How to use Cycle Calculator for BMS Batteries?

1. Save calculate_remaining_cycles_BMS.py in a folder

2. Call it from command prompt using following command:
python calculate_remaining_cycles_BMS.py *"filepath" battery_id [install_datetime] [total_cycles] [cycle_limit] [charge_weight]*

Example:
python calculate_remaining_cycles_BMS.py "../data/BMS/BMS Confidential BC19.xlsx" COM5_03 "Jul 20 2017 10:33PM" 300 100 0.6

Beware of handling spaces within filepath and install_datetime in mac. Spaces might have to be escaped.
It might be like: python calculate_remaining_cycles_BMS.py ../data/BMS/BMS\ Confidential\ BC19.xlsx COM5_03 Jul\ 20\ 2017\ 10:30PM

3. Following are the parameters that can be used to control the program:

filepath:	Path to the excel file to be read. Mandatory Parameter.
battery_id:	Battery ID from the file. Mandatory Parameter.
install_datetime:	Battery Install DateTime. Default: First datetime of first cycle provided.
total_cycles:	Total number of cycles (charge, discharged combined). Default: 200
cycle_limit:	Battery's cycle limit in Amp-hours. Default: 50
charge_weight:	Weightage given to charge cycle. Default: 0.5. Discharge weight = 1 - charge weight

```
[vpai@manjaro src]$ python calculate_remaining_cycles_BMS.py  ../data/raw/BMS\ Confidential\ Data/NOSBC19.xlsx COM5_03 "Jul 20 2017 10:33PM" 200 50 0.5

Reading File: ../data/raw/BMS Confidential Data/NOSBC19.xlsx.

------------------------------------------------------------
Input Parameters:
------------------------------------------------------------
filepath = ../data/raw/BMS Confidential Data/NOSBC19.xlsx
battery_id = COM5_03
install_datetime = 2017-07-20 22:33:00
total_cycles = 200
cycle_limit = 50.0
charge_weight = 0.5
discharge_weight = 0.5

------------------------------------------------------------
Output:
------------------------------------------------------------
Cycles Completed as per data: 1.57
Days per cycle: 0.19
Total Completed cycles (estimated hindsight + as per data): 124.57
Predicted Remaining Cycles: 75.0
Forecast End Date: 2017-08-28 00:30:35.031847

[vpai@manjaro src]$
```

The above screenshot demos the usage of calculate_remaining_cycles.py program

Note:

1. Only the first two parameters, viz., filepath and battery_id are mandatory. Rest all the parameters are optional and have default values as specified above.
2. The install_datetime parameter should follow the exact format:
   format: "MMM DD YYYY hh:mma",
   example: "Jan 15 2018 12:00AM"
3. The order of the parameters is fixed
   filepath battery_id *[install_datetime] [total_cycles] [cycle_limit] [charge_weight]*

That means,

- if you want to use *total_cycles,* you have to compulsorily provide *install_datetime* before it.
- if you want to use *charge_weight,* you have to compulsorily provide *install_datetime, total_cycles* and *cycle_limit* in same order before it.

How Cycle_Calculator works?

1. Read Input:
   filepath, battery_id, install_datetime, total_cycles, cycle_limit, charge_weight
2. Read the excel file from provided *filepath*
3. Only use the data of the provided *batter_id,* ignore rest
4. For each cycle calculate *mean_current, min datetime and max datetime*
5. Remove all the cycles that are not of type 'Charging' or 'Discharging'
6. For each cycle calculate the time in hours the cycle was running (*max_datetime - min_datetime*)
7. For each cycle calculate following:
   *amp_hour = | current | * hours*
   ratio = amp_hour / cycle_limit

Preview of data in memory

	cycle_no	cycle_type	current	min_time	max_time	hours	AMP_HOUR	ratio
0	160	Charging	24.700000	2017-08-13 10:43:48	2017-08-13 10:43:48	0.0	0.000000	0.000000
1	160	Discharging	-15.470653	2017-08-13 10:41:28	2017-08-13 14:02:37	3.0	46.411960	0.928239
2	161	Charging	22.989489	2017-08-13 14:46:43	2017-08-13 17:49:23	3.0	68.968467	1.379369
3	161	Discharging	-20.903039	2017-08-13 14:02:53	2017-08-13 16:06:17	2.0	41.806077	0.836122

8. Calculate following:
   - for each charge row, *charge_weighted_sum* = SUM (*ratio*) * *charge_weight*
   - for each discharge row, dis*charge_weighted_sum* = SUM (*ratio*) * (1 - *charge_weight*)
   - *cycles_completed_as_per_data = charge_weighted_sum + discharge_weighted_sum*
   - *days_per_cycle = total_days_as_per_data / cycles_completed_as_per_data*
   - *total_days_from_install = first_datetime_as_per_data – install_datetime*
   - *total_estimated_cycles_completed* = ROUND *(total_days_from_install / days_per_cycle)* + *cycles_completed_as_per_data*
   - *remaining_cycles* = ROUND (*total_cycles – total_estimated_cycles_completed*)
   - *remaining_days = remaining_cycles * days_per_cycle*
   - *forecast_end_datetime = last_datetime_as_per_data + remaining_days*

9. **Output following:**
   - Cycles Completed as per data: is the calculated cycles utilized on the given data in the excels
   - Days per cycle: is the total number or the fraction of days for which your one complete cycle utilization lasts. One cycle utilization can be a fraction too if you are under-utilizing or over-utilizing your cycles. Thus the algorithm assumes that the days per cycle is affected by your average utilization pattern.
   - Total Completed Cycles (estimated hindsight + as per data) : is the estimated number of cycles you might have utilized in the past (considering the current pattern), plus the cycles actually utilized as per the given data
   - Predicted Remaining Cycles: is the resultant got after deducting the cycles utilized from total cycles given by the manufacturer
   - Forecast End Date: is the estimated end date of the battery life