**Health Tracker**

Name: Sahan Gowda G (sgowdag)

Github link: <https://github.com/sahaann/simplilearn_health_tracker.git>

**DESCRIPTION**

**Lyka, the sporting shoe company, approaches you to create a simple python app that helps in tracking workouts, energy (or calories) burnt, BMI, weekly rewards, monthly rewards, fastest week, slowest month, etc. Some of the features are shown below.**

**The application will accept the following initial inputs from any user:**

Name, Age, Sex, Weight, Height

**You should be able to use either the FPS or Metric system with regards to weight and height.**

**The app should accept the next set of inputs, which are:**

1. **Day**
2. **Number of steps**
3. **Time taken (hours: minutes: seconds)**

**You can provide a number of inputs based on the following: Assumptions: Day = 1 for Monday, Day = 2 for Tuesday, etc.**

**Sample Input and Output**

**A typical 1-week sample input of 7 days will have the following input:**

1, 7500, 1:02:05

2, 8500, 1:06:12

3, 3500, 0:52:25

4, 4250, 0:59:35

5, 7800, 1:01:55

6, 8500, 1:12:15

7, 9500, 1:15:25

**The sample output for this data should be:**

Weekly Average: 4.64 Km/hr (or 2.95 mi/hr)

**A typical 1-month sample input of 5 days will have the following input:**

1, 7500, 1:02:05

2, 8500, 1:06:12

3, 3500, 0:52:25

4, 4250, 0:59:35

5, 7800, 1:01:55

6, 8500, 1:12:15

7, 9500, 1:15:25

1, 7500, 1:02:05

2, 8500, 1:06:12

3, 3500, 0:52:25

4, 4250, 0:59:35

5, 7800, 1:01:55

6, 8500, 1:12:15

7, 9500, 1:15:25

1, 7500, 1:02:05

2, 8500, 1:06:12

3, 3500, 0:52:25

4, 4250, 0:59:35

5, 7800, 1:01:55

6, 8500, 1:12:15

7, 9500, 1:15:25

1, 7500, 1:02:05

2, 8500, 1:06:12

3, 3500, 0:52:25

4, 4250, 0:59:35

5, 7800, 1:01:55

6, 8500, 1:12:15

7, 9500, 1:15:25

**Any breaks in the schedule can be provided as follows. Again, a sample of 1-week data is given:**

1, 7500, 1:02:05

2, 0, 0:00:0

3, 3500, 0:52:25

4, 4250, 0:59:35

5, 0, 0:00:00

6, 8500, 1:12:15

7, 9500, 1:15:25

**Sample calculation for missing dates (days) and computation of speed and distance**

Weekly Average: 4.44 Km/hr (or 2.77 mi/hr)

No Awards this week, as there are breaks in the schedule

**This can be done across the monthly data as well, as shown previously. Naturally, there are some weeks of achievement in terms of speed and total distance.**

**For 1-month data, say, the output should include the number of weeks of achievement. Note that, if there are missing workout days, that week cannot be considered for the computation of the monthly award.**

Congrats! You have got a 3 7/7 award for this month, **if there are 3 consecutive weeks of workouts, with a break of one/two days of workout in one particular week.**

**Similarly, for 4-month data, the output should include the number of months of achievement. Again, note that if the days have missing workout days, they cannot be considered for monthly achievement.**

Congrats! You have got a 2 M/M award for this month, **if there are 3 consecutive months of workout, and there is a week in a month, where there is a one or two days break with no workouts.**

**For a complete input/output cycle for a week, typically, the app should provide the following:**

**Input - - - - - -**

Name: Ramana

Sex: Male

Age (years): 45

Weight (Kg): 70

Height (cms): 196

**Workout Input- - - - - - -**

1, 7500, 1:02:05

2, 8500, 1:06:12

3, 3500, 0:52:25

4, 4250, 0:59:35

5, 7800, 1:01:55

6, 8500, 1:12:15

7, 9500, 1:15:25

**Output- - - - - - -**

Hi Mr. Ramana

Your BMI is: 18.3. Try to put on some weight!!

Your Weekly achievement is as follows:

No breakout in Sessions: You get a 7/7 award

Your Fastest Speed is: 5.52 Km/hr

Your Longest Distance is: 6.7 km

Your Slowest Speed is: 3.06 Km/hr

Your Shortest Distance is: 2.5 Km

Your Weekly Average Speed is: 4.64 Km/hr

Your Weekly Average Distance is: 35.39 Km

**For an incomplete input/output cycle for a week, typically, the app should provide the following:**

**Input - - - - - -**

Name: Ramana

Sex: Male

Age (years): 45

Weight (Kg): 70

Height (cms): 196

**Workout Input- - - - - - -**

1, 7500, 1:02:05

2, 0, 0:00:0

3, 3500, 0:52:25

4, 4250, 0:59:35

5, 0, 0:00:00

6, 8500, 1:12:15

7, 9500, 1:15:25

**Output- - - - - - -**

Your BMI is: 18.3. Try to put on some weight!! Your Weekly achievement is as follows: Your Fastest Speed is: 5.52 Km/hr

Your Longest Distance is: 6.7 km

Your Slowest Speed is: 3.06 Km/hr

Your Shortest Distance is: 2.5 Km

Your Weekly Average Speed is: 4.44 Km/hr Your Weekly Average Distance is: 23.75 Km

**This is extendable across monthly data as well.**

**Tools used:**

* Python 3.11
* PyCharm CE

**Source Code:**

Health Tracker

import bmi\_calculator as bmi  
import workouts as workout  
  
def main():  
  
 name = bmi.get\_name()  
 gender = bmi.get\_gender()  
 bmi.get\_age()  
 your\_bmi = bmi.calculate\_bmi()  
  
 km\_distances**,** speed**,** weekly\_average\_km**,** weekly\_average\_mi**,** weekly\_average\_distance = workout.get\_workout\_input()  
  
 print("\nOutput -------------------------------------------------\n")  
  
 print(f"Hi {gender}. {name}")  
  
 if your\_bmi <= **18.5**:  
 print(f"Your BMI is: {your\_bmi}. You have some work ahead of you, but it's great that you are taking this first step. Try to put on some weight. You are considered UNDERWEIGHT.")  
 elif your\_bmi <= **24.9**:  
 print(f"Your BMI is: {your\_bmi}. That is a good weight, you are starting from a great place!. You are AVERAGE, which is considered NORMAL.")  
 elif your\_bmi <= **29.9**:  
 print(f"Your BMI is: {your\_bmi}. You have some work ahead of you, but it's great that you are taking this first step. Try to loose some weight. You are considered OVERWEIGHT.")  
 else:  
 print(f"Your BMI is: {your\_bmi}. There's a lot you could gain by losing a little weight. You are considered OBESE.")  
  
 print(f"Your Weekly achievement is as follows:")  
 for step in km\_distances:  
 if step == **0**:  
 print("No Awards this week, as there are breaks in the schedule")  
 else:  
 print("No breakout in Sessions: You get a 7/7 award")  
 break  
 print(f"Your Fastest Speed is: {round(max(speed)**, 2**)} Km/hr")  
 print(f"Your Longest Distance is: {max(km\_distances)} Km")  
 print(f"Your Slowest Speed is: {round(min(speed)**, 2**)} Km/hr")  
 print(f"Your Shortest Distance is: {min(km\_distances)} Km")  
 print(f"Your Weekly Average Speed is: {round(weekly\_average\_km**,2**)} Km//hr (or {round(weekly\_average\_mi**,2**)} mi/hr)")  
 print(f"Your Weekly Average Distance is: {round(weekly\_average\_distance**, 2**)} Km")  
  
if \_\_name\_\_ =="\_\_main\_\_":  
 main()

Workouts:

from datetime import \*  
import calendar  
  
  
def get\_workout\_input():  
 print("\nworkout input -------------------------------")  
 while True:  
 sample = input("\nDo you wish input a 7-days sample or 1-month sample? ")  
 if sample == "7-days" or sample == "7 days" or sample == "7" or sample == "7-days sample":  
 print("Please enter your sample of days worked out")  
   
 time\_durations = []  
 km\_distances = []  
  
 for i in range (**1,8**):  
 try:  
 day**,** steps**,** times = input().split(",")  
 except:  
 print("wrong input. Please try again")  
 day**,** steps**,** times = input().split(",")  
  
 # LIST OF TIME TAKEN (HOURS:MINUTES:SECONDS)   
 time\_durations.append(times)  
  
 #STEPS (CM -> KM)  
 km\_distances.append(float(steps)/**1000**)  
  
 # DAY  
 days= {  
 **1**: "Monday"**,  
 2**: "Tuesday"**,  
 3**: "Wednesday"**,  
 4**: "Thursday"**,  
 5**: "Friday"**,  
 6**: "Saturday"**,  
 7**: "Sunday"**,** }  
  
 day\_data = days.get(int(day)**,** "No such Day")  
  
 conversion\_factor = **0.621371** get\_hours = [float(timeduration.split(":")[**0**]) + float(timeduration.split(":")[**1**])/**60** + float(timeduration.split(":")[**2**])/**3600** for timeduration in time\_durations]  
 mi\_distances = [x\*conversion\_factor for x in km\_distances]  
  
 speed = [x/y for x**,**y in zip(km\_distances**,** get\_hours)]  
 mi\_per\_hour = [round(x/y**, 2**) for x**,**y in zip(mi\_distances**,** get\_hours)]  
 km\_per\_hour = [round(x/y**, 2**) for x**,**y in zip(km\_distances**,** get\_hours)]  
  
 weekly\_average\_km = sum(km\_per\_hour)/len(km\_per\_hour)  
 weekly\_average\_mi = sum(mi\_per\_hour)/len(mi\_per\_hour)  
 weekly\_average\_distance = sum(km\_distances)/len(km\_distances)  
  
 return km\_distances**,** speed**,** weekly\_average\_km**,** weekly\_average\_mi**,** weekly\_average\_distance  
  
 elif sample == "1-month" or sample == "1 month" or sample == "1" or sample == "1-month sample":  
 current\_day = datetime.now()  
 days\_in\_month = calendar.monthrange(current\_day.year**,** current\_day.month)  
 res = days\_in\_month[**1**]  
  
 print("Please enter your sample of days worked out")  
  
 time\_durations = []  
 km\_distances = []  
  
 for i in range(**1,** res+**1**):  
 try:  
 day**,** steps**,** time = input().split(",")  
 except:  
 print("wrong input. Please try again")  
 day**,** steps**,** time = input().split(",")  
  
 #STEPS (CM -> KM)  
 km\_distances.append(float(steps)/**1000**)  
  
 # LIST OF TIME TAKEN (HOURS:MINUTES:SECONDS)   
 time\_durations.append(times)  
  
 # DAY  
 days= {  
 **1**: "Monday"**,  
 2**: "Tuesday"**,  
 3**: "Wednesday"**,  
 4**: "Thursday"**,  
 5**: "Friday"**,  
 6**: "Saturday"**,  
 7**: "Sunday"**,** }  
  
 day\_data = days.get(int(day)**,** "No such Day")  
  
 conversion\_factor = **1.609344** get\_hours = [float(timeduration.split(":")[**0**]) + float(timeduration.split(":")[**1**])/**60** + float(timeduration.split(":")[**2**])/**3600** for timeduration in time\_durations]  
 mi\_distances = [x\*conversion\_factor for x in km\_distances]  
  
 speed = [x/y for x**,**y in zip(km\_distances**,** get\_hours)]  
 mi\_per\_hour = [round(x/y**, 2**) for x**,**y in zip(mi\_distances**,** get\_hours)]  
 km\_per\_hour = [round(x/y**, 2**) for x**,**y in zip(km\_distances**,** get\_hours)]  
  
 weekly\_average\_km = sum(km\_per\_hour)/len(km\_per\_hour)  
 weekly\_average\_mi = sum(mi\_per\_hour)/len(mi\_per\_hour)  
 weekly\_average\_distance = sum(km\_distances)/len(km\_distances)  
  
 return km\_distances**,** speed**,** weekly\_average\_km**,** weekly\_average\_mi**,** weekly\_average\_distance  
   
 else:  
 print("Invalid Sample")  
 continue

BMI Calculator

print("input -------------------------------------\n")  
  
  
def get\_name():  
 name = input("Name: ").capitalize()  
 return name  
  
  
def get\_gender():  
 sex = input("Sex: ").capitalize()  
 if sex == "Male":  
 sex = "Mr"  
 return sex  
 elif sex == "Female":  
 sex = "Ms"  
 return sex  
 else:  
 return sex  
  
  
  
def get\_age():  
 while True:  
 age = input("Age (years): ")  
 if age.isdigit():  
 return age  
 else:  
 age = input("Age (years): ")  
 continue  
  
  
def get\_weight():  
 while True:  
 weight = input("Weight (Kg): ")  
 if weight.isdigit():  
 return float(weight)  
 else:  
 continue  
  
  
def get\_height():  
 while True:  
 height = input("Height (cms): ")  
 if height.isdigit():  
 return float(height)  
 else:  
 continue  
  
  
def calculate\_bmi():  
 weight = get\_weight()  
 height = get\_height()  
  
 bmi = weight / (height/**100**)\*\***2** #divided height (cm) with 100 to get height(m)  
  
 return bmi

Screenshot:

