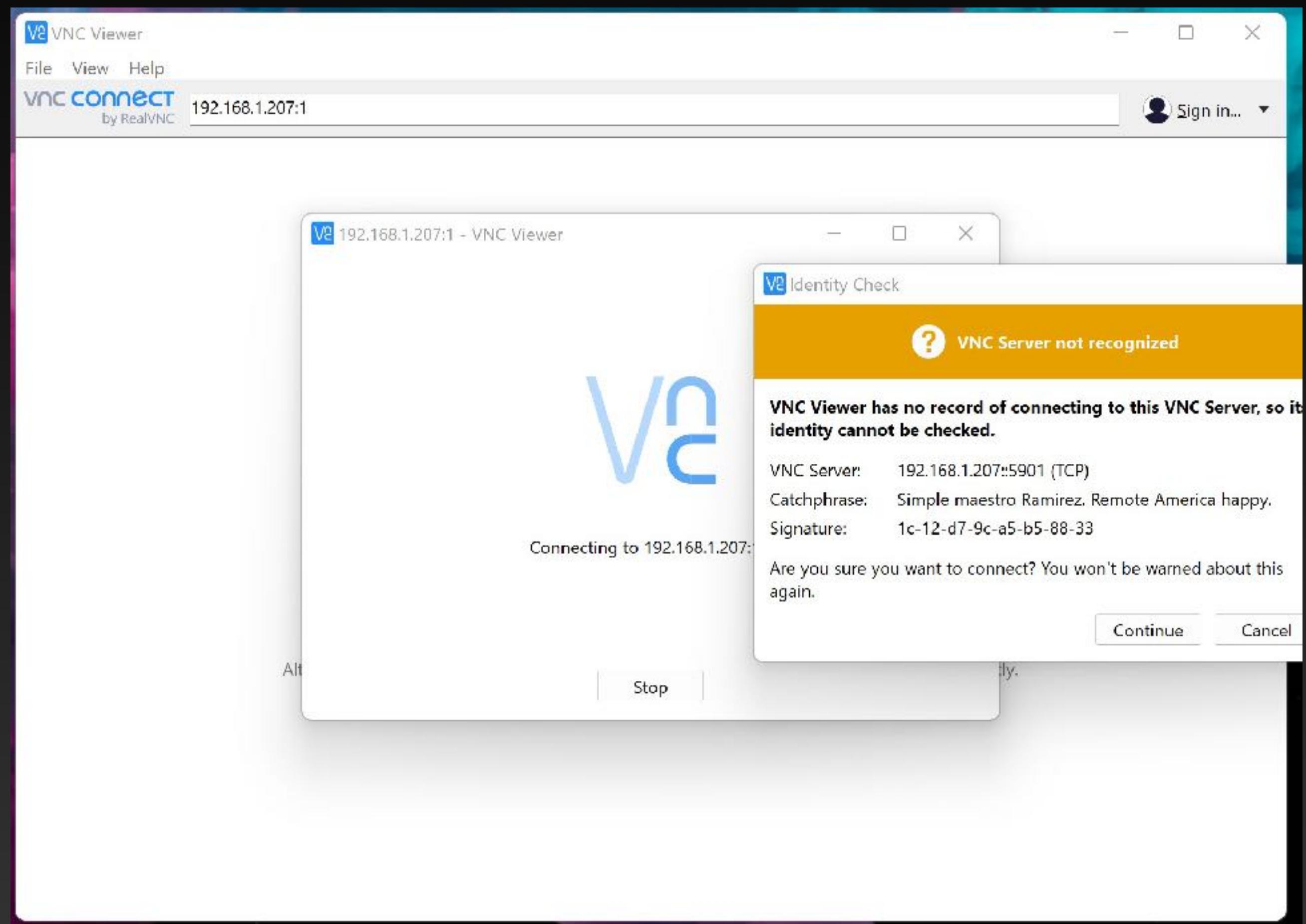
# Remote Access-VNC

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



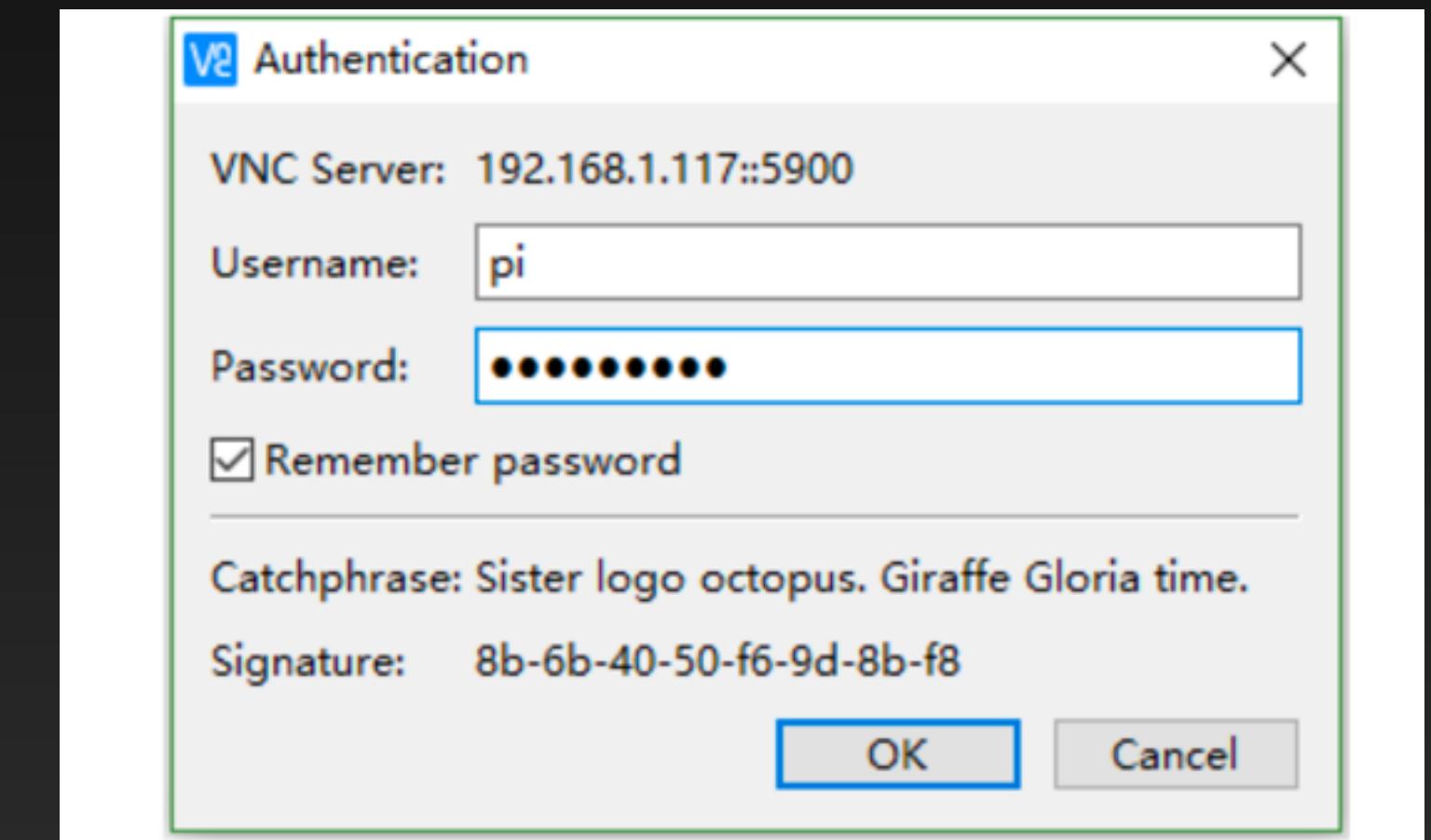
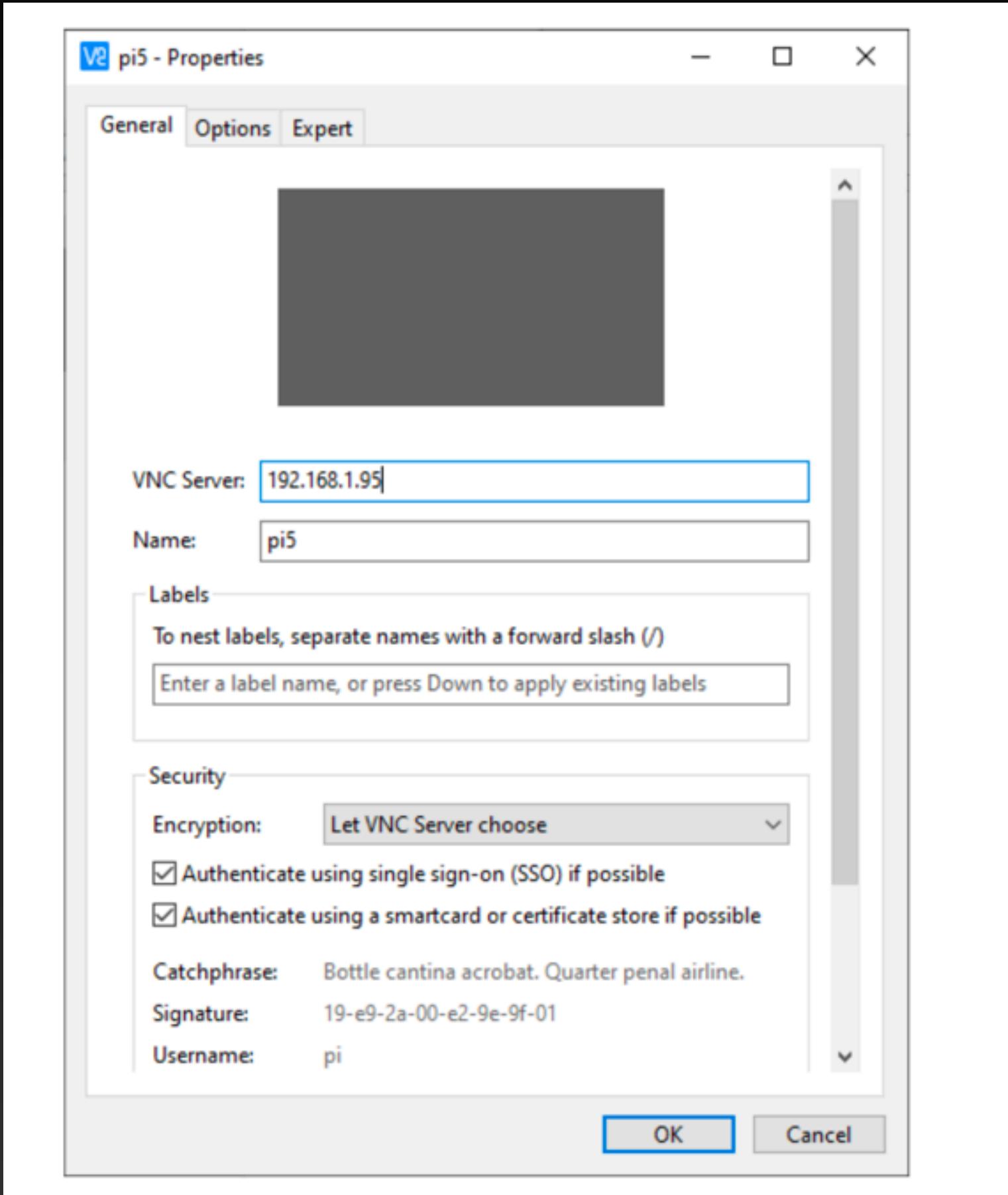
# Remote Access-VNC

- Put your Pi IP address in the installed VNC viewer



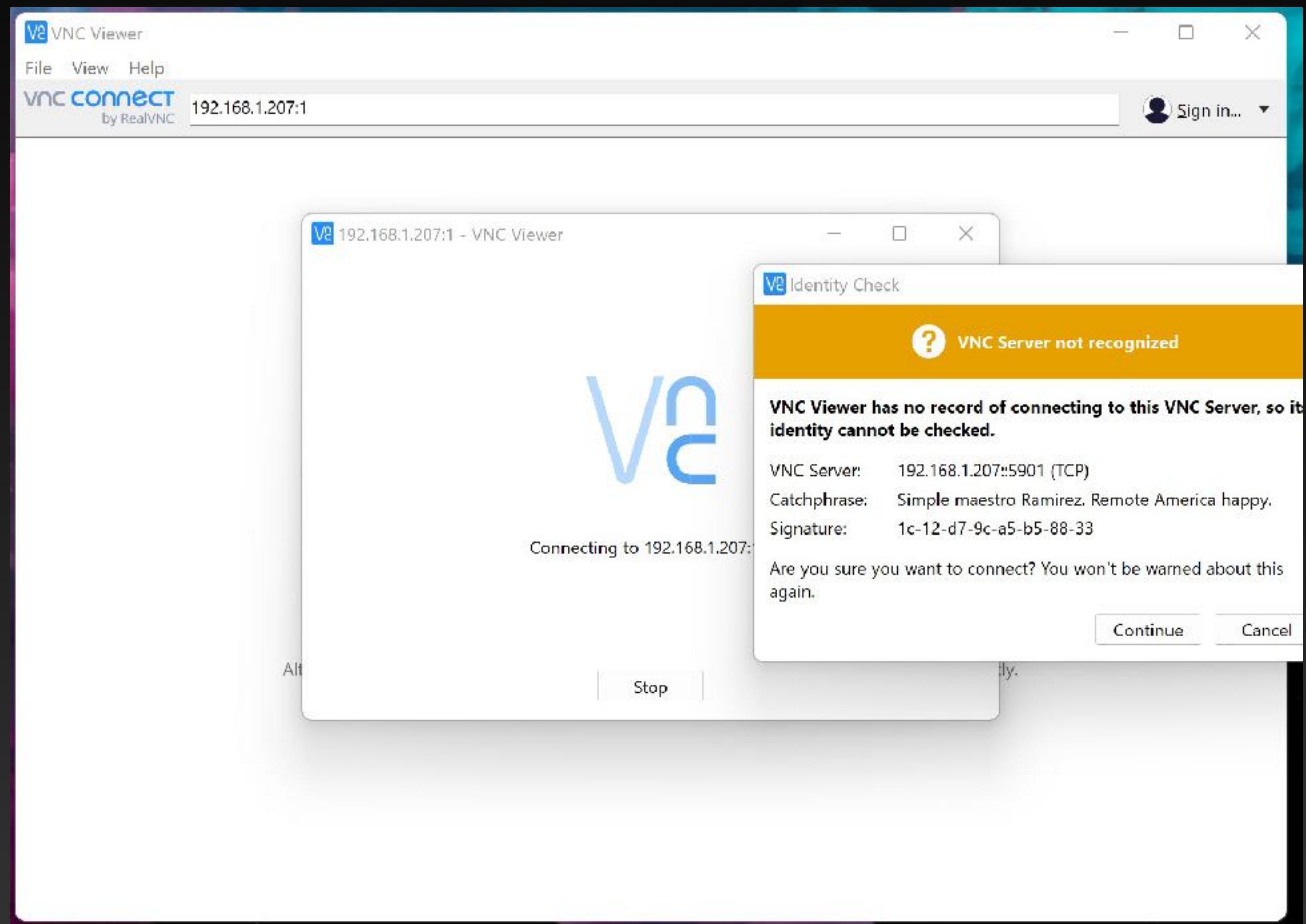
# Remote Access-VNC

- Put your Pi IP address in the installed VNC viewer



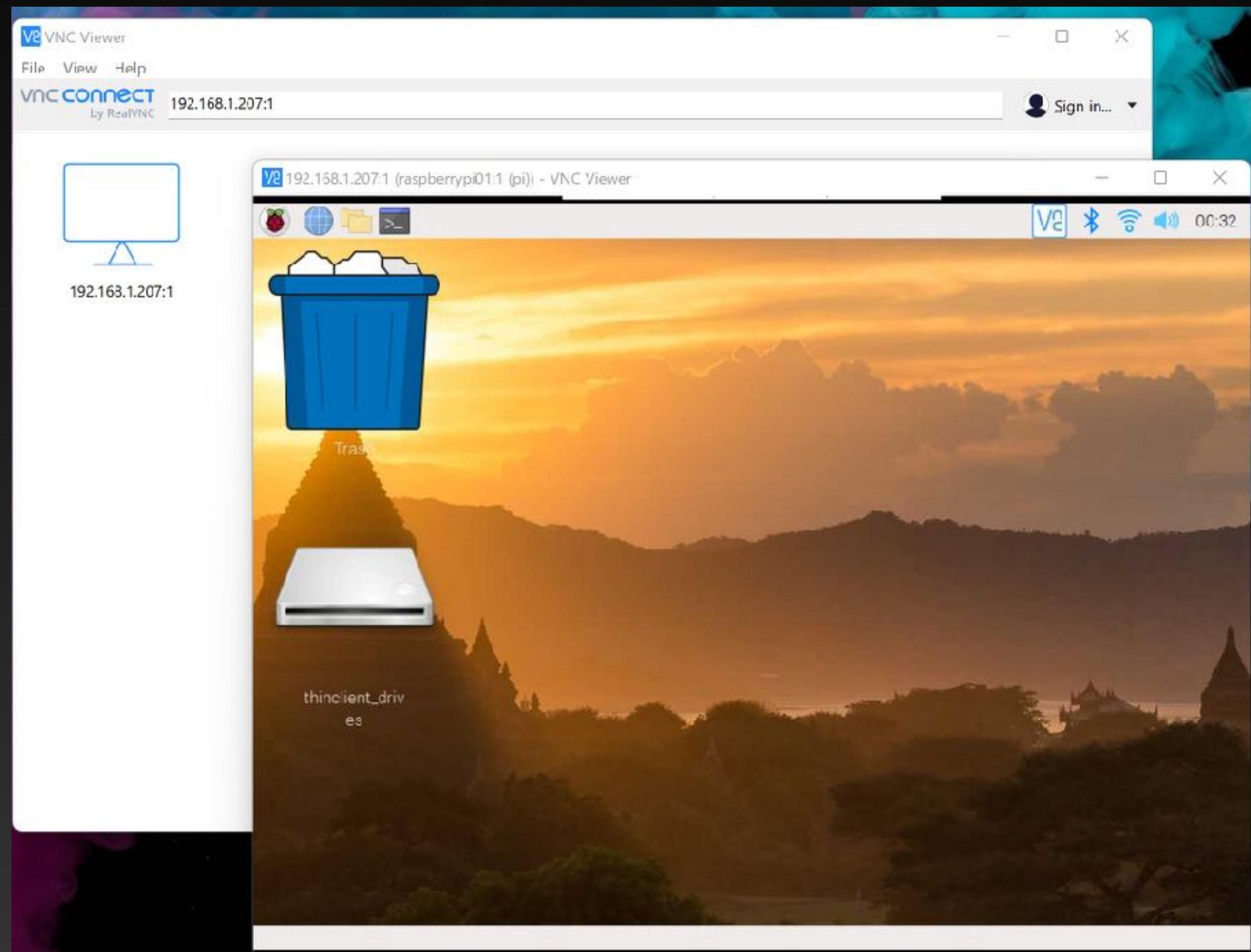
# Remote Access-VNC

- 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](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.

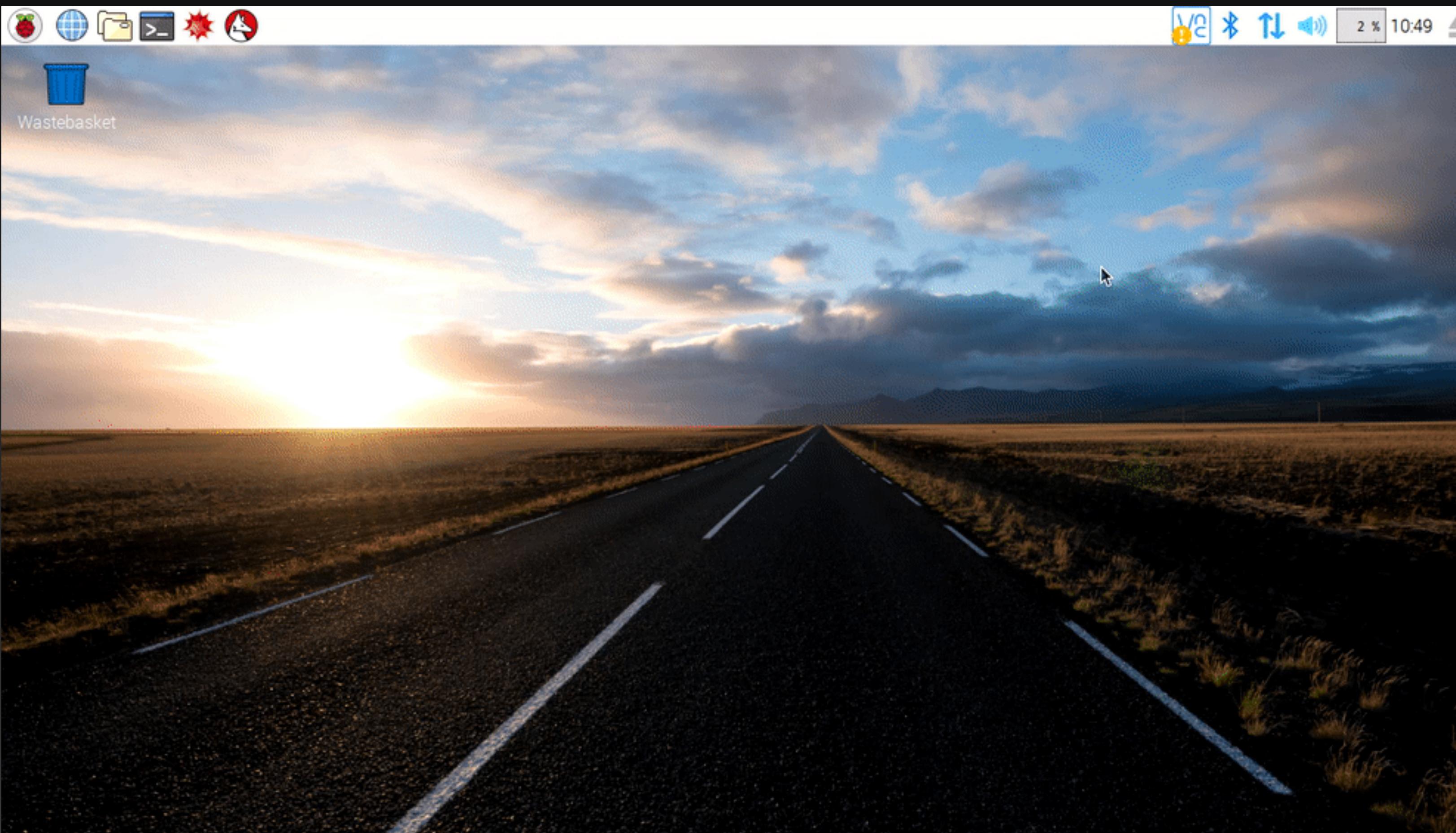
First, in your `home` directory, create a directory called `gpio-music-box`. You will use the new directory to store all your files for the project.

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

# GPIO music box

## Set up your project

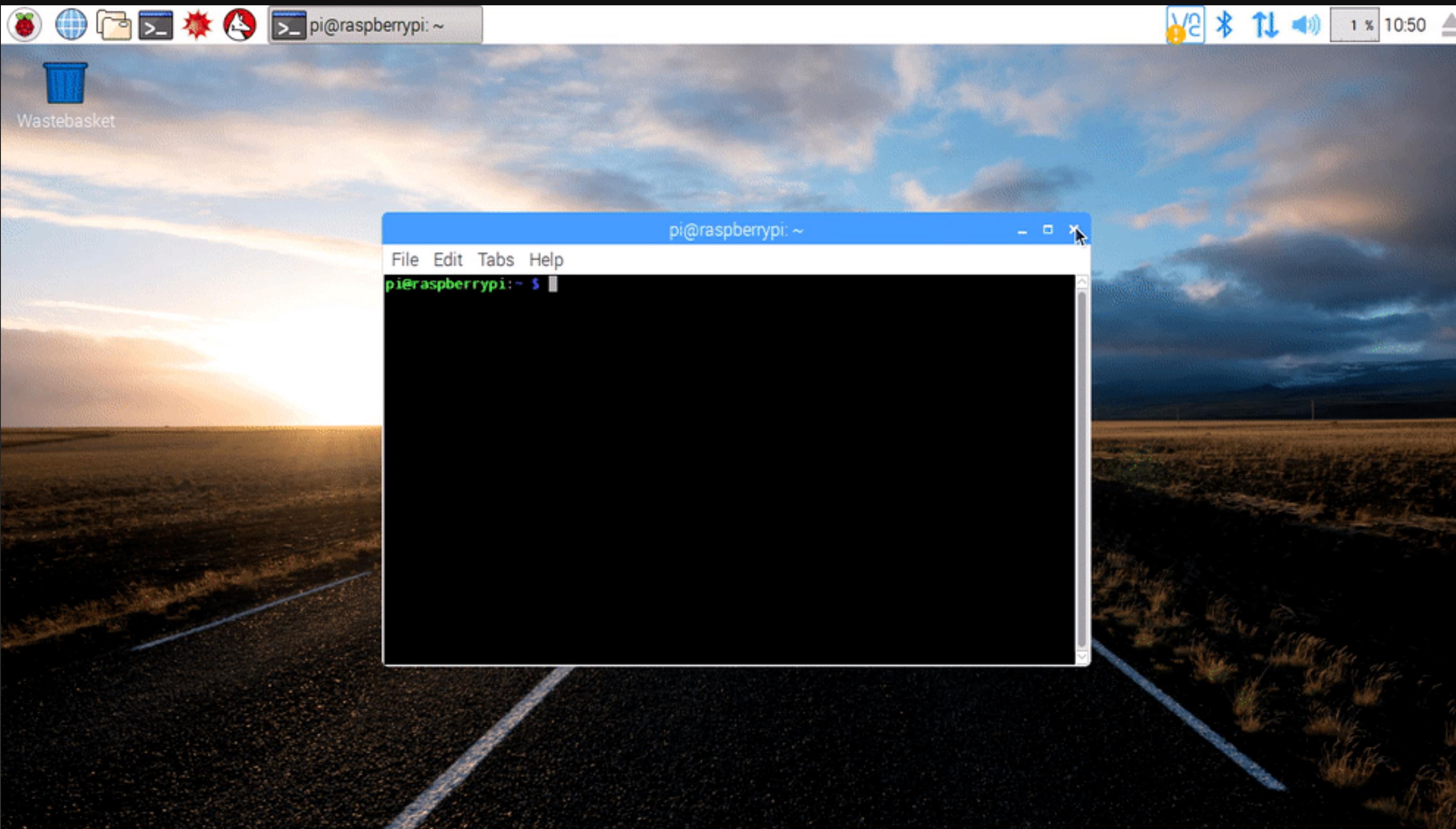
Method 1 - Using the GUI



# 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/sounds/alsa`. 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:

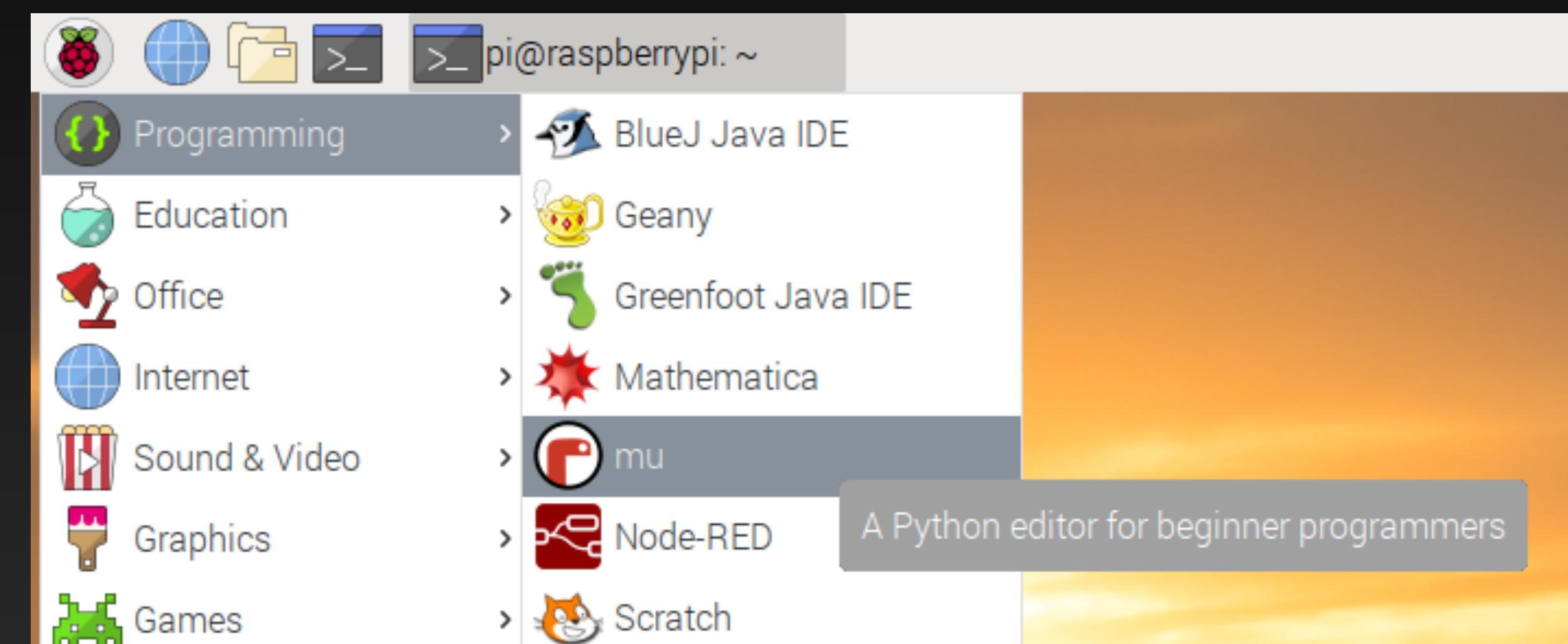
```
cp /usr/share/sounds/alsa/* ~/gpio-music-box/samples/
```

# 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

## Connect your buttons

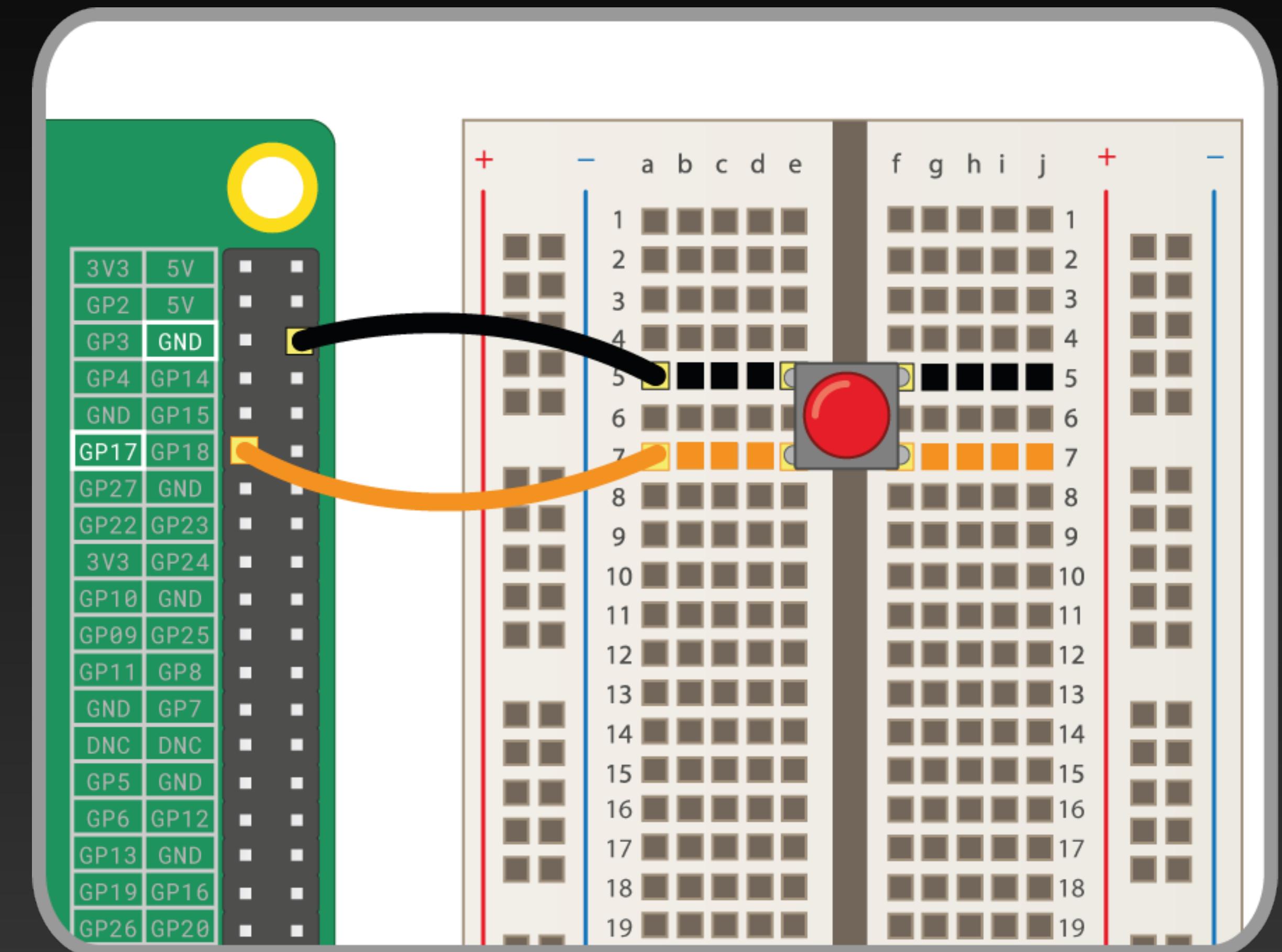
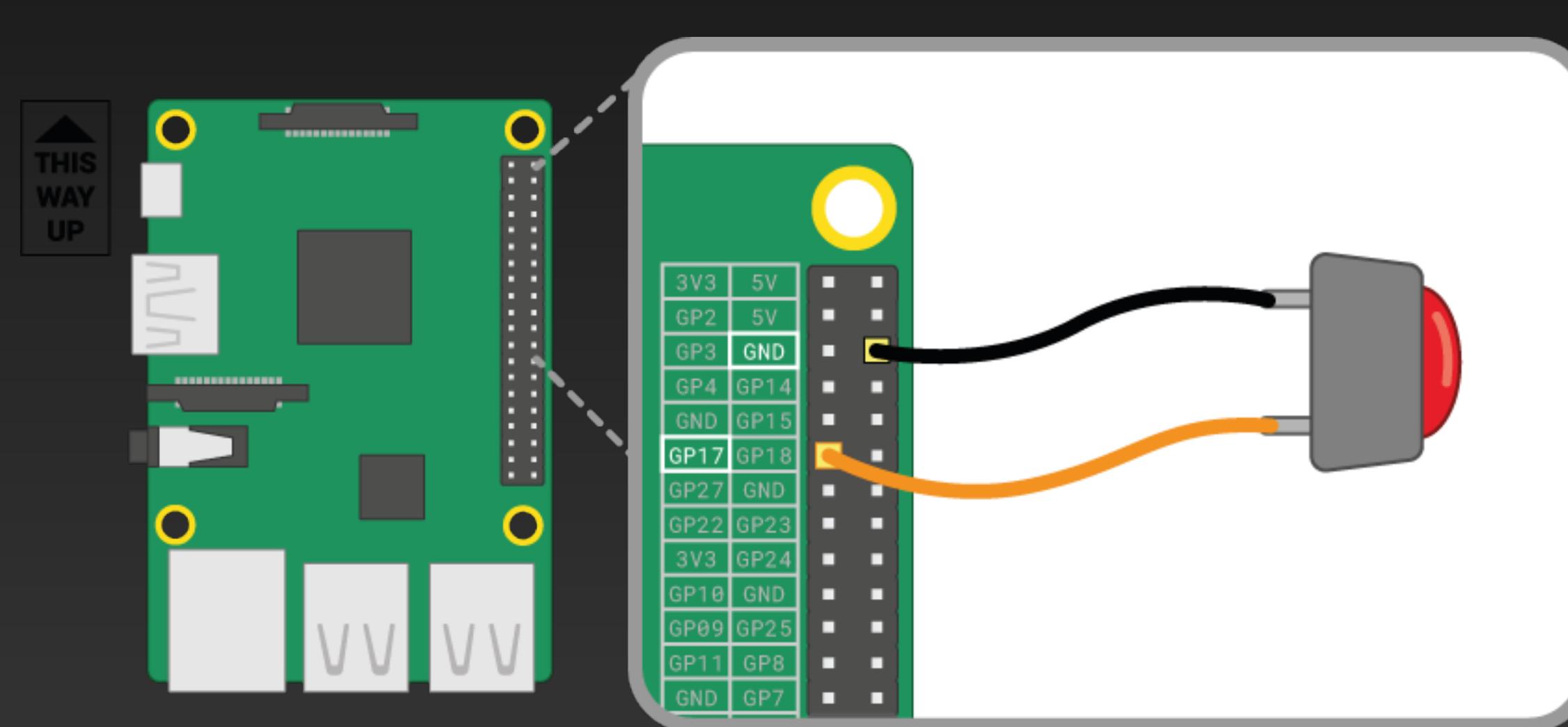
A button is one of the simplest input components you can wire to a Raspberry Pi. It's a non-polarised component, which means you can place it in a circuit either way round and it will work.

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

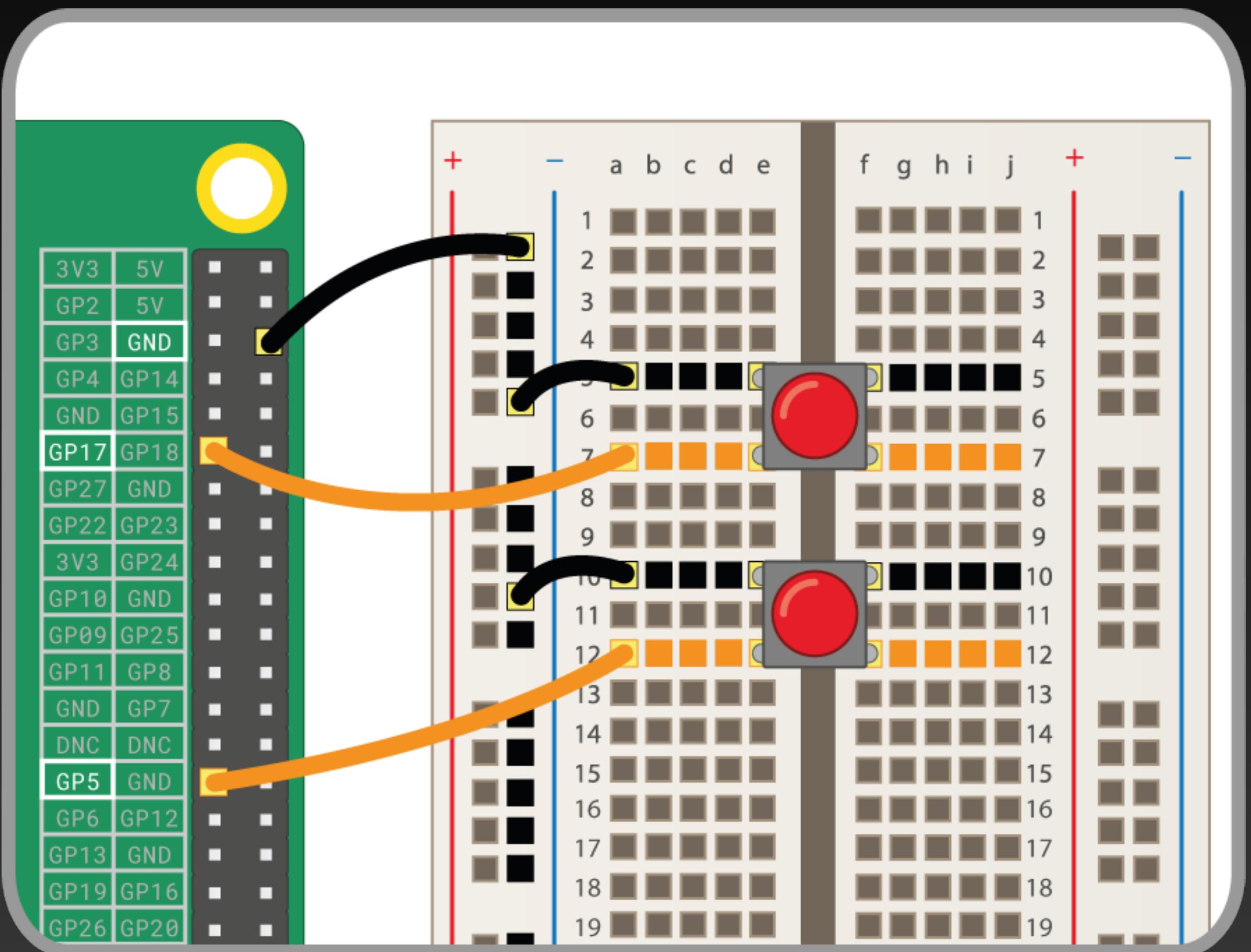
The diagrams below shows how to wire a two-leg or four-leg button to a Raspberry Pi. In both cases, GPIO 17 is the input pin



# GPIO music box

## Connect your buttons

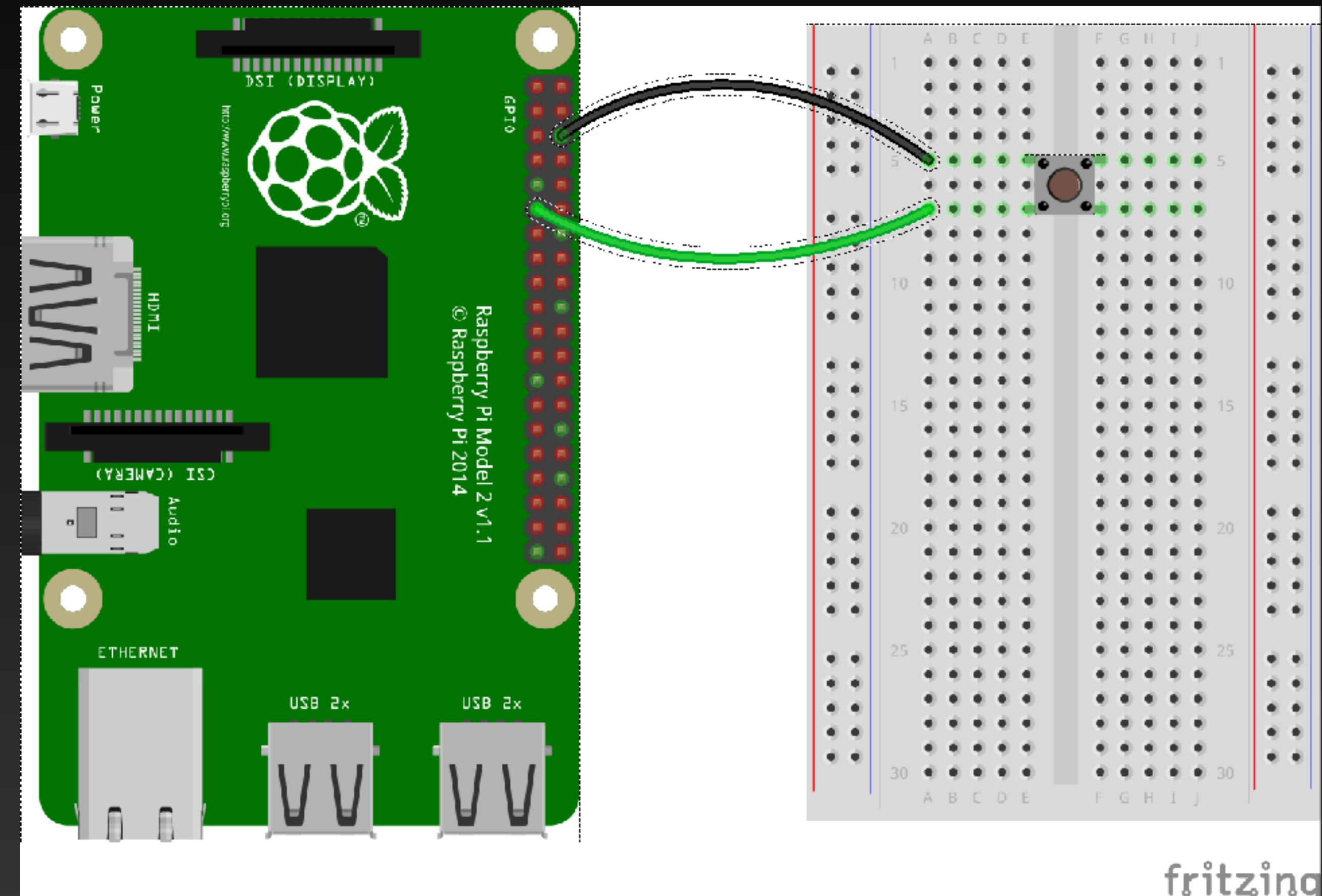
If you are using multiple buttons, then it is often best to use a common ground to avoid connecting too many jumper leads to GND pins. You can wire the negative rail on the breadboard to a single ground pin, which allows all the buttons to use the same ground rail.



# GPIO music box

## Test the button

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



# GPIO music box

## Test the button

```
import pygame
from gpiozero import Button

pygame.init()

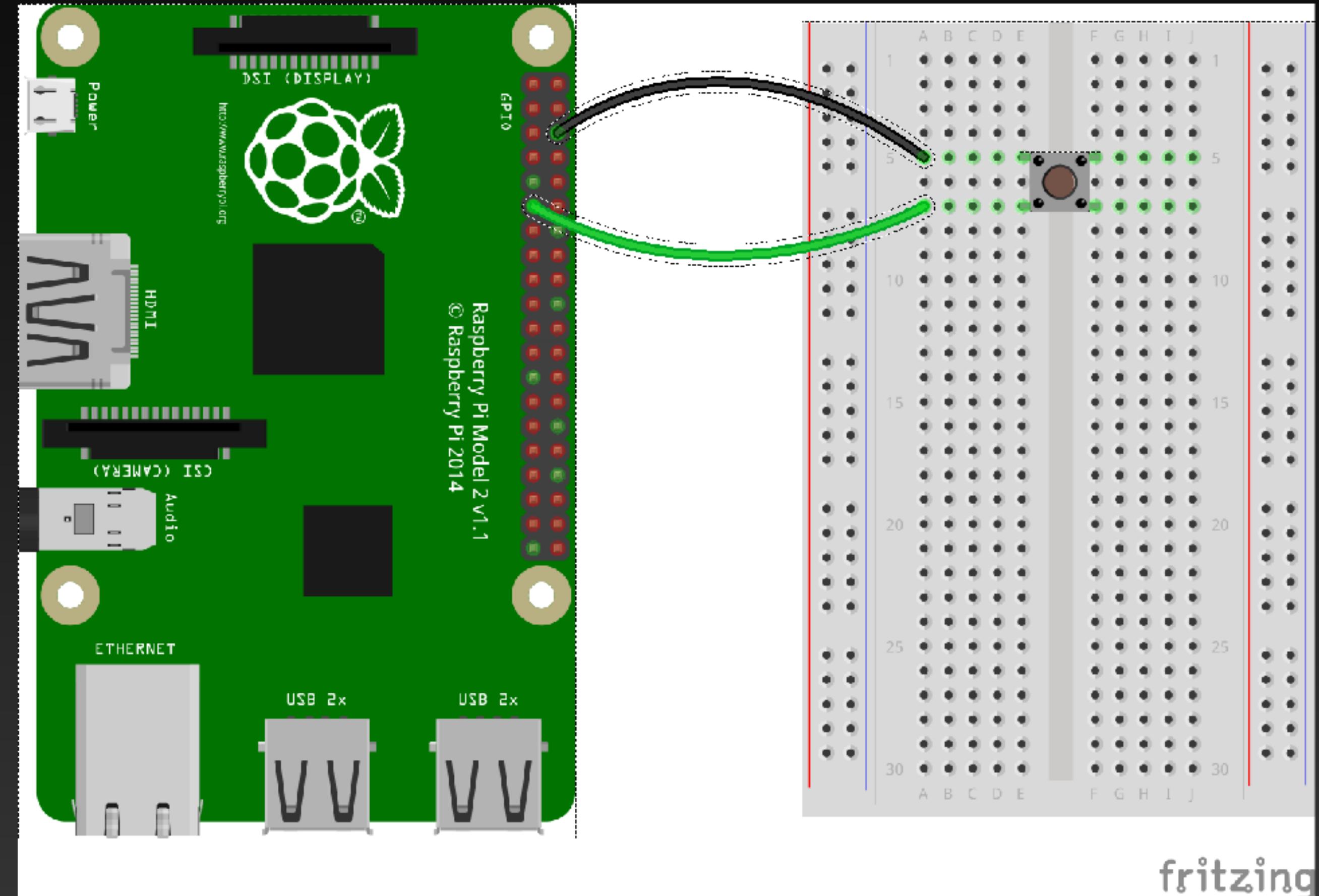
Noise = pygame.mixer.Sound("/home/pi/GPIO-Music-Box/
Noise.wav")

btn_Noise = Button(17)

btn_Noise.when_pressed = Noise.play
```

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

```
btn_Noise.when_pressed =
Noise.play
```



fritzing

# 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.

