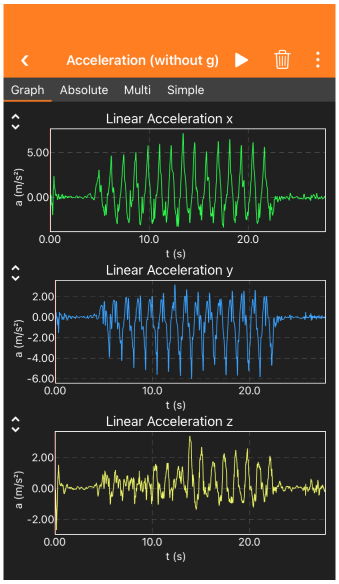
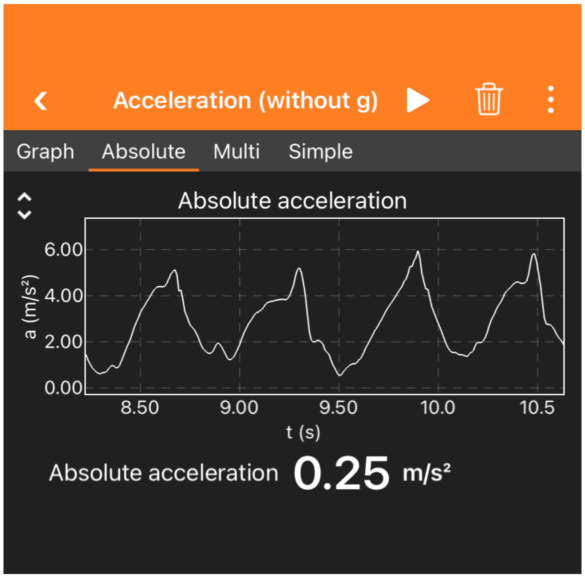
# Step Counter

## Choose the sensor

For this assignment we decided to make use of the phyphox smartphone app to help us record x, y, and z, and absolute acceleration data when walking as shown in the screenshots below. We decided to use this phyphox sensor because it uses the accelerometer that is already built into our phone device and allows us to make use of the sensor. The sample rate for this assignment is 100 Hz based off of the phyphox app. Phyphox Sensor database info located at this link: [https://phyphox.org/sensordb/](https://phyphox.org/sensordb/) states the accelerometer sampling rate per second for various mobile devices





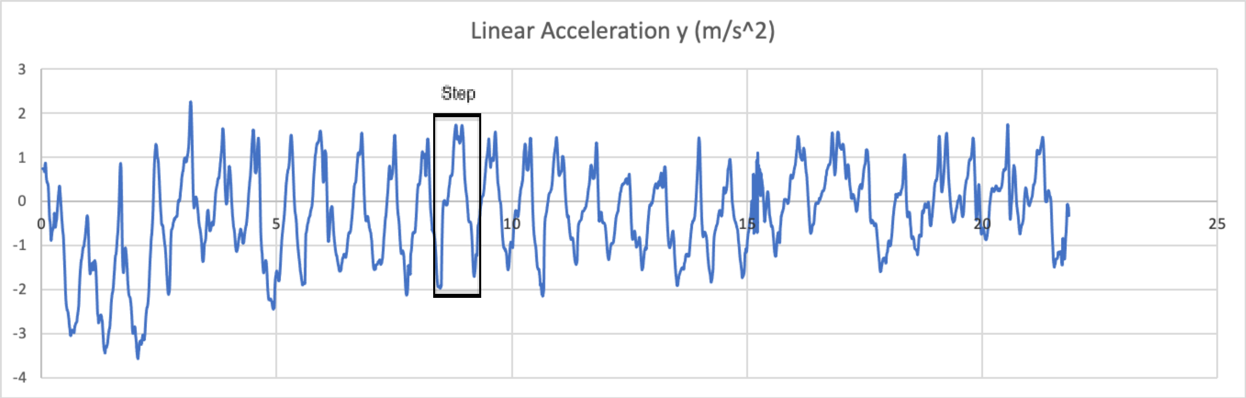
Absolute acceleration 0.25 m/s²

## Choose the location to place the sensor

The location that we have decided to place the sensor in is in our hand while walking as it felt most natural to us. We felt like this made the best sense to us because sometimes people may not necessarily have pockets to place their phone in and may just be carrying their phone in their hand. We also chose to place the phone sensor in our hands because typically as a person walks, their hands are moving or swaying side to side.

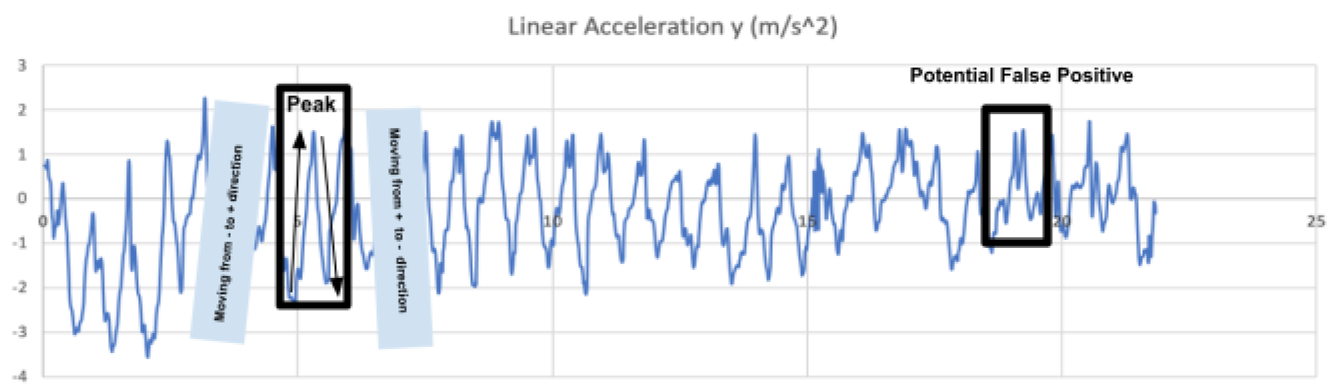
## Observe the signal

When we observed the signal, the y linear acceleration data from phyphox, we saw that the signal appeared almost periodic, representing potential steps that were being taken. There was a pattern of peaks and drops which we inferred to represent the steps that were taken. However, we also noticed that our signal produced a great amount of noise, especially in the drop after a peak, which makes sense as a phone accelerometer can not have perfect sensor measurements. We have included a snapshot of a graph of our y linear acceleration data to show the general trend in the walking motion that we have discussed so far.



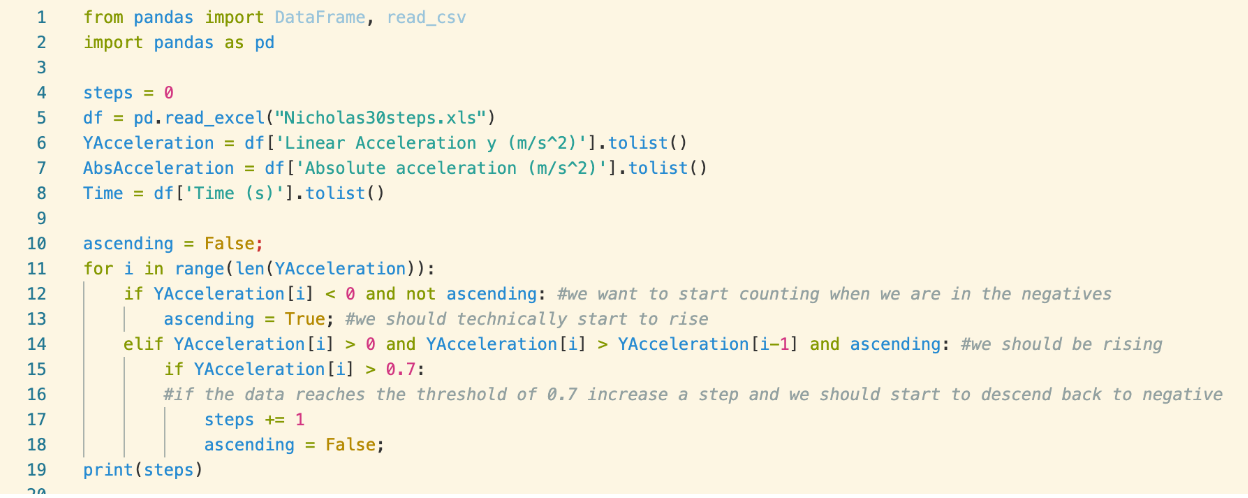
## Choose the feature

We decided that the features that we will use to detect steps are the peaks and valleys in the y linear acceleration data. We noticed that most of the “step” frames would start in the negative values (below zero / x-axis) and ascend up to the positive values (above zero / x-axis) reaching a max or peak value and start to descend back down to below zero. These step frames also appeared to take a similar amount of time if we were to compare their widths. We decided that we could use this ascending from negatives to positives and descending from positives to negatives direction to help us determine if a step was taken. Of course we will have to take into account a certain threshold that must be met in order to rule out false positives caused by the sensor noise. These false positives ascend, reach a peak, and start to descend but don’t actually go below 0 like a true step would.



## Implement the step counting algorithm

Our step count algorithm will run offline and will count steps when we input the y linear acceleration excel file to the algorithm. The language that we have decided to use is Python for convenience and the pandas library to help us with working with the excel data we received from the phyphox application. Our python step counting program requires that the user be in the directory as the data excel file in order to be accessed by pandas. Essentially, our step counting algorithm starts counting steps when it notices a negative value and then proceeds to set a variable called “ascending” to be True to signify that our data values should start to increase until a peak is reached. During this period where data is increasing, if during a particular discrete moment our y acceleration is greater than the discrete moment before and we happen to reach a threshold where the y acceleration is greater than 0.7, that is enough to believe that a step has been taken and we increase the step counter. We chose a threshold to help us not count false positives as a step. At the same time, the variable “ascending” is reset back to False since the data values should start to decrease back down to the negatives. This process is expected to repeat for the entirety of the data values, mirroring the pattern of peaks and valleys we noticed in our data graph. The total number of steps are then printed out at the end.



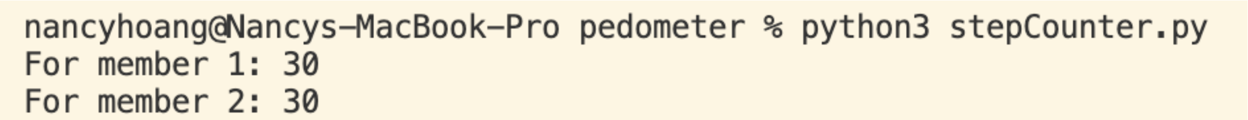
```python
from pandas import DataFrame, read_csv
import pandas as pd

steps = 0
df = pd.read_excel("Nicholas30steps.xls")
YAcceleration = df['Linear Acceleration y (m/s^2)'].tolist()
AbsAcceleration = df['Absolute acceleration (m/s^2)'].tolist()
Time = df['Time (s)'].tolist()

ascending = False;
for i in range(len(YAcceleration)):
    if YAcceleration[i] < 0 and not ascending: #we want to start counting when we are in the negatives
        ascending = True; #we should technically start to rise
    elif YAcceleration[i] > 0 and YAcceleration[i] > YAcceleration[i-1] and ascending: #we should be rising
        if YAcceleration[i] > 0.7:
        #if the data reaches the threshold of 0.7 increase a step and we should start to descend back to negative
            steps += 1
            ascending = False;
print(steps)
```

## Test if it correctly measures steps

The formula to determine Average accuracy is (member 1 accuracy + member 2 accuracy) / 2. Each of the members in our pair took 30 steps while the sensor recorded the walking data (Nancy30steps.xls, Nicholas30steps.xls files). The step counter program printed out 30 steps for both member 1 and member 2 which is exactly how many steps each person took. Our average accuracy is 100% (100+100/2).



```
nancyhoang@Nancys-MacBook-Pro pedometer % python3 stepCounter.py
For member 1: 30
For member 2: 30
```