In [ ]:

```python
import pandas as pd
import json
from statistics import mean, median, stdev, variance
```

In [ ]:

```python
#sort out valid users only
no = [6,11,21,26,28,29,37,38,40,48,55]
user_num=[]
for i in range(0,60):
  if i not in no:
    user_num.append(i)
print(len(user_num))
```

```
49
```

In [ ]:

```python
#Build user id strings
def get_user(number):
  if number<10:
    number = '0' + str(num)
  else:
    number = str(num)
  user = 'u' + number
  return user
```

In [ ]:

```python
dir = '/StudentLife/Data'
```

In [ ]:

```python
#Gather data of walking activity
mean_walk = dict()
median_walk = dict()
std_walk = dict()
var_walk = dict()
for num in user_num:
  user = get_user(num)
  filename = dir+'/EMA/response/Exercise/Exercise_' + user + '.json'
  ex_json = open(filename)
  data = json.load(ex_json)
  walk = []
  for i in range(0, len(data)):
    if 'walk' in data[i].keys():
      walk.append(int(data[i]['walk']))
    else:
      continue
  if len(walk)>1:
    mean_walk[user]=mean(walk)
    median_walk[user]=median(walk)
    std_walk[user]=stdev(walk)
    var_walk[user]=variance(walk)
  elif len(walk)==1:
    mean_walk[user]=mean(walk)
    median_walk[user]=median(walk)
    std_walk[user]=0
    var_walk[user]=0
  else:
    mean_walk[user]=0
    median_walk[user]=0
    std_walk[user]=0
    var_walk[user]=0
  ex_json.close
print(mean_walk)
```

```
{'u00': 1.7027027027027026, 'u01': 2.4285714285714284, 'u02': 2.4166666666666665, 'u03': 2.125, 'u04': 2.5833333333333335, 'u05': 0, 'u07': 2.3076923076923075, 'u08': 2.4193548387096775, 'u09': 0, 'u10': 2.1, 'u12': 1.8888888888888888, 'u13':
0, 'u14': 3.75, 'u15': 1.8, 'u16': 2.5789473684210527, 'u17': 2.6666666666666665, 'u18': 2.4, 'u19': 2.340909090909091, 'u20': 2.6666666666666665, 'u22': 1, 'u23': 1.3529411764705883, 'u24': 2.111111111111111, 'u25': 2.2, 'u27': 1.454545454545
4546, 'u30': 2.6666666666666665, 'u31': 2.875, 'u32': 2.04, 'u33': 2.3823529411764706, 'u34': 3.3333333333333335, 'u35': 3, 'u36': 2.4166666666666665, 'u39': 1, 'u41': 3, 'u42': 2.272727272727273, 'u43': 2.7941176470588234, 'u44': 2.3863636363
636362, 'u45': 1.8571428571428572, 'u46': 2.3076923076923075, 'u47': 2.111111111111111, 'u49': 1.36, 'u50': 2, 'u51': 1.8571428571428572, 'u52': 1.7, 'u53': 1.8095238095238095, 'u54': 3.5, 'u56': 2, 'u57': 2.96875, 'u58': 2.8666666666666667,
'u59': 3}
```

In [ ]:

```python
#Gather data of user's mood
mean_mood = dict()
median_mood = dict()
std_mood = dict()
var_mood = dict()
for num in user_num:
  user = get_user(num)
  filename = dir+'/EMA/response/Mood 2/Mood 2_' + user + '.json'
  ex_json = open(filename)
  data = json.load(ex_json)
  mood = []
  for i in range(0, len(data)):
    if 'how' in data[i].keys():
      mood.append(int(data[i]['how']))
    else:
      continue
  if len(mood)>1:
    mean_mood[user]=mean(mood)
    median_mood[user]=median(mood)
    std_mood[user]=stdev(mood)
    var_mood[user]=variance(mood)
  elif len(mood)==1:
    mean_mood[user]=mean(mood)
    median_mood[user]=median(mood)
    std_mood[user]=0
    var_mood[user]=0
  else:
    mean_mood[user]=0
    median_mood[user]=0
    std_mood[user]=0
    var_mood[user]=0
  ex_json.close
print(mean_mood)
```

```
{'u00': 1.4, 'u01': 1, 'u02': 1.75, 'u03': 2, 'u04': 1.7142857142857142, 'u05': 0, 'u07': 0, 'u08': 2.142857142857143, 'u09': 0, 'u10': 1.9090909090909092, 'u12': 2.3333333333333335, 'u13': 0, 'u14': 2, 'u15': 0, 'u16': 2.25, 'u17': 2.18181818
18181817, 'u18': 1.75, 'u19': 1.75, 'u20': 0, 'u22': 1.1666666666666667, 'u23': 1, 'u24': 2, 'u25': 0, 'u27': 2, 'u30': 0, 'u31': 0, 'u32': 1.6666666666666667, 'u33': 2.0666666666666667, 'u34': 1, 'u35': 1.8333333333333333, 'u36': 1.6875, 'u3
9': 0, 'u41': 2.5, 'u42': 1, 'u43': 1, 'u44': 1.7142857142857142, 'u45': 1.375, 'u46': 2.25, 'u47': 1.5714285714285714, 'u49': 1.5555555555555556, 'u50': 0, 'u51': 2.0833333333333335, 'u52': 1.3333333333333333, 'u53': 2, 'u54': 0, 'u56': 1.833
3333333333333, 'u57': 1.4736842105263157, 'u58': 1.8333333333333333, 'u59': 1.9489795918367347}
```

In [ ]:

```python
#Gather data of user's stress level
conversion = {'1': '5', '2': '4', '3': '1', '4': '2', '5': '3'}
mean_stress = dict()
median_stress = dict()
std_stress = dict()
var_stress = dict()
for num in user_num:
  user = get_user(num)
  filename = dir+'/EMA/response/Stress/Stress_' + user + '.json'
  ex_json = open(filename)
  data = json.load(ex_json)
  stress = []
  for i in range(0, len(data)):
    if 'level' in data[i].keys():
      stress.append(int(conversion[data[i]['level']]))
    else:
      continue
  if len(stress)>1:
    mean_stress[user]=mean(stress)
    median_stress[user]=median(stress)
    std_stress[user]=stdev(stress)
    var_stress[user]=variance(stress)
  elif len(stress)==1:
    mean_stress[user]=mean(stress)
    median_stress[user]=median(stress)
    std_stress[user]=0
    var_stress[user]=0
  else:
    mean_stress[user]=0
    median_stress[user]=0
    std_stress[user]=0
    var_stress[user]=0
  ex_json.close
print(mean_stress)
```

```
{'u00': 3.4864864864864864, 'u01': 3.8125, 'u02': 4, 'u03': 3.5517241379310347, 'u04': 3.5348837209302326, 'u05': 2.875, 'u07': 2.981132075471698, 'u08': 3.8444444444444446, 'u09': 3, 'u10': 3.641509433962264, 'u12': 3.3666666666666667, 'u13':
0, 'u14': 3.5135135135135136, 'u15': 4.3, 'u16': 3.2710280373831777, 'u17': 1.380952380952381, 'u18': 3.625, 'u19': 3.78021978021978, 'u20': 4.3, 'u22': 3.581818181818182, 'u23': 4.545454545454546, 'u24': 3.8461538461538463, 'u25': 3.15, 'u2
7': 3.76, 'u30': 3.2, 'u31': 3.5, 'u32': 3.5606060606060606, 'u33': 2.5172413793103448, 'u34': 4, 'u35': 3.6923076923076925, 'u36': 4.482142857142857, 'u39': 5, 'u41': 3.4545454545454546, 'u42': 3.0344827586206895, 'u43': 3.5714285714285716,
'u44': 3.896551724137931, 'u45': 3.4642857142857144, 'u46': 2.6363636363636362, 'u47': 4.076923076923077, 'u49': 2.901639344262295, 'u50': 4, 'u51': 4.4772727272727275, 'u52': 3.5853658536585367, 'u53': 4.342105263157895, 'u54': 3.85, 'u56':
3.6315789473684212, 'u57': 3.8656716417910446, 'u58': 4.3090909090909095, 'u59': 3.680297397769517}
```

In [ ]:

```python
#Gather data of user's sleep duration
mean_sleeptime = dict()
median_sleeptime = dict()
std_sleeptime = dict()
var_sleeptime = dict()
for num in user_num:
  user = get_user(num)
  filename = dir+'/EMA/response/Sleep/Sleep_' + user + '.json'
  ex_json = open(filename)
  data = json.load(ex_json)
  sleephour = []
  for i in range(0, len(data)):
    if 'hour' in data[i].keys():
      sleephour.append(int(data[i]['hour']))
    else:
      continue
  if len(sleephour)>1:
    mean_sleeptime[user]=mean(sleephour)
    median_sleeptime[user]=median(sleephour)
    std_sleeptime[user]=stdev(sleephour)
    var_sleeptime[user]=variance(sleephour)
  elif len(sleephour)==1:
    mean_sleeptime[user]=mean(sleephour)
    median_sleeptime[user]=median(sleephour)
    std_sleeptime[user]=0
    var_sleeptime[user]=0
  else:
    mean_sleeptime[user]=0
    median_sleeptime[user]=0
    std_sleeptime[user]=0
    var_sleeptime[user]=0
  ex_json.close
print(mean_sleeptime)
```

```
{'u00': 7.109090909090909, 'u01': 7.222222222222222, 'u02': 7.923076923076923, 'u03': 8.071428571428571, 'u04': 6.470588235294118, 'u05': 7.5, 'u07': 7.041666666666667, 'u08': 7, 'u09': 8, 'u10': 7.787234042553192, 'u12': 7.846153846153846, 'u
13': 8, 'u14': 6.966666666666667, 'u15': 7.315789473684211, 'u16': 7.907692307692308, 'u17': 4.9, 'u18': 5.909090909090909, 'u19': 7.836363636363636, 'u20': 6.416666666666667, 'u22': 8.3, 'u23': 7.739130434782608, 'u24': 7.695652173913044, 'u2
5': 7.4, 'u27': 5.125, 'u30': 7.117647058823529, 'u31': 5.454545454545454, 'u32': 8.148936170212766, 'u33': 6.853658536585366, 'u34': 6.857142857142857, 'u35': 6.96, 'u36': 7.5625, 'u39': 7.8, 'u41': 7.571428571428571, 'u42': 6.71428571428571
4, 'u43': 7.48, 'u44': 9.068965517241379, 'u45': 7.7272727272727275, 'u46': 6.260869565217392, 'u47': 6.555555555555555, 'u49': 6.804878048780488, 'u50': 6.833333333333333, 'u51': 7.0588235294117645, 'u52': 7.304347826086956, 'u53': 6.61290322
5806452, 'u54': 5.583333333333333, 'u56': 9.285714285714286, 'u57': 7.491803278688525, 'u58': 7.258620689655173, 'u59': 6.095890410958904}
```

In [ ]:

```python
#Gather data of user's sleep wellness
mean_sleeprate = dict()
median_sleeprate = dict()
std_sleeprate = dict()
var_sleeprate = dict()
for num in user_num:
  user = get_user(num)
  filename = dir+'/EMA/response/Sleep/Sleep_' + user + '.json'
  ex_json = open(filename)
  data = json.load(ex_json)
  sleeprate = []
  for i in range(0, len(data)):
    if 'rate' in data[i].keys():
      sleeprate.append(int(data[i]['rate']))
    else:
      continue
  if len(sleeprate)>1:
    mean_sleeprate[user]=mean(sleeprate)
    median_sleeprate[user]=median(sleeprate)
    std_sleeprate[user]=stdev(sleeprate)
    var_sleeprate[user]=variance(sleeprate)
  elif len(sleeprate)==1:
    mean_sleeprate[user]=mean(sleeprate)
    median_sleeprate[user]=median(sleeprate)
    std_sleeprate[user]=0
    var_sleeprate[user]=0
  else:
    mean_sleeprate[user]=0
    median_sleeprate[user]=0
    std_sleeprate[user]=0
    var_sleeprate[user]=0
  ex_json.close
print(mean_sleeprate)
```

```
{'u00': 1.981818181818182, 'u01': 1.5925925925925926, 'u02': 2, 'u03': 1.7142857142857142, 'u04': 2.0588235294117645, 'u05': 1.5, 'u07': 2, 'u08': 1.9024390243902438, 'u09': 1.6666666666666667, 'u10': 2.106382978723404, 'u12': 1.61538461538461
54, 'u13': 1, 'u14': 1.6666666666666667, 'u15': 1.894736842105263, 'u16': 1.9384615384615385, 'u17': 2.675, 'u18': 1.7575757575757576, 'u19': 1.6545454545454545, 'u20': 1.8333333333333333, 'u22': 1.55, 'u23': 1.608695652173913, 'u24': 2.304347
8260869565, 'u25': 2.0666666666666667, 'u27': 2.625, 'u30': 1.5294117647058822, 'u31': 2.5454545454545454, 'u32': 2.3404255319148937, 'u33': 2.292682926829268, 'u34': 1.2857142857142858, 'u35': 1.44, 'u36': 1.96875, 'u39': 1, 'u41': 1.928571428
5714286, 'u42': 1.5714285714285714, 'u43': 2.24, 'u44': 1.1379310344827587, 'u45': 1.0909090909090908, 'u46': 2.347826086956522, 'u47': 1.5555555555555556, 'u49': 2.024390243902439, 'u50': 2.1666666666666665, 'u51': 1.9705882352941178, 'u52':
1.2608695652173914, 'u53': 2.7096774193548385, 'u54': 2.25, 'u56': 2.0476190476190474, 'u57': 1.9508196721311475, 'u58': 1.5517241379310345, 'u59': 2.1232876712328768}
```

In [ ]:

```python
#Gather data of user's instantaneous mood
#user would choose one of 16 pictures shown that suit their current feeling, and the index of the chosen picture is recorded - photographic effect meter (PAM) method
mean_PAM = dict()
median_PAM = dict()
std_PAM = dict()
var_PAM = dict()
for num in user_num:
  user = get_user(num)
  filename = dir+'/EMA/response/PAM/PAM_' + user + '.json'
  ex_json = open(filename)
  data = json.load(ex_json)
  PAM = []
  for i in range(0, len(data)):
    if 'picture_idx' in data[i].keys():
      PAM.append(int(data[i]['picture_idx']))
    else:
      continue
  if len(PAM)>1:
    mean_PAM[user]=mean(PAM)
    median_PAM[user]=median(PAM)
    std_PAM[user]=stdev(PAM)
    var_PAM[user]=variance(PAM)
  elif len(PAM)==1:
    mean_PAM[user]=mean(PAM)
    median_PAM[user]=median(PAM)
    std_PAM[user]=0
    var_PAM[user]=0
  else:
    mean_PAM[user]=0
    median_PAM[user]=0
    std_PAM[user]=0
    var_PAM[user]=0
  ex_json.close
print(mean_PAM)
```

```
{'u00': 8.605128205128205, 'u01': 9.971428571428572, 'u02': 9, 'u03': 8, 'u04': 8.65137614678899, 'u05': 9.333333333333334, 'u07': 9.993197278911564, 'u08': 8.863636363636363, 'u09': 10, 'u10': 9.706070287539935, 'u12': 8.965174129353233, 'u1
3': 7.25, 'u14': 8.176767676767676, 'u15': 9.075471698113208, 'u16': 9.35933147632312, 'u17': 11.26046511627907, 'u18': 11.182389937106919, 'u19': 11.942708333333334, 'u20': 6.392156862745098, 'u22': 7.119047619047619, 'u23': 5.50314465408805,
'u24': 7.658682634730539, 'u25': 9.572916666666666, 'u27': 9.434065934065934, 'u30': 9.850340136054422, 'u31': 8.768292682926829, 'u32': 9.9, 'u33': 8.304794520547945, 'u34': 8.555555555555555, 'u35': 8.843902439024391, 'u36': 8.25, 'u39': 8.14
2857142857142, 'u41': 7.390243902439025, 'u42': 7.118881118881119, 'u43': 8.51528384279476, 'u44': 8.652173913043478, 'u45': 7.69811320754717, 'u46': 8.164102564102564, 'u47': 7.40566037735849, 'u49': 9.423868312757202, 'u50': 8.57894736842105
3, 'u51': 6.936241610738255, 'u52': 8.25, 'u53': 10.197247706422019, 'u54': 7.480769230769231, 'u56': 10.336842105263157, 'u57': 9.36339522546419, 'u58': 5.761467889908257, 'u59': 9.528604118993135}
```

In [ ]:

```python
#Gather data about user's personality traits, including enthusiasm, criticalness, dependability, anxiousness, experience-will, reservedness, sympathy,
#                                                          disorganization, calmness, and conventional
from statistics import mean, median, stdev, variance
mean_enthusiasm = dict()
mean_criticalness = dict()
mean_dependability = dict()
mean_anxiousness = dict()
mean_experienceWill = dict()
mean_reservedness = dict()
mean_sympathy = dict()
mean_disorganization = dict()
mean_calmness = dict()
mean_conventional = dict()
median_enthusiasm = dict()
std_enthusiasm = dict()
var_enthusiasm = dict()
median_criticalness = dict()
std_criticalness = dict()
var_criticalness = dict()
median_dependability = dict()
std_dependability = dict()
var_dependability = dict()
median_anxiousness = dict()
std_anxiousness = dict()
var_anxiousness = dict()
median_experienceWill = dict()
std_experienceWill = dict()
var_experienceWill = dict()
median_reservedness = dict()
std_reservedness = dict()
var_reservedness = dict()
median_sympathy = dict()
std_sympathy = dict()
var_sympathy = dict()
median_disorganization = dict()
std_disorganization = dict()
var_disorganization = dict()
median_calmness = dict()
std_calmness = dict()
var_calmness = dict()
median_conventional = dict()
std_conventional = dict()
var_conventional = dict()
keys_dict = {'enthusiastic': [mean_enthusiasm, median_enthusiasm, std_enthusiasm, var_enthusiasm],
             'critical': [mean_criticalness, median_criticalness, std_criticalness, var_criticalness],
             'dependable': [mean_dependability, median_dependability, std_dependability, var_dependability],
             'anxious': [mean_anxiousness, median_anxiousness, std_anxiousness, var_anxiousness],
             'experiences': [mean_experienceWill, median_experienceWill, std_experienceWill, var_experienceWill],
             'reserved': [mean_reservedness, median_reservedness, std_reservedness, var_reservedness],
             'sympathetic': [mean_sympathy, median_sympathy, std_sympathy, var_sympathy],
             'disorganized': [mean_disorganization, median_disorganization, std_disorganization, var_disorganization],
             'calm': [mean_calmness, median_calmness, std_calmness, var_calmness],
             'conventional': [mean_conventional, median_conventional, std_conventional, var_conventional]
             }
for num in user_num:
  user = get_user(num)
  filename = dir+'/EMA/response/Behavior/Behavior_' + user + '.json'
  ex_json = open(filename)
  data = json.load(ex_json)
  for key in keys_dict.keys():
    data_collector = []
    for i in range(0, len(data)):
      if key in data[i].keys() and data[i][key] != 'null':
        data_collector.append(int(data[i][key]))
      else:
        continue
    if len(data_collector)!=0:
      keys_dict[key][0][user]=mean(data_collector)
      keys_dict[key][1][user]=median(data_collector)
      if len(data_collector) > 1:
        keys_dict[key][2][user]=stdev(data_collector)
        keys_dict[key][3][user]=variance(data_collector)
      else:
        keys_dict[key][2][user]=0
        keys_dict[key][3][user]=0
    else:
      keys_dict[key][0][user]=0
      keys_dict[key][1][user]=0
      keys_dict[key][2][user]=0
      keys_dict[key][3][user]=0
  ex_json.close
```

In [ ]:

```python
#initialize a dataframe with indexes are users' id
df = pd.DataFrame({'A': mean_walk.keys()})
df.head(5)
```

Out[ ]:

	A
0	u00
1	u01
2	u02
3	u03
4	u04

In [ ]:

```python
#Construct the dataframe using gathered data
features_list = [
                 mean_walk, mean_mood, mean_stress, mean_sleeptime, mean_sleeprate,
                 mean_enthusiasm, mean_criticalness, mean_dependability, mean_anxiousness, mean_experienceWill,
                 mean_reservedness, mean_sympathy, mean_disorganization, mean_calmness,
                 mean_conventional, median_enthusiasm, std_enthusiasm, var_enthusiasm,
                 median_criticalness, std_criticalness, var_criticalness,
                 median_dependability, std_dependability, var_dependability,
                 median_anxiousness, std_anxiousness, var_anxiousness,
                 median_experienceWill, std_experienceWill, var_experienceWill,
                 median_reservedness, std_reservedness, var_reservedness,
                 median_sympathy, std_sympathy, var_sympathy,
                 median_disorganization, std_disorganization, var_disorganization,
                 median_calmness, std_calmness, var_calmness,
                 median_conventional, std_conventional, var_conventional, median_walk, std_walk, var_walk, median_mood, std_mood, var_mood, median_stress,
                 std_stress, var_stress, median_sleeptime, std_sleeptime, var_sleeptime, median_sleeprate, std_sleeprate, var_sleeprate,
                 mean_PAM, median_PAM, std_PAM, var_PAM
                 ]
features_name = ['mean_walk', 'mean_mood', 'mean_stress', 'mean_sleeptime', 'mean_sleeprate', 'mean_enthusiasm', 'mean_criticalness', 'mean_dependability', 'mean_anxiousness', 'mean_experienceWill', 'mean_reservedness', 'mean_sympathy', 'mean_
                 'median_criticalness', 'std_criticalness', 'var_criticalness',
                 'median_dependability', 'std_dependability', 'var_dependability',
                 'median_anxiousness', 'std_anxiousness', 'var_anxiousness',
                 'median_experienceWill', 'std_experienceWill', 'var_experienceWill',
                 'median_reservedness', 'std_reservedness', 'var_reservedness',
                 'median_sympathy', 'std_sympathy', 'var_sympathy',
                 'median_disorganization', 'std_disorganization', 'var_disorganization',
                 'median_calmness', 'std_calmness', 'var_calmness',
                 'median_conventional', 'std_conventional', 'var_conventional', 'median_walk',
                 'std_walk', 'var_walk', 'median_mood', 'std_mood', 'var_mood', 'median_stress', 'std_stress', 'var_stress', 'median_sleeptime',
                 'std_sleeptime', 'var_sleeptime', 'median_sleeprate', 'std_sleeprate', 'var_sleeprate',
                 'mean_PAM', 'median_PAM', 'std_PAM', 'var_PAM'
                 ]

for i in range(0, len(features_name)):
  df[features_name[i]] = df['A'].map(features_list[i])
df.head(5)
```

Out[ ]:

	A	mean_walk	mean_mood	mean_stress	mean_sleeptime	mean_sleeprate	mean_enthusiasm	mean_criticalness	mean_dependability	mean_anxiousness	...	median_sleeptime	std_sleeptime	var_sleeptime	median_sleeprate	std_sleeprate	var_sleeprate	mean_PAM	median_PAM	std_P
0	u00	1.702703	1.400000	3.486486	7.109091	1.981818	3.166667	2.615385	3.416667	3.769231	...	8.0	3.016453	9.098990	2.0	1.079968	1.166330	8.605128	7.0	3.687
1	u01	2.428571	1.000000	3.812500	7.222222	1.592593	3.000000	1.000000	3.000000	1.000000	...	7.0	1.219500	1.487179	2.0	0.572394	0.327635	9.971429	11.0	4.266
2	u02	2.416667	1.750000	4.000000	7.923077	2.000000	5.000000	2.500000	2.500000	1.500000	...	8.0	1.128648	1.273846	2.0	0.400000	0.160000	9.000000	10.0	4.099
3	u03	2.125000	2.000000	3.551724	8.071429	1.714286	3.333333	2.833333	4.166667	1.666667	...	7.5	2.141799	4.587302	1.5	0.809991	0.656085	8.000000	7.0	4.395
4	u04	2.583333	1.714286	3.534884	6.470588	2.058824	3.000000	3.500000	3.000000	3.500000	...	6.0	0.861123	0.741533	2.0	0.547234	0.299465	8.651376	10.0	4.114

5 rows × 65 columns

In [ ]:

```python
#Assign users their class by firstly read the data about users' response to a survey
output = pd.read_csv(dir+'/survey/PerceivedStressScale.csv')
drop = []
for i in range(0, 45):
  if output['uid'].iloc[i] not in list(df['A']):
    drop.append(i)
output = output.drop(drop, axis=0)
output.head(5)
```

Out[ ]:

	uid	type	1. In the last month, how often have you been upset because of something that happened unexpectedly?	2. In the last month, how often have you felt that you were unable to control the important things in your life?	3. In the last month, how often have you felt nervous and "stressed"?	4. In the last month, how often have you felt confident about your ability to handle your personal problems?	5. In the last month, how often have you felt that things were going your way?	6. In the last month, how often have you found that you could not cope with all the things that you had to do?	7. In the last month, how often have you been able to control irritations in your life?	8. In the last month, how often have you felt that you were on top of things?	9. In the last month, how often have you been angered because of things that were outside of your control?	10. In the last month, how often have you felt difficulties were piling up so high that you could not overcome them?
0	u00	pre	Sometime	Sometime	Fairly often	Fairly often	Sometime	Very often	Sometime	Sometime	Fairly often	Fairly often
1	u01	pre	Sometime	Sometime	Sometime	Sometime	Fairly often	Sometime	Fairly often	Fairly often	Almost never	Almost never
2	u02	pre	Fairly often	Sometime	Sometime	Fairly often	Almost never	Sometime	Almost never	Sometime	Sometime	Almost never
3	u03	pre	Sometime	Almost never	Sometime	Almost never	Sometime	Never	Almost never	Never	Never	Never
4	u04	pre	Almost never	Almost never	Fairly often	Sometime	Sometime	Fairly often	Sometime	Almost never	Sometime	Sometime

In [ ]:

```python
#Encode categorical data in a particular suitable way
condition = set()
for n in range(0, len(output.columns)-2):
  for m in range(0, output.shape[0]):
    condition.add(output[output.columns[n+2]].iloc[m])

con_conv = {'Almost never': 1, 'Fairly often': 3, 'Never': 0, 'Sometime': 2, 'Very often': 4}

for n in range(0, len(output.columns)-2):
  for m in range(0, output.shape[0]):
    output[output.columns[n+2]].iloc[m] = con_conv[output[output.columns[n+2]].iloc[m]]

for m in range(0, output.shape[0]):
  for r in [4,5,7,8]:
    output[output.columns[r]].iloc[m] = 4-output[output.columns[r]].iloc[m]
```

In [ ]:

```python
#Generate each users' class
sum_dict = dict()
for m in range(0, output.shape[0]):
  sum = 0
  for n in range(0, len(output.columns)-2):
    sum += output[output.columns[n+2]].iloc[m]
  sum_dict[output['uid'].iloc[m]] = sum

con_dict = dict()
for n in range(0, len(list(sum_dict.keys()))):
  con = sum_dict[list(sum_dict.keys())[n]]
  if con <= 20:
    con_dict[list(sum_dict.keys())[n]] = 0
  if con > 20:
    con_dict[list(sum_dict.keys())[n]] = 1
print(con_dict)
```

```
{'u00': 0, 'u01': 0, 'u02': 0, 'u03': 0, 'u04': 0, 'u05': 0, 'u07': 0, 'u08': 0, 'u09': 0, 'u10': 1, 'u12': 0, 'u13': 0, 'u14': 0, 'u15': 0, 'u16': 1, 'u17': 0, 'u18': 1, 'u19': 1, 'u20': 0, 'u22': 0, 'u23': 1, 'u24': 0, 'u27': 0, 'u30': 0, 'u
31': 1, 'u32': 0, 'u33': 0, 'u34': 0, 'u35': 0, 'u36': 0, 'u42': 0, 'u43': 0, 'u44': 0, 'u45': 1, 'u46': 1, 'u47': 0, 'u49': 0, 'u50': 1, 'u51': 0, 'u52': 0, 'u53': 0, 'u54': 0, 'u56': 0, 'u57': 0, 'u58': 0, 'u59': 0}
```

In [ ]:

```python
#Assign class to each users and finalize the dataframe
output['Total'] = output['uid'].map(con_dict)
df['con'] = df['A'].map(con_dict)
df = df.dropna()
print(df.shape)
df.head(5)
```

```
(46, 66)
```

Out[ ]:

	A	mean_walk	mean_mood	mean_stress	mean_sleeptime	mean_sleeprate	mean_enthusiasm	mean_criticalness	mean_dependability	mean_anxiousness	...	std_sleeptime	var_sleeptime	median_sleeprate	std_sleeprate	var_sleeprate	mean_PAM	median_PAM	std_PAM	var_PAM	co
0	u00	1.702703	1.400000	3.486486	7.109091	1.981818	3.166667	2.615385	3.416667	3.769231	...	3.016453	9.098990	2.0	1.079968	1.166330	8.605128	7.0	3.687396	13.596889	0.
1	u01	2.428571	1.000000	3.812500	7.222222	1.592593	3.000000	1.000000	3.000000	1.000000	...	1.219500	1.487179	2.0	0.572394	0.327635	9.971429	11.0	4.266218	18.200617	0.
2	u02	2.416667	1.750000	4.000000	7.923077	2.000000	5.000000	2.500000	2.500000	1.500000	...	1.128648	1.273846	2.0	0.400000	0.160000	9.000000	10.0	4.099443	16.805430	0.
3	u03	2.125000	2.000000	3.551724	8.071429	1.714286	3.333333	2.833333	4.166667	1.666667	...	2.141799	4.587302	1.5	0.809991	0.656085	8.000000	7.0	4.395577	19.321101	0.
4	u04	2.583333	1.714286	3.534884	6.470588	2.058824	3.000000	3.500000	3.000000	3.500000	...	0.861123	0.741533	2.0	0.547234	0.299465	8.651376	10.0	4.114437	16.928593	0.

5 rows × 66 columns

In [ ]: