Assignment 4

Problem 1.1

I read the data set, replace and strip unwanted characters and converted all the values to float. Finally sum the values to find total amount spent

```
#Reading the csv file
file_csv=pd.read_csv("/Users/metuhead/Desktop/FE 520-Python/HW4/Homework4_Dataset/
res_purchase_2014.csv")

file_csv
#Showing Amount column in csv file

file_csv["Amount"]
# Replacing and splitting unwanted charachters in the column

df_Amount=file_csv["Amount"].map(lambda x: str(x).replace("(","-").replace("$","").strip(")")
.strip("zero").strip(" "))
#df_Amount=file_csv["Amount"].map(lambda x: str(x).strip("$"))

df_Amount=pd.to_numeric(df_Amount)
#df_Amount=df_Amount.astype(float)
# Calculating sum of Amount column

df_Amount.sum()
```

```
188040606.2299999
```

Problem 1.2

Filtered the Amount column by WW Grainger vendor and summed it up to find total amount spent Vendor WW GRAINGER

```
#1.2
# displaying Amount and Vendor columns
file_csv[["Amount","Vendor"]]


# Filtering the Amount column by WW GRAINGER vendor
df_GRAINGER=df_Amount[file_csv["Vendor"] =="WW GRAINGER"]

#sum of the Amount spent in WW GRAINGER
df_GRAINGER.sum()
```

```
5089417.48
```

Problem 1.3

Filtered the Amount spent by WW Supercenter, and summed it up

```
#1.3
# Filtering the Amount column by WW SUPERCENTER vendor
df_SUPERCENTER=df_Amount[file_csv["Vendor"]=="WM SUPERCENTER"]

#sum of the Amount spent in WW SUPERCENTER
df_SUPERCENTER.sum()
```

```
31777.83
```

Problem 1.4

Filtered the Amount spent by MCCC and chose only amounts spent in Grocery stores and supermarkets and summed it up.

```
#1.4
# displaying Amount and MCC columns
file_csv[["Amount","Merchant Category Code (MCC)"]]
# Filtering the Amount column by  MCC column where MCC is Grocery Stores
df_GROCERY=df_Amount[file_csv["Merchant Category Code (MCC)"]== "GROCERY STORES,AND
SUPERMARKETS"]
df_GROCERY

#sum of the Amount spent in Grocery Stores
df_GROCERY.sum()
```

```
1271339.9799999997
```

Problem 2.1

Imported and read the data sets

```
#Q1
df_Balancesheet= pd.read_excel("/Users/metuhead/Desktop/FE 520-Python/HW4/Homework4_Dataset/
Energy.xlsx")

df_Ratings=  pd.read_excel("/Users/metuhead/Desktop/FE 520-Python/HW4/Homework4_Dataset/
EnergyRating.xlsx")

#displaying Energy dataframe
df_Balancesheet
```

	Global Company Key	Data Date	Fiscal Year	Fiscal Quarter	Fiscal Year-end Month	Industry Format	Level of Consolidation - Company Interim Descriptor	Population Source	Data Format	Ticker Symbol	.
0	1380	20100331	2010	1	12	INDL	C	D	STD	HES	.
1	1380	20100630	2010	2	12	INDL	C	D	STD	HES	.
2	1380	20100930	2010	3	12	INDL	C	D	STD	HES	.
3	1380	20101231	2010	4	12	INDL	C	D	STD	HES	.
4	1380	20110331	2011	1	12	INDL	C	D	STD	HES	.

Problem 2.2

Filled all na values with 0 in both Balance sheet and Ratings dataframe.Then replace all 0 values with na values. Calculated 90%treshhold

```
#Q2
#filling all na with zeros for Energy
df_Balancesheet=df_Balancesheet.fillna(0)

df_Balancesheet
# Replacing all nan  with 0

df_Balancesheet=df_Balancesheet.replace(0,np.nan)
df_Balancesheet

df_Ratings
#filling all na with zeros for Ratings

df_Ratings=df_Ratings.fillna(0)

df_Ratings
# Replacing all nan values with 0

df_Ratings= df_Ratings.replace(0,np.nan)

df_Ratings
#Getting number of rows and col in Balance sheet dataframe

df_Balancesheet.shape
# Calculating 90% of the column numbers as threshold

tresh= 844*0.9

tresh
```

Dropped columns which includes more than 90% missing values or zero

```
df_Balancesheet=df_Balancesheet.dropna(axis=1,thresh=759.6)

df_Balancesheet
#Getting number of rows and columns in Rating dataframe

df_Ratings.shape
# # Calculating 90% of the column numbers as threshold

tresh1=2522*0.9

tresh1
#print the results

print(df_Ratings.shape, tresh1)
#Dropping columns including more than 90% zero or missing value

df_Ratings=df_Ratings.dropna(axis=1,thresh=2269.8)
df_Ratings
```

Problem 2.3

Replace all nan values with the mean of the columns for both data set

```
#Q3
##Replacing NaN values with mean
df_Balancesheet=df_Balancesheet.replace("NaN",df_Balancesheet.mean())
# or this one df_Balancesheet=df_Balancesheet.fillna(df_Balancesheet.mean())

df_Ratings=df_Ratings.replace("NaN",df_Ratings.mean())
# or this one  df_Ratings=df_Ratings.fillna(df_Ratings.mean())
#Selecting numerical parts

num_df_Balancesheet=df_Balancesheet.select_dtypes([np.number])
num_df_Ratings=df_Ratings.select_dtypes([np.number])
num_df_Ratings
num_df_Balancesheet
```

Problem 2.4

Defined a function to normalize the table and applied the function in both data frames

```
#Q4
## defining a function to normalize the table
def norm(x):

  x_new= (x-x.min())/(x.max()-x.min())

  return x_new


#Normalizing the table for both Balancesheet and Ratings

num_df_Balancesheet=num_df_Balancesheet.apply(norm)
#num_df_Balancesheet
num_df_Ratings=num_df_Ratings.apply(norm)
num_df_Ratings
```

Problem 2.5

Created two separate data frames for Current Assets-Other Total, Current Assets-Total and stacked them.
Created list which includes length, mean, std, min, and max of the data frame.
Converted lists into Series and concatenated them up

```
#Q5
# Creating separate dataframe includes:
##['Current Assets - Other - Total', 'Current Assets - Total']
des_cur=num_df_Balancesheet[["Current Assets - Other - Total"]]
des_cur_tot = num_df_Balancesheet[["Current Assets - Total"]]

#Stacking the dataframes
des_stack_cur_tot=des_cur_tot.stack()
des_stack=des_cur.stack()

#Creating lists which inludes length, mean, stdev, min of the variable
s1=[len(des_stack),mean(des_stack),stdev(des_stack),min(des_stack),max(des_stack)]
s2=[len(des_stack_cur_tot),mean(des_stack_cur_tot),stdev(des_stack_cur_tot),min
(des_stack_cur_tot),max(des_stack_cur_tot)]

#Converting lists into Series
S1=pd.Series(s1)
S2=pd.Series(s2)

#Concate series and creating a dataframe
pd.concat([S1,S2],axis=1)
```

	0	1
0	830.000000	830.000000
1	0.109210	0.126170
2	0.168063	0.179994
3	0.000000	0.000000
4	1.000000	1.000000

Problem 2.6
Created correlation matrix for specified variables

```
##Q6
#Reading and sorting the new balance sheet
df_NewBalancesheet=pd.read_excel("/Users/metuhead/Desktop/FE 520-Python/HW4/Homework4_Dataset/
Energy.xlsx")

df_NewBalancesheet=df_NewBalancesheet[["Current Assets - Other - Total","Current Assets - Total","Other
Long-term Assets","Assets Netting & Other Adjustments"]]
#Creating correlation matrix for new balance sheet

df_NewBalancesheet.corr()
```

	Current Assets - Other - Total	Current Assets - Total	Other Long-term Assets	Assets Netting & Other Adjustments
Current Assets - Other - Total	1.000000	0.790047	0.637424	0.047226
Current Assets - Total	0.790047	1.000000	0.677142	-0.081643
Other Long-term Assets	0.637424	0.677142	1.000000	-0.030717
Assets Netting & Other Adjustments	0.047226	-0.081643	-0.030717	1.000000

Problem 2.7

Split the company name and got the last word

```
### Q7
##Split the Company name column and get the last word
df_Balancesheet["Name"]=df_Balancesheet["Company Name"].str.split().str
[-1]

df_Balancesheet
```

ıg ıl :e :)	Interest and Related Expense- Total	Operating Expense- Total	Stock Exchange Code	CIK Number	Active/Inactive Status Marker	Current ISO Country Code - Incorporation	Name
0	85.0	7628.0	11	4447	A	USA	CORP
0	84.0	6412.0	11	4447	A	USA	CORP
0	95.0	6452.0	11	4447	A	USA	CORP
0	102.0	7600.0	11	4447	A	USA	CORP
0	101.0	8373.0	11	4447	A	USA	CORP
.	...	...	...	...	...	...	...
0	103.0	12965.0	11	1510295	A	USA	CORP

Problem 2.8

Merged two data sets by Data Date and Global Company Key

```
### Q8
# Merging two datasets based on 'datadate' and 'Global CompanyKey'
Matched=pd.merge(df_Ratings,df_Balancesheet,how="inner",on=["Data Date","Global Company Key"])
#display new dataframe

Matched
```

	Global Company Key	S&P Domestic Long Term Issuer Credit Rating	Data Date	Address Line 1	Ticker Symbol_x	Fiscal Year	Fiscal Quarter	Fiscal Year-end Month	Industry Format	Level of Consolidation - Company Interim Descriptor	...	]]
0	1380	BBB-	20100331	1185 Avenue of the Americas, 40th Floor	HES	2010	1	12	INDL	C	...	
1	1380	BBB-	20100630	1185 Avenue of the Americas, 40th Floor	HES	2010	2	12	INDL	C	...	

Problem 2.9

Created a new variable called Rating which includes a letter grade for corresponding values

```
# Q9
# Creating a key and values for ratings and corresponding values
credit={"AAA": 0,"AA+": 1,"AA":2,"AA-":3,"A+":4,"A":5,"A-":6,"BBB+":7,"BBB":8,"BBB-":9,"BB
+":10,"BB":11,"others":12,"":12}
# Mapping through all ratings and assigning numbers

Matched["Rate"]=Matched["S&P Domestic Long Term Issuer Credit Rating"].map(credit)
# display Matched dataframe

Matched
```

	Global Company Key	S&P Domestic Long Term Issuer Credit Rating	Data Date	Address Line 1	Ticker Symbol_x	Fiscal Year	Fiscal Quarter	Fiscal Year-end Month	Industry Format	Level of Consolidation - Company Interim Descriptor
0	1380	BBB-	20100331	1185 Avenue of the Americas, 40th Floor	HES	2010	1	12	INDL	C
1	1380	BBB-	20100630	1185 Avenue of the Americas, 40th	HES	2010	2	12	INDL	C

Problem 2.10

Created a histogram to calculate the frequency of rating for company whose name ends with CO

```
#Q10
#Filtering Matched dataframe with Name equals"CO"
df_frequency=Matched[Matched["Name"]=="CO"]
# Creating histogram for the distribution of the data

df_frequency["Rate"].hist()
Matched.to_csv('deneme.csv')
```