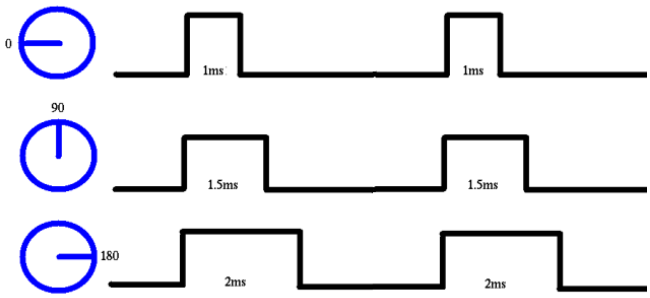
Raspberry Pi Lab Number 1

Servo angles are controlled through a process known as pulse width modulation (PWM). Pulse width modulation varies the percentage of time the signal is at logic 1 during the period of the signal. The numbered pins of the Raspberry Pi can be used for PWM. By varying the width of pulses on their control line, servos can be set to any angle within their range of motion. The length of the pulse is also known as the duty cycle of the signal. A typical servo motor expects to be updated every **20 ms** with a pulse between 1 ms and approximately 2 ms, or in other words, between a 5 and 10% duty cycle on a **50 Hz** waveform. These values are approximate and each servo should be tested for its limits before use. The image below shows typical values for some angles.

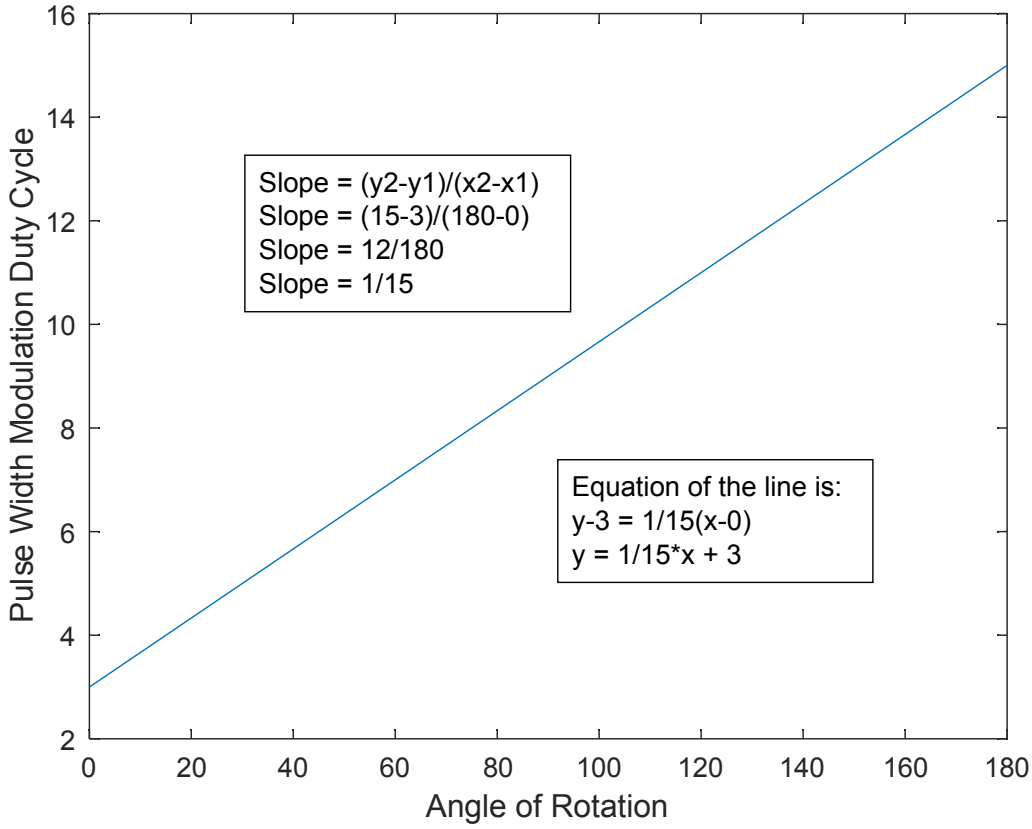


Testing our Servos to find limits:

```
import RPi.GPIO as GPIO
pin = 18
farLeft = 1
farRight = 15
GPIO.setmode(GPIO.BCM)          # Set the pin numbering to those on the board
GPIO.setup(pin, GPIO.OUT)       # Set the chosen pin(18) to output mode
pwm = GPIO.PWM(pin,50)          # Set up a 50Hz PWM signal on the chosen pin
pwm.start(5)
pwm.ChangeDutyCycle(farLeft)    #  You should only use one of these two lines at
pwm.ChangeDutyCycle(farRight)   #  a time.  Find the left limit and then change to
                                #  the right.
```

Change farLeft up by one until the servo changes position. Then use the last number that didn't move as the farLeft duty cycle.

Change farRight down by one until the servo changes position. Then use the last number that didn't move as the farRight duty cycle.



These duty cycles represent 0 and 180 (or as far as the motor will rotate. You can estimate the maximum angle if it is over or under 180) degrees on the servo motor.

Plot the two points and find the slope of the resulting line.

The equation of this line can then be used to calculate the correct duty cycle for any angle between 0 and 180 degrees.

All servo motors are different so this should be done for each motor used.

Duty Cycle = slope * angle + offset

Duty Cycle = 1.0/15.0 * angle + 3 in the example above

Connect the motor to the breadboard.

Caution: Reversing the current through a servo motor can cause damage to the motor. Please take care when making these connections.

1. The red wire should be connected to the +5v column of the breadboard

2. The black wire should be connected to the -5v column of the breadboard
3. The white wire (this wire may be any color other than red or black but is usually white or yellow) is the control wire and should be connected to pin number 18 if using the example setup above. Any of the ***numbered*** pins may be used. **Do not connect to any pin labeled 5V or 3V3.**

You can now run the code segment above to find the limits of your motor.

When finished, wire up the two buttons as follows.

1. Place the button across the channel in the center of the board
2. Connect the upper contact to the -5V column of the breadboard
3. Connect the lower contact to one of the numbered pins so that the digital data can be read
4. Connect the lower contact to the +5V column of the breadboard using a 10K Ohm resistor.

This completes the electrical connections necessary for this lab.

Having found the limits of the motor and constructed the described circuit using the Raspberry pi and its connected breadboard, you are to write python code so that the motor will begin at 90 degrees and the buttons will add or subtract from this angle so that the motor can be set in any position. You can use the code above as a starting point for your code.