```
#Max Huggins
#1/22/19
#Blinking LED with threading

import RPi.GPIO as GPIO
import time
import threading

#Here's where you would set up the GPIO pins
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(12,GPIO.IN)
GPIO.setup(16,GPIO.IN)
GPIO.setup(37,GPIO.OUT)

#Defining function that will run parallel to while loop from thread
def butt_change():
    while True: #runs infinitely in parallel with the while loop in the try: section
        butt_1 = GPIO.input(12)#Check to see what's up with the buttons
        butt_2 = GPIO.input(16)
        global rate #Make rate available to everywhere in code
        if butt_1 == False: #If button 1 is pressed then increase rate 1.1x
            rate = rate*1.1
            print('The new rate is:', rate)
        if butt_2 == False: #If button 2 is pressed then decrease rate .90x
           if rate <= .005: #if it is blinking this fast you can't see much blinking so don't decre
               print('The LED cannot blink any faster!')
           else:
               rate = rate*.9
               print('The new rate is:', rate)

try:
    y = threading.Thread(target = butt_change) #This is initializing the thread to attack that butt
    rate = 1 #initializing my rate
    y.start() #This starts the thread's attack on the function. (Runs the function)
    while True: #This infinite loop runs parallel to the thread such that you can click buttons reg
        GPIO.output(37, True)
        time.sleep(rate)
        GPIO.output(37,False)
        time.sleep(rate)

except KeyboardInterrupt:
    #Exits when you press CTRL+C
    print('The pancakes are burnt...')

except:
    #This catches ALL other exceptions including errors.
    #You won't get any error messages for debugging
    #so only use it once your code is working
    print("Other error or exception occured!")

finally:
    GPIO.cleanup() #this ensures a clean exit
```