### Introduction

In this final project, available to Verified learners only, we'll attempt to predict the type of physical activity (e.g., walking, climbing stairs) from tri-axial smartphone accelerometer data. Smartphone accelerometers are very precise, and different physical activities give rise to different patterns of acceleration.

*(Note on project availability: while project submission is only available to Verified learners, all learners are welcome to work on the project on their own and still have access to the instructions describing the project.)*

### Input Data

The input data used for training in this project consists of two files. The first file, `train_time_series.csv`, contains the raw accelerometer data, which has been collected using the Beiwe research platform, and it has the following format:

```
timestamp, UTC time, accuracy, x, y, z
```

You can use the **`timestamp`** column as your time variable; you'll also need the last three columns, here labeled **`x`**, **`y`**, and **`z`**, which correspond to measurements of linear acceleration along each of the three orthogonal axes.

The second file, `train_labels.csv`, contains the activity labels, and you'll be using these labels to train your model. Different activities have been numbered with integers. We use the following encoding: 1 = standing, 2 = walking, 3 = stairs down, 4 = stairs up. Because the accelerometers are sampled at high frequency, the labels in `train_labels.csv` are only provided for every 10th observation in `train_time_series.csv`.

### Activity Classification

Your goal is to classify different physical activities as accurately as possible. To test your code, you're also provided a file called `test_time_series.csv`, and at the end of the project you're asked to provide the activity labels predicted by your code for this test data set. Only the course staff have the corresponding true labels for the test data, and the accuracy of your code will be determined as the percentage of correct classifications. Note that in both cases, for training and testing, the input file consists of a single (3-dimensional) time series. To test the accuracy of your code, you'll be asked to upload your predictions as a CSV file. This file called `test_labels.csv` is provided to you, but it only contains the time stamps needed for prediction; you'll need to augment this file by adding the corresponding class predictions (1,2,3,4).

### Code Run Time

In addition to providing the predictions, you're also asked to time the running time of your code, starting at the moment when you load in the test data set and ending at the moment you're done computing your predictions. You'll be asked to enter this running time, and the goal is to see how fast your code runs compared to the code of others. Because computing speeds vary for several reasons, including hardware and implementation of the code, these numbers aren't directly comparable, and for this reason your grading will not be affected by them. However, it may still be interesting to you to see how long the code of other participants takes to solve the problem.

**Project Submission**

You're expected to implement your solution using a Jupyter notebook. Once you're done, you're asked to upload the notebook, which will be peer reviewed by other course participants. This review will impact your final course grade, so you should write your code as clearly code as possible, include comments, and use meaningful variable names.

You can approach this problem any way you'd like. It may be beneficial to search the web for possible solutions, or you may try to solve this problem from scratch. We recommend that you build your code in stages, encapsulating different parts (tasks) of the problem into functions. There are many ways to solve this problem. Good luck!

**Files for Download**

You will need to download these four files to complete your project:

- train_labels.csv
- train_time_series.csv
- test_labels.csv
- test_time_series.csv