VIA501 2021-Fall Homework Assignment # 1

Remarks:

Write the code yourself. ***Cheating is strictly forbidden***.

For each problem write your code in the function format and give the names of the functions as problem numbers, for example for the solution of problem1:

```
def problem1(input):
  return something
```

Put the codes for all problems into one file (jupter notebook file) and name that file using your student username in the following format: badays_midterm.ipynb. The notebook file should definitely contain the outputs of the functions, if applicable. Sample solution file (sample_solution.ipynb) is given to you to show how to organize your solutions.

Also write your name and student number inside the jupyter notebook as well.

Give as much as documentation for your script using comments.

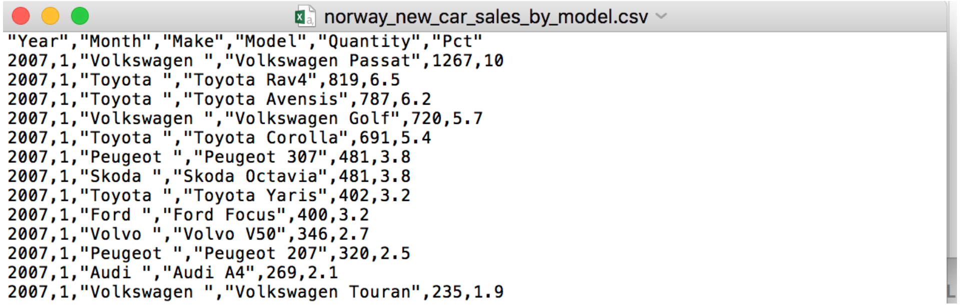
Note1: For the following problems, **do not** use libraries like Numpy or PANDAS. You are allowed to use only built-in python modules (os, sys, time, collections etc.)

Note2: If your homework solution file has problems in structure you can lose up to ***20*** points!!

For example, if you didn't write solutions in a function format or if you did not arrange input arguments properly you may lose points.

Problem 1 (25 Points).

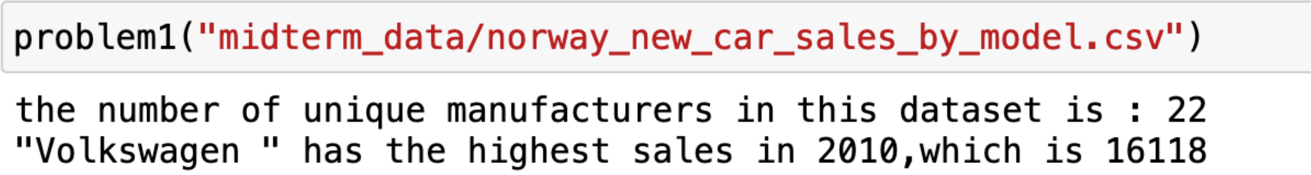
Norway_new_car_sales_by_model.csv file contains information of the new car sales in Norway between the years 2007-2017. The dataset was obtained from www.kaagle.com web site. The dataset comprises of monthly car sale quantity for various manufacturers and models. Make columns shows the manufacturer and Pct column shows the percent share in monlty total sales. Note that you should fix a typo for xa0Mercedes-Benz. Using this dataset do the following tasks:



```
"Year","Month","Make","Model","Quantity","Pct"
2007,1,"Volkswagen ","Volkswagen Passat",1267,10
2007,1,"Toyota ","Toyota Rav4",819,6.5
2007,1,"Toyota ","Toyota Avensis",787,6.2
2007,1,"Volkswagen ","Volkswagen Golf",720,5.7
2007,1,"Toyota ","Toyota Corolla",691,5.4
2007,1,"Peugeot ","Peugeot 307",481,3.8
2007,1,"Skoda ","Skoda Octavia",481,3.8
2007,1,"Toyota ","Toyota Yaris",402,3.2
2007,1,"Ford ","Ford Focus",400,3.2
2007,1,"Volvo ","Volvo V50",346,2.7
2007,1,"Peugeot ","Peugeot 207",320,2.5
2007,1,"Audi ","Audi A4",269,2.1
2007,1,"Volkswagen ","Volkswagen Touran",235,1.9
```

a) Print the number of unique manufacturers in this dataset.

b) Find the manufacturer that has the highest car sales in 2010?



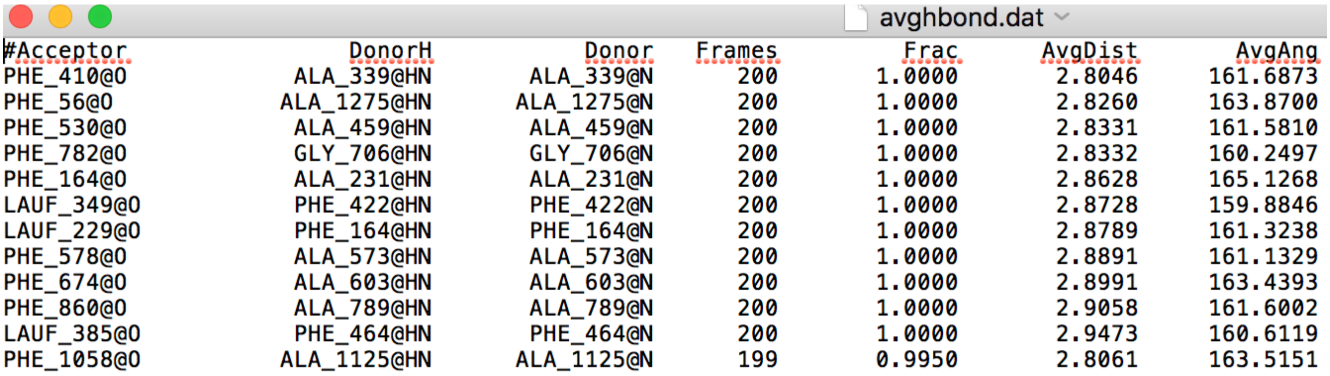
```
problem1("midterm_data/norway_new_car_sales_by_model.csv")
```

```
the number of unique manufacturers in this dataset is : 22
"Volkswagen " has the highest sales in 2010,which is 16118
```

Problem 2 (25 Points).

You can use Numpy or PANDAS just for this problem.

In this problem, you are going to deal with a data coming from a scientific research. You are supposed to process the data that was obtained by a scientific program. The data are the hydrogen bond (hbond) number obtained from molecular simulations. To better understand check out the "**avghbond.dat**" file.



avghbond.dat

```
#Acceptor                 DonorH           Donor   Frames        Frac     AvgDist      AvgAng
PHE_410@O             ALA_339@HN      ALA_339@N      200      1.0000      2.8046    161.6873
PHE_56@O             ALA_1275@HN     ALA_1275@N      200      1.0000      2.8260    163.8700
PHE_530@O             ALA_459@HN      ALA_459@N      200      1.0000      2.8331    161.5810
PHE_782@O             GLY_706@HN      GLY_706@N      200      1.0000      2.8332    160.2497
PHE_164@O             ALA_231@HN      ALA_231@N      200      1.0000      2.8628    165.1268
LAUF_349@O            PHE_422@HN      PHE_422@N      200      1.0000      2.8728    159.8846
LAUF_229@O            PHE_164@HN      PHE_164@N      200      1.0000      2.8789    161.3238
PHE_578@O             ALA_573@HN      ALA_573@N      200      1.0000      2.8891    161.1329
PHE_674@O             ALA_603@HN      ALA_603@N      200      1.0000      2.8991    163.4393
PHE_860@O             ALA_789@HN      ALA_789@N      200      1.0000      2.9058    161.6002
LAUF_385@O            PHE_464@HN      PHE_464@N      200      1.0000      2.9473    160.6119
PHE_1058@O           ALA_1125@HN     ALA_1125@N      199      0.9950      2.8061    163.5151
```

In fact, you are going to produce this "avghbond.dat" file in this problem. Hydrogen bonds form between Donor and Acceptor atoms through a hydrogen atom. You don't have to bother science behind the problem. In the first column you have Acceptor atom information which is given as the following format:

Aminoacit type _Aminoasit index @ atom type
For example In the second row this file

- "PHE" shows the aminoacid type,
- 410 shows the amino acid index
- "O" shows the atom type which oxygen.

DonorH and Donor columns have the same format.

Frames shows the number of time frames that this hbond take place. "Frac" column shows the fraction of the time that this hbond form: hbond number time frames normalized by the total time frame. AvgDist and AvgAng is the average distance and angle for this Hbond. You will ignore the last two columns in this problem.

The raw data that your going to process is in the "allhbond.dat" file. In the first raw you have the all hbond definitions. For example, the first raw looks like:

```
'#Frame   PHE_92@O-ALA_21@N-HN LYS_95@O-GLU_24@N-HN PHE_116@O-ALA_45@N-HN ALA_111@O-GLY_46@N-HN L
```

Here, data for each hbond is given as :
AcceptorAminoasitType_AcceptorAminoasitIndex@ AcceptorAtom -DonorAminoasitType_ DonorAminoasitIndex@DonorAtom-HydrogenAtomType

Rows starting from 2nd line towards the rest of the file has the following format:

```
'       1                    1                    1                     1                    1      '
```

In each column you have 1 or 0 stating that the hydrogen bond in that column forms or not.
Each row shows the hbond data for each time frame obtained from simulation. You have totall 201 lines meaning you have in total 200 simulation frame. Note that the first row is the header line.

In this problem you are going to write python class to process the data. The class name should be hbond_reader. The class should have the following methods(functions):

a) Write_avghbond

This method should calculate statistics for each hbond and write the “avghbond.dat” except for the last two column.

b) Write_hbondnumber

This method should calculate total number of hbonds in each time frame and writes “hbond_number_tseries.dat” file .

c) Check_intraaminoacid

This methods should check if any hbond forms between the atoms of the same amino acid index and prints these hbonds and number of frames that these hbonds forms. If the method does not find any of this type hbonds, it should print “no intra amino acid hbond found”.

To do this, basically you need to find hbonds that have the same DonorAminoasitIndex and AcceptorAminoasitIndex.

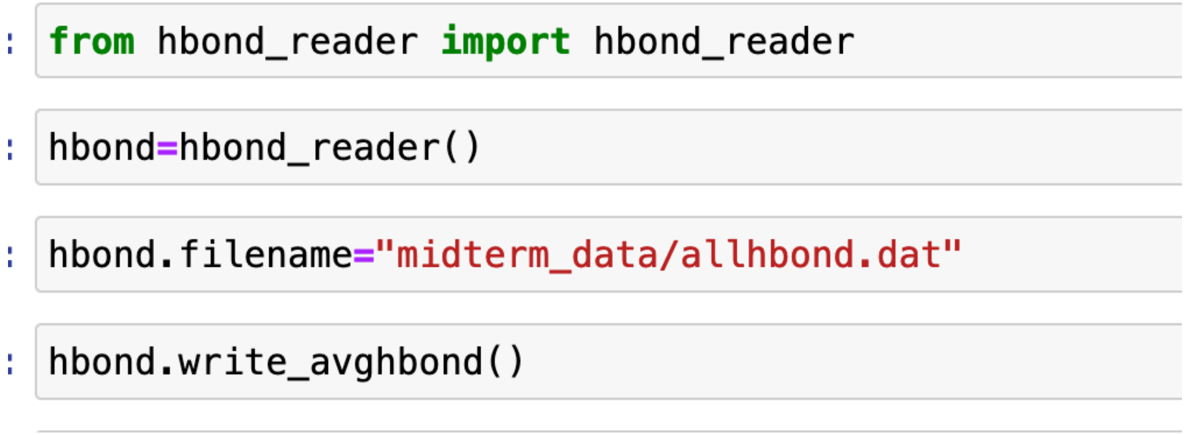
d) Find_highest_donortype

This method should find the amino acid type that gives the highest number of hbond donor.

The end user should only enter raw data file name (allhbond.dat) in the class as input. The class should have an attribute called as filename.

Do not put this solution into a jupyter notebook file, instead put into *.py file and name the file as **hbond_reader.py**

An example usage of your solution should be:



```
from hbond_reader import hbond_reader
```

```
hbond=hbond_reader()
```

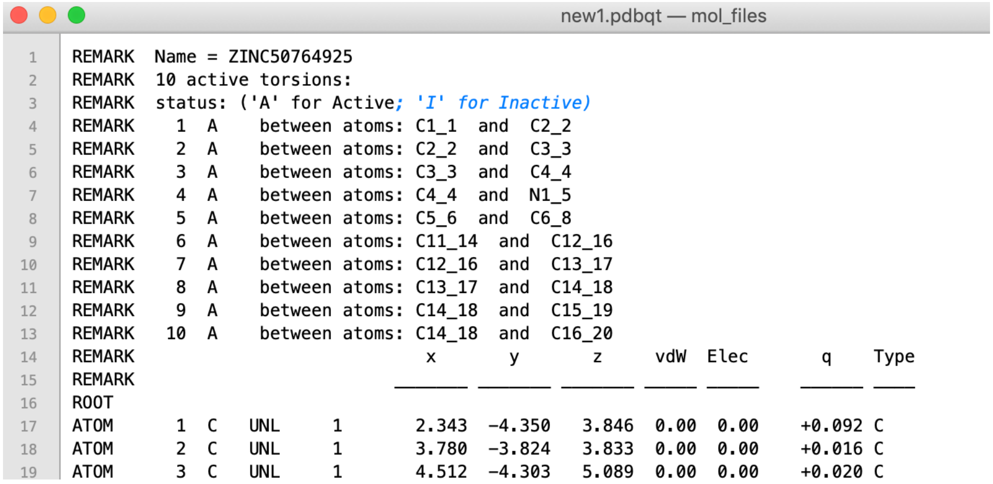
```
hbond.filename="midterm_data/allhbond.dat"
```

```
hbond.write_avghbond()
```

Problem 3 (25 Points)

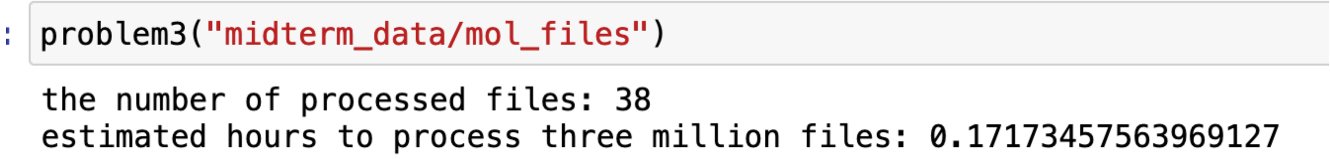
This example is taken from a real-life problem. Sometimes, you might need to make small changes to the output of a program. In one situation, I had to change filename of three million files. In this problem, you are going to do the same operation on only small number of files.

You are given a folder named “mol_files” which contain files having “.pdbqt” extension. These files were produced using a scientific software. The file looks like figure given below. In the first line database code of a molecule is given. For example, in the file new1.pdbqt it is “ZINC507664925”.
We want to change the file name from new1.pdbqt to “ZINC507664925.pdbqt”. Write a function that takes ***the folder name*** as the input and changes the names of the all pdbqt files in that folder. Print the number of files processed. Also, print how many hours would it take to process three million files.



```
REMARK  Name = ZINC50764925
REMARK  10 active torsions:
REMARK  status: ('A' for Active; 'I' for Inactive)
REMARK    1  A    between atoms: C1_1  and  C2_2
REMARK    2  A    between atoms: C2_2  and  C3_3
REMARK    3  A    between atoms: C3_3  and  C4_4
REMARK    4  A    between atoms: C4_4  and  N1_5
REMARK    5  A    between atoms: C5_6  and  C6_8
REMARK    6  A    between atoms: C11_14  and  C12_16
REMARK    7  A    between atoms: C12_16  and  C13_17
REMARK    8  A    between atoms: C13_17  and  C14_18
REMARK    9  A    between atoms: C14_18  and  C15_19
REMARK   10  A    between atoms: C14_18  and  C16_20
REMARK                         x       y       z     vdW  Elec       q    Type
REMARK                      _______ _______ _______ _____ _____    ______ ____
ROOT
ATOM      1  C   UNL     1       2.343  -4.350   3.846  0.00  0.00    +0.092 C
ATOM      2  C   UNL     1       3.780  -3.824   3.833  0.00  0.00    +0.016 C
ATOM      3  C   UNL     1       4.512  -4.303   5.089  0.00  0.00    +0.020 C
```

The output of your function should be like the following figure.



```
problem3("midterm_data/mol_files")
```

```
the number of processed files: 38
estimated hours to process three million files: 0.17173457563969127
```

Problem 4 (25 Points).

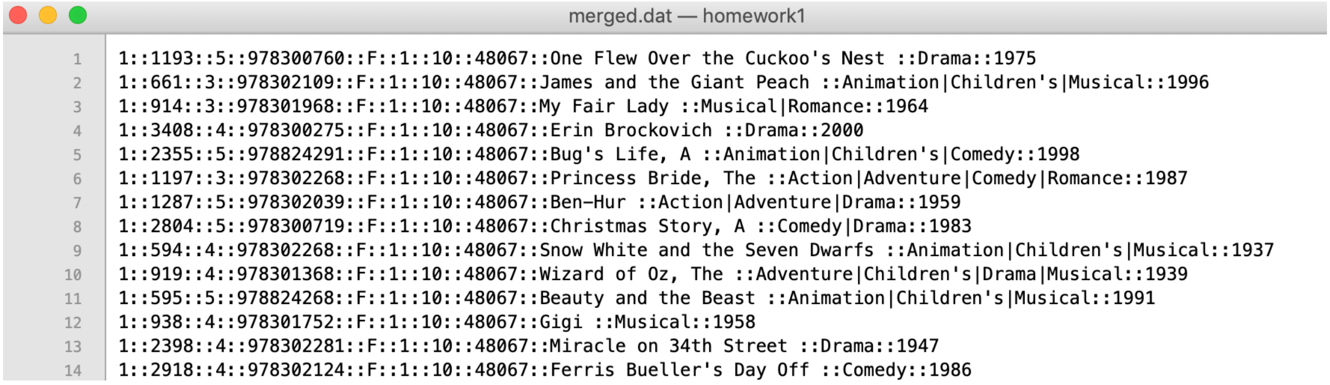
Remember the movie rating example presented in week1. We merged the contents of movies.dat, ratings.dat and users.dat files into one dataframe table using pandas library. Here, in this problem you are asked do the same operation without using pandas library. To get the movie rating data go to this link: https://grouplens.org/datasets/movielens/ and download the ml-1m.zip file.

Write a function that takes three files as input and writes a text file (named as “merged.dat”) for merged data. Merged data should have the following order.

user_id movie_id rating timestamp gender age occupation zip title genres Year

Note that we added year information as an extra column!
The input arguments for the functions should be exactly in the following order:

Merged.dat file should be like this:

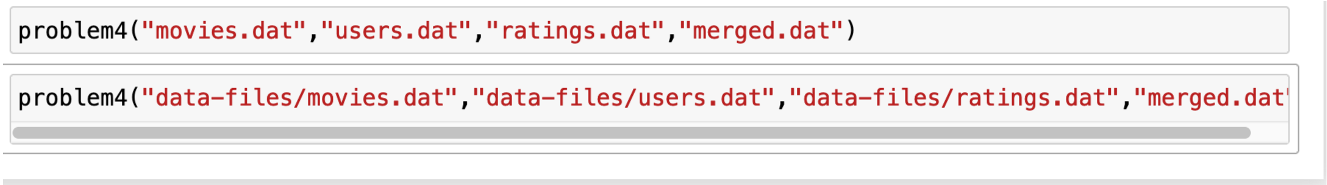


```
1::1193::5::978300760::F::1::10::48067::One Flew Over the Cuckoo's Nest ::Drama::1975
1::661::3::978302109::F::1::10::48067::James and the Giant Peach ::Animation|Children's|Musical::1996
1::914::3::978301968::F::1::10::48067::My Fair Lady ::Musical|Romance::1964
1::3408::4::978300275::F::1::10::48067::Erin Brockovich ::Drama::2000
1::2355::5::978824291::F::1::10::48067::Bug's Life, A ::Animation|Children's|Comedy::1998
1::1197::3::978302268::F::1::10::48067::Princess Bride, The ::Action|Adventure|Comedy|Romance::1987
1::1287::5::978302039::F::1::10::48067::Ben-Hur ::Action|Adventure|Drama::1959
1::2804::5::978300719::F::1::10::48067::Christmas Story, A ::Comedy|Drama::1983
1::594::4::978302268::F::1::10::48067::Snow White and the Seven Dwarfs ::Animation|Children's|Musical::1937
1::919::4::978301368::F::1::10::48067::Wizard of Oz, The ::Adventure|Children's|Drama|Musical::1939
1::595::5::978824268::F::1::10::48067::Beauty and the Beast ::Animation|Children's|Musical::1991
1::938::4::978301752::F::1::10::48067::Gigi ::Musical::1958
1::2398::4::978302281::F::1::10::48067::Miracle on 34th Street ::Drama::1947
1::2918::4::978302124::F::1::10::48067::Ferris Bueller's Day Off ::Comedy::1986
```

One final warning:

If a function like problem4 requires a file or folder write your function work appropritately independently from the path provided by the user.

For example, if the required files for problem4 are in a folder named “data-files” in my computer I should be able to run the function for problem4 like this:



```
problem4("movies.dat","users.dat","ratings.dat","merged.dat")
```

```
problem4("data-files/movies.dat","data-files/users.dat","data-files/ratings.dat","merged.dat
```

Also, input file names in the problem4 function should be in the same order as given above.

Note that the solution for this problem should not take more than **one minute**!