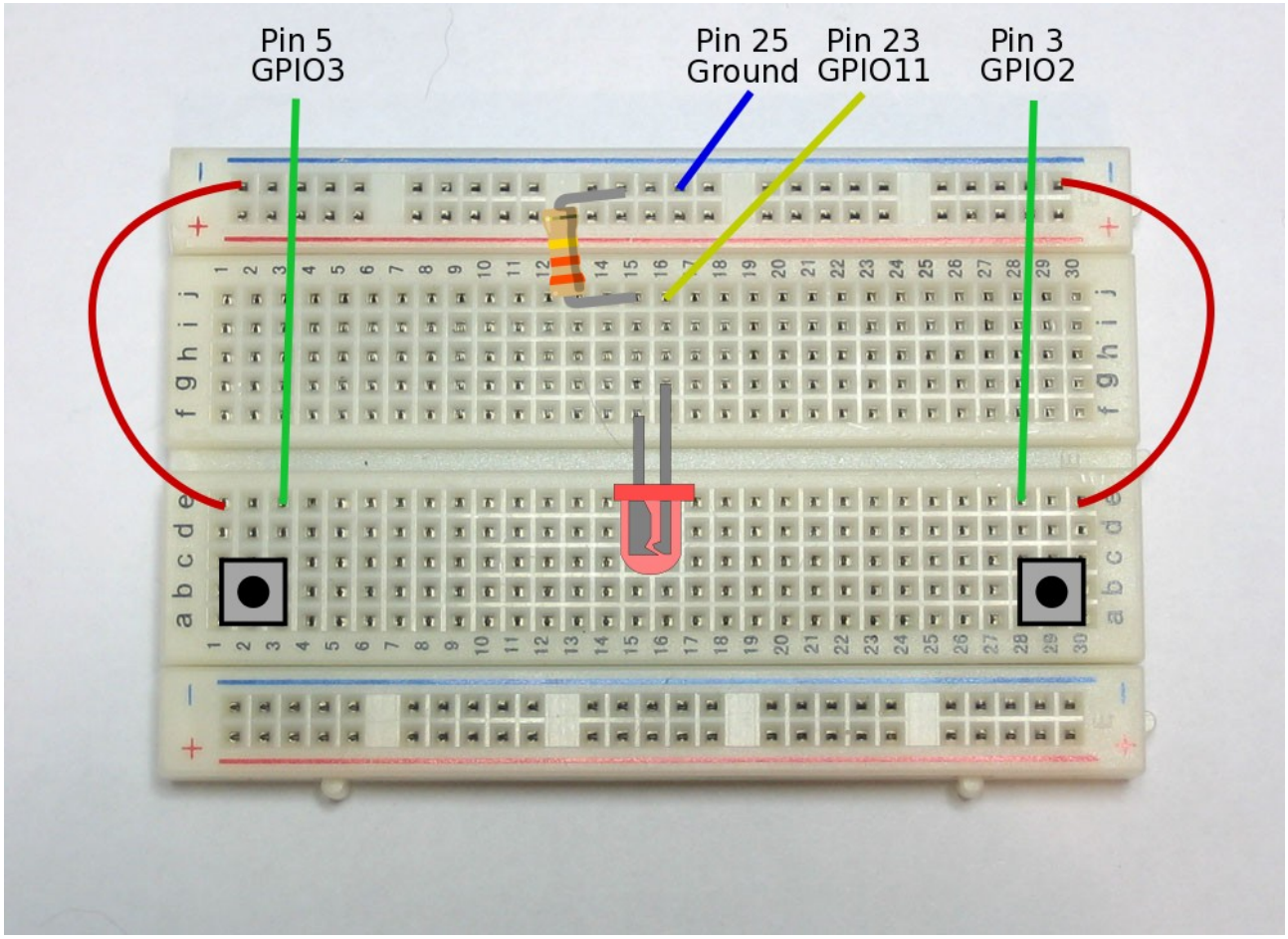
Quick Reaction Game

This week (and possibly next week too) we are going to create a quick reaction game with an LED, two buttons and a bit of Python. The aim of the game will be to be the first person to press the button when the light goes out. First, we will need to put everything together on the **breadboard** (a board with lots of holes that lets you put electrical bits together easily):



Now check that you have got this right. It is possible to damage the Raspberry Pi if you connect the pins incorrectly.

When typing in the code on the following pages you will keep adding to the same file. The parts **in bold** indicate new or changed code. The comments (the lines starting with a '#' character) will make it easier to find the parts of the file that needs changing so please type them all in even though they do not change way the program runs.

Start editing the file with nano:

```
pi@raspberry~ $ nano reaction.py
```

Note, when you put '.py' at the end of the filename nano knows it is a python program and will add colours to help you read it.

1 Controlling the Light

First of all, lets turn the LED on for a few seconds and then turn it off.

```
import RPi.GPIO as GPIO
import time

# Make sure the GPIO pins are ready
GPIO.cleanup()
GPIO.setmode(GPIO.BOARD)

# Choose which GPIO pins to use
led = 23

# Set the LED as an output
GPIO.setup(led, GPIO.OUT)

# Turn the LED on
GPIO.output(led, 1)
# Wait for 5 seconds
time.sleep(5)
# Turn the LED off
GPIO.output(led, 0)

GPIO.cleanup()
```

Now save the file and run it to see what happens:

```
pi@raspberry~ $ sudo python reaction.py
```

If the LED doesn't come on for 5 seconds, go back see if you can out what is wrong. Make sure you try this after every new bit of code.

The object of the game is going to be to see who can press the button first when the light goes out so it would be better if the length of time it stayed on for were random. The code below shows how to do this. Remember that the **bold** parts are new or changed and the rest is what you have already written. I have missed some bits out (shown by '...') to save space.

```
import RPi.GPIO as GPIO
import time
import random

...

# Turn the LED on
GPIO.output(led, 1)
# Wait for a random length of time, between 5 and 10 seconds
time.sleep(random.uniform(5, 10))
# Turn the LED off
GPIO.output(led, 0)
...
```

2 Detecting the Buttons

So far, so good. Now we want to detect when a button is pressed. The way we will do this is to have a loop that keeps going until one of the buttons is pressed.

One odd thing is that the buttons are **on** if they **are not** pressed and **off** when they **are**. This is why the code says 'Left button pressed' when it finds that 'leftButton' is 'False'.

```
...

# Choose which GPIO pins to use
led = 23
rightButton = 3
leftButton = 5

# Set the buttons as inputs and the LED as an output
GPIO.setup(led, GPIO.OUT)
GPIO.setup(rightButton, GPIO.IN)
GPIO.setup(leftButton, GPIO.IN)

...

# Turn the led off
GPIO.output(led, 0)
# Wait until a button has been pressed
while GPIO.input(leftButton) and GPIO.input(rightButton):
    pass # Do nothing!
# See if the left button has been pressed
if GPIO.input(leftButton) == False:
    print “Left button pressed”
# See if the right button has been pressed
if GPIO.input(rightButton) == False:
    print “Right button pressed”

GPIO.cleanup()
```

Notice that the line after 'while' is **indented** (it has spaces at the start). Python knows which lines are in the loop (and also for the 'if' blocks) by how far they are indented, so make sure you put the spaces in correctly. It looks best if you use 4 spaces for each indentation.

Wouldn't it be better if it told you who has won instead of just which button was pressed? For this, we need to find out the players names.

```
import RPi.GPIO as GPIO
import time
import random

# Find out the names of the players
leftName = raw_input(“What is the left player's name? “)
rightName = raw_input(“What is the right player's name? “)
# Put the names in a list
names = [ leftName, rightName ]
```

```
...

# See if the left button has been pressed
if GPIO.input(leftButton) == False:
    print names[0] + " won"
# See if the right button has been pressed
if GPIO.input(rightButton) == False:
    print names[1] + " won"

GPIO.cleanup()
```

The two 'print' lines near the end do exactly the same thing but with a different name. We can use a **function** to avoid having to repeat the same bit of code again and again. While we are doing this, lets add variables for scores for each player, which we will use later.

```
import RPi.GPIO as GPIO
import time
import random

# Say who won
def winGame(player):
    print names[player] + " won!"
    scores[player] += 1

# Find out the names of the players
leftName = raw_input("What is the left player's name? ")
rightName = raw_input("What is the right player's name? ")
# Put the names in a list
names = [ leftName, rightName ]
# Set a score for each player
scores = [ 0, 0 ]

...

# See if the left button has been pressed
if GPIO.input(leftButton) == False:
    winGame(0)
# See if the right button has been pressed
if GPIO.input(rightButton) == False:
    winGame(1)

GPIO.cleanup()
```

3 Making the Game Better

Now try this – while the LED is on keep one of the buttons pressed. Do you see that the player wins immediately when the light goes out? That isn't very fair, is it? Lets make it so that if a player just keeps the button pressed then he or she automatically loses instead.

Remember in Scratch, we had a block called 'If ... Else ...' that meant we could choose to do one of

two things? In Python, we can do the same but it is possible to choose between more options. In this case we will do one thing if the left button is pressed, another thing if the right button is pressed and then something else if neither are.

Notice that you have already type everything after the line saying 'else' but you need to put spaces at the start of each line so that Python knows that it is all in the 'else' section.

```
...

# Turn the led on
GPIO.output(led, 1)
# Wait for between 5 and 10 seconds
time.sleep(random.uniform(5, 10))
# Check to see if a button is pressed
# If so, the other player wins
if GPIO.input(leftButton) == False:
    winGame(1)
elif GPIO.input(rightButton) == False:
    winGame(0)
else:
    # Turn the led off
    GPIO.output(led, 0)
    # Wait until a button has been pressed
    while GPIO.input(leftButton) and GPIO.input(rightButton):
        pass # Do nothing!
    # See if the left button has been pressed
    if GPIO.input(leftButton) == False:
        winGame(0)
    # See if the right button has been pressed
    if GPIO.input(rightButton) == False:
        winGame(1)

GPIO.cleanup()
```

Finally, lets change it so that you can play more than one game. It may look like there is a lot to type in but for most of the lines after 'for game in range(games)' all you need to do is add some spaces at the start of lines so that Python knows that it is all in the loop..

```
...

# Find out the name of the players
leftName = raw_input("What is the left player's name? ")
rightName = raw_input("What is the right player's name? ")
games = int(raw_input("How many games do you want to play? "))

...

# Set the buttons as inputs and the light as an output
GPIO.setup(led, GPIO.OUT)
GPIO.setup(rightButton, GPIO.IN)
GPIO.setup(leftButton, GPIO.IN)
```

```
# Play all the games
for game in range(games):
    # Turn the led on
    GPIO.output(led, 1)
    # Wait for between 5 and 10 seconds
    time.sleep(random.uniform(5, 10))
    # Check to see if a button is pressed
    # If so, the other player wins
    if GPIO.input(leftButton) == False:
        winGame(1)
    elif GPIO.input(rightButton) == False:
        winGame(0)
    else:
        # Turn the led off
        GPIO.output(led, 0)
        # Wait until a button has been pressed
        while GPIO.input(leftButton) and GPIO.input(rightButton):
            pass # Do nothing!
        # See if the left button has been pressed
        if GPIO.input(leftButton) == False:
            winGame(0)
        # See if the right button has been pressed
        if GPIO.input(rightButton) == False:
            winGame(1)

print "Scores:"
print "  " + names[0] + ": " + str(scores[0])
print "  " + names[1] + ": " + str(scores[1])

GPIO.cleanup()
```

4 Challenges

- Can you add two extra lights, one next to each button, to show who won each time? Hint: Use pins 19 and 21. You will need two more LEDs and two more resistors.
- Can you find out how long it took for the players to press the button after the LED turned off? Maybe you could get more points the faster you press the button.

5 Code Listing

Here is the whole code in one place. You can check to see if your file looks like this.

```
import RPi.GPIO as GPIO
import time
import random

# Say who won and add one to the score
def winGame(player):
    print names[player] + " won!"
    scores[player] += 1

# Find out the name of the players
leftName = raw_input("What is the left player's name? ")
rightName = raw_input("What is the right player's name? ")
games = int(raw_input("How many games do you want to play? "))
# Put the names in a list
names = [ leftName, rightName ]
# Set a score for each player
scores = [ 0, 0 ]

# Make sure the GPIO pins are ready
GPIO.cleanup()
GPIO.setmode(GPIO.BOARD)

# Choose which GPIO pins to use
led = 23
rightButton = 3
leftButton = 5

# Set the buttons as inputs and the LED as an output
GPIO.setup(led, GPIO.OUT)
GPIO.setup(rightButton, GPIO.IN)
GPIO.setup(leftButton, GPIO.IN)

# Play all the games
for game in range(games):
    # Turn the LED on
    GPIO.output(led, 1)
    # Wait for between 5 and 10 seconds
    time.sleep(random.uniform(5, 10))
    # Check to see if a button is pressed
    # If so, the other player wins
    if GPIO.input(leftButton) == False:
        winGame(1)
    elif GPIO.input(rightButton) == False:
        winGame(0)
    else:
```

```
        # Turn the LED off
        GPIO.output(led, 0)
        # Wait until a button has been pressed
        while GPIO.input(leftButton) and GPIO.input(rightButton):
            pass # Do nothing!
        # See if the left button has been pressed
        if GPIO.input(leftButton) == False:
            winGame(0)
        # See if the right button has been pressed
        if GPIO.input(rightButton) == False:
            winGame(1)

print "Scores:"
print "  " + names[0] + ": " + str(scores[0])
print "  " + names[1] + ": " + str(scores[1])

GPIO.cleanup()
```