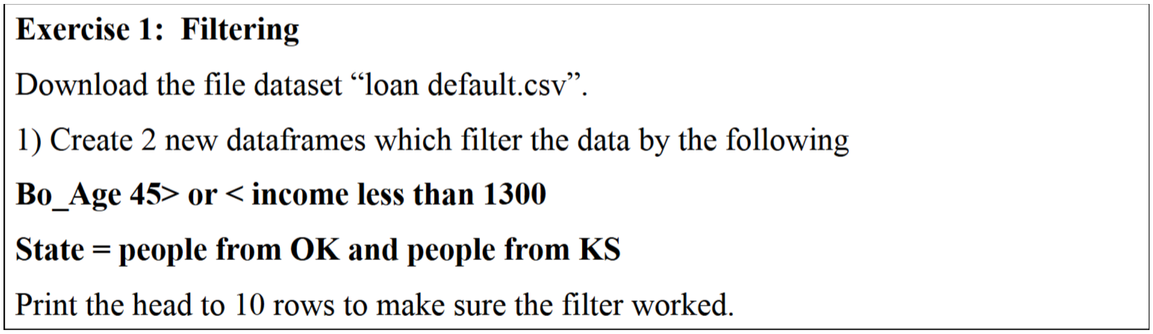
Class 7 - July 29th Notes

EX - 1

Slide 12



Exercise 1: Filtering

Download the file dataset “loan default.csv”.

1) Create 2 new dataframes which filter the data by the following

Bo_Age 45> or < income less than 1300

State = people from OK and people from KS

Print the head to 10 rows to make sure the filter worked.

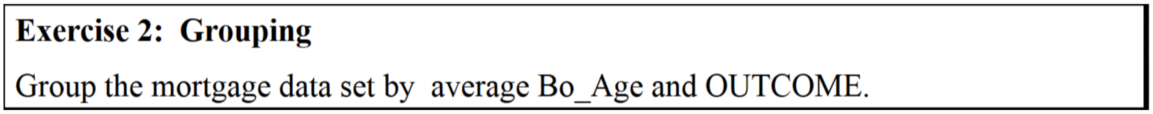
In [1]:
```
#Ex 1
import pandas as pd
loan = pd.read_csv("loan default.csv")
# display(loan)

## borrowers over 45 or income less than 1300:
borrowers_over_45_and_income_less_1300 = loan[(loan['Bo_Age']>45) | (loan['Tot_mthl
display(borrowers_over_45_and_income_less_1300)

# ##borrowers only from OK and KS:
borrowers_only_OK_KS = loan[(loan['State']=='OK') | (loan['State']=='KS')]
display(borrowers_only_OK_KS)
```

	Cust_ID	Bo_Age	Ln_Orig	Orig_LTV_Ratio_Pct	Credit_score	First_home	Tot_mthly_d
6	1006	53	136000.0	100	661	N	
9	1009	49	161050.0	97	661	N	
25	1025	49	89000.0	100	668	N	
31	1031	47	88000.0	100	694	N	
33	1033	47	186210.0	90	749	N	
...	...	...	...	...	...	...	
15136	16136	54	184000.0	70	628	N	
15137	16137	46	138600.0	100	649	N	
15141	16141	53	298950.0	95	684	N	
15144	16144	55	71000.0	100	790	N	
15150	16150	49	259900.0	100	641	Y	

2565 rows × 16 columns

	Cust_ID	Bo_Age	Ln_Orig	Orig_LTV_Ratio_Pct	Credit_score	First_home	Tot_mthly_d
48	1048	33	127000.0	100	678	N	
100	1100	31	166950.0	95	729	N	
103	1103	36	92150.0	97	620	N	
122	1122	36	170905.0	95	664	N	
202	1202	30	232000.0	80	641	N	
...	...	...	...	...	...	...	
12276	13276	31	86500.0	100	660	N	
12303	13303	26	92365.0	100	750	N	
13054	14054	27	96400.0	100	753	N	
13934	14934	38	80000.0	80	783	N	
14244	15244	24	113393.0	97	675	N	

118 rows × 16 columns



Exercise 2: Grouping

Group the mortgage data set by average Bo_Age and OUTCOME.

In [2]:
```
df1 = loan.groupby("OUTCOME")['Bo_Age'].mean()
display(df1)
```

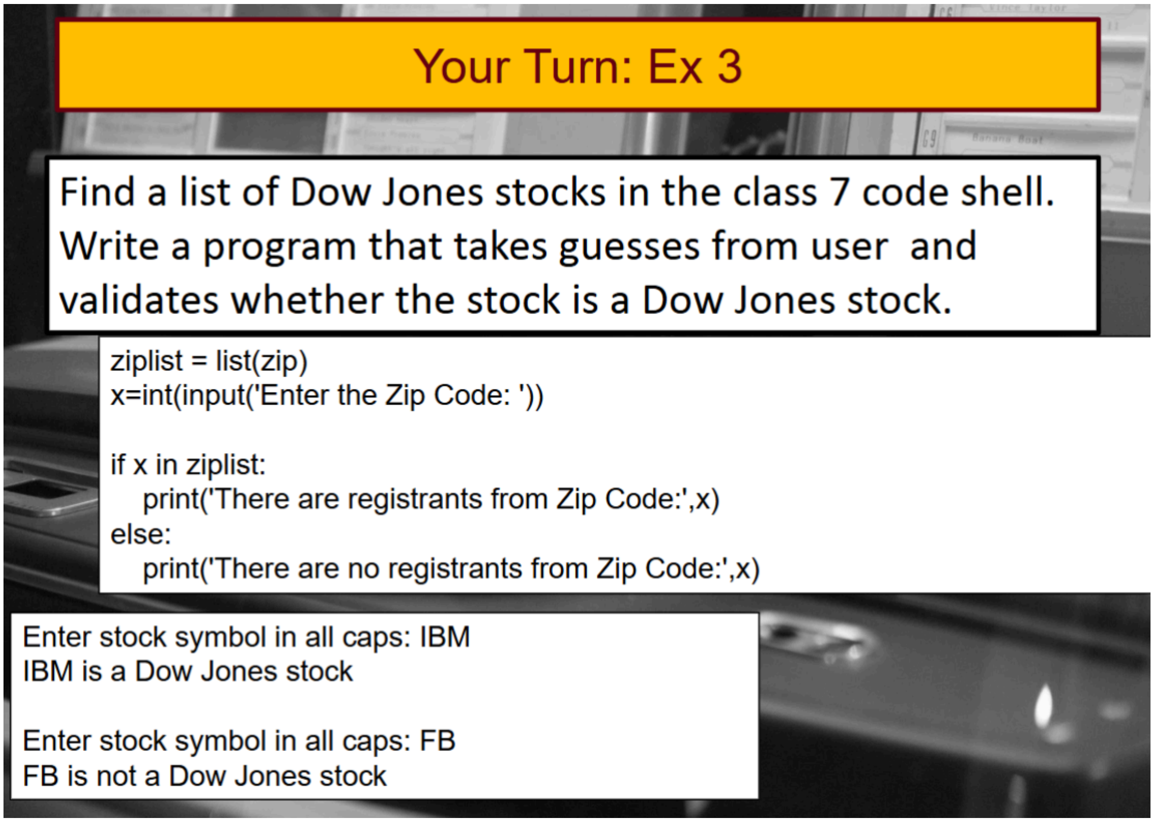
```
OUTCOME
default        36.674129
non-default    36.796082
Name: Bo_Age, dtype: float64
```

In [3]:
```
#Ex 2 Create and display two tables:

#1 Summarize mean outcome for every numeric variable
df2 = loan.groupby("OUTCOME").mean(numeric_only=True)
display(df2)
```

```
#2 Summarize the count of defaulter/non-defaulters (outcome variable)
df3 = loan.groupby("OUTCOME")['Cust_ID'].count()
display(df3)
```

	Cust_ID	Bo_Age	Ln_Orig	Orig_LTV_Ratio_Pct	Credit_score	Tot_mthl
OUTCOME						
default	9093.500000	36.674129	165255.559627	93.338308	654.231343	
non-default	8561.896888	36.796082	153146.314209	93.072809	688.579554	

```
OUTCOME
default          402
non-default    14751
Name: Cust_ID, dtype: int64
```



Your Turn: Ex 3

Find a list of Dow Jones stocks in the class 7 code shell. Write a program that takes guesses from user and validates whether the stock is a Dow Jones stock.

```
ziplist = list(zip)
x=int(input('Enter the Zip Code: '))

if x in ziplist:
    print('There are registrants from Zip Code:',x)
else:
    print('There are no registrants from Zip Code:',x)
```

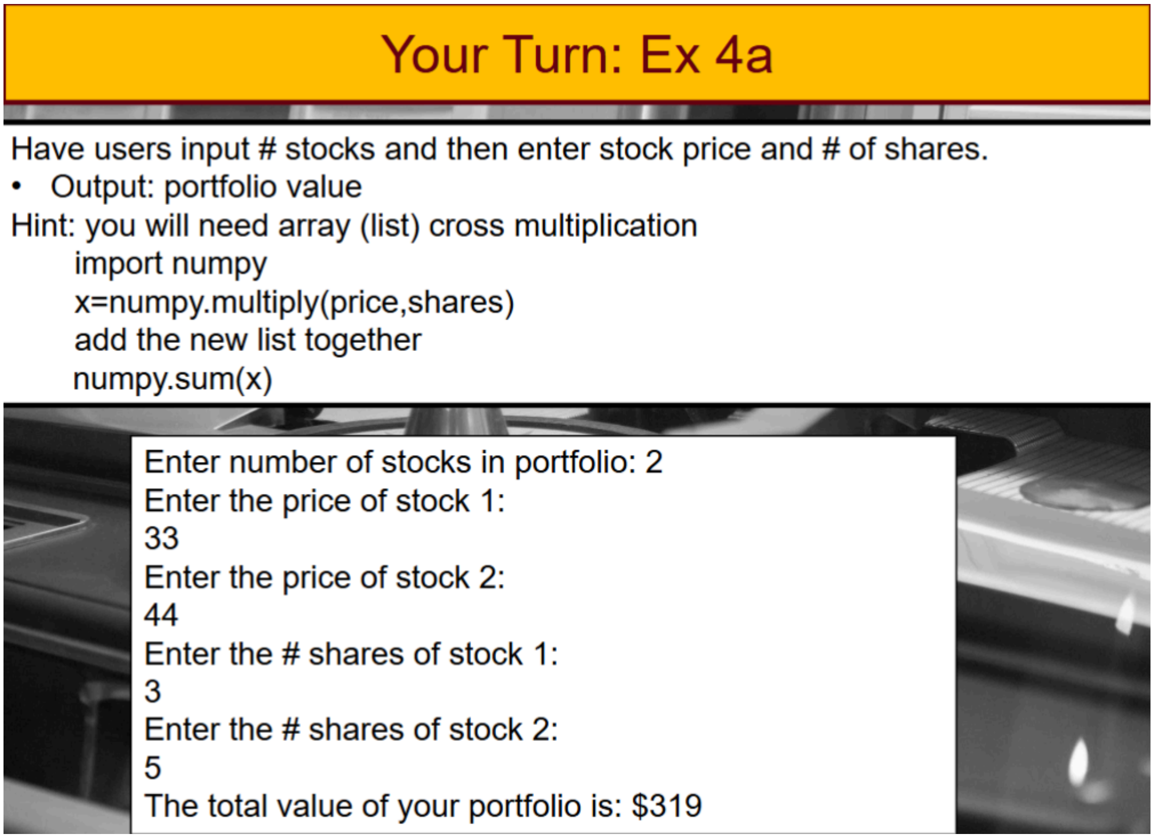
```
Enter stock symbol in all caps: IBM
IBM is a Dow Jones stock

Enter stock symbol in all caps: FB
FB is not a Dow Jones stock
```

In [4]:

```
###Ex 3 shell
dowjones = ['MMM','AXP','AAPL','BA','CAT','CVX','CSCO','KO','DIS',
            'DWDP','XOM','GE','GS','HD','IBM']

guess = str(input("Enter stock symbol in all caps: ")).upper()
print("Enter stock symbol in all caps: {}".format(guess))

if guess in dowjones:
    print("{} is a Dow Jones stock".format(guess))
```

```
else:
    print("{} is not a Dow Jones stock.".format(guess))
```

```
Enter stock symbol in all caps:
 is not a Dow Jones stock.
```



Your Turn: Ex 4a

Have users input # stocks and then enter stock price and # of shares.

- Output: portfolio value

Hint: you will need array (list) cross multiplication

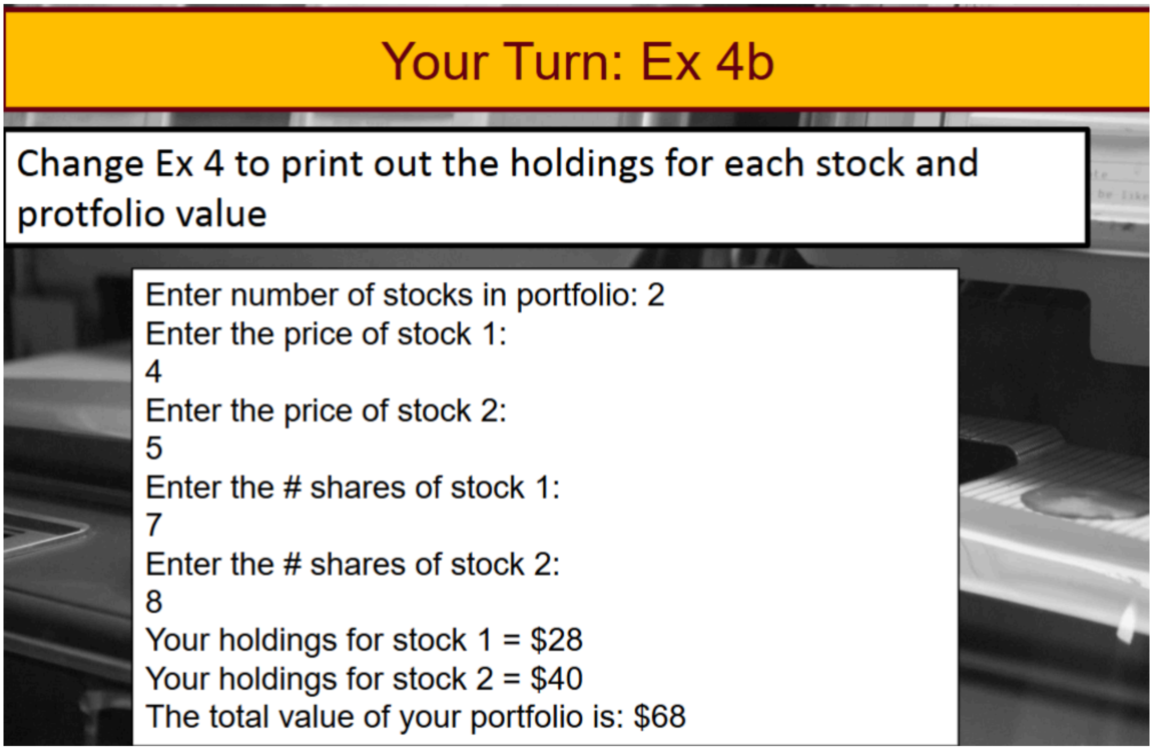
```
import numpy
x=numpy.multiply(price,shares)
add the new list together
numpy.sum(x)
```

```
Enter number of stocks in portfolio: 2
Enter the price of stock 1:
33
Enter the price of stock 2:
44
Enter the # shares of stock 1:
3
Enter the # shares of stock 2:
5
The total value of your portfolio is: $319
```

In [6]:

```
stock_num = int(input("Enter number of stocks in portfolio: "))
stock_prices = []
stock_numbers = []
total_list = []

for i in range(stock_num):
    stock_price = float(input('Enter the price of stock {}: '.format(i+1)))
    stock_prices.append(stock_price)

for i in range(stock_num):
    stock_number = float(input('Enter the # shares of stock {}: '.format(i+1)))
    stock_numbers.append(stock_number)

for i in range(len(stock_prices)):
    total = stock_prices[i]*stock_numbers[i]
    # print(total)
    total_list.append(total)

# print(total_list)

x = sum(total_list)
print("The total value of your portfolio is: ${}".format(x))
```

```
The total value of your portfolio is: $319.0
```



Your Turn: Ex 4b

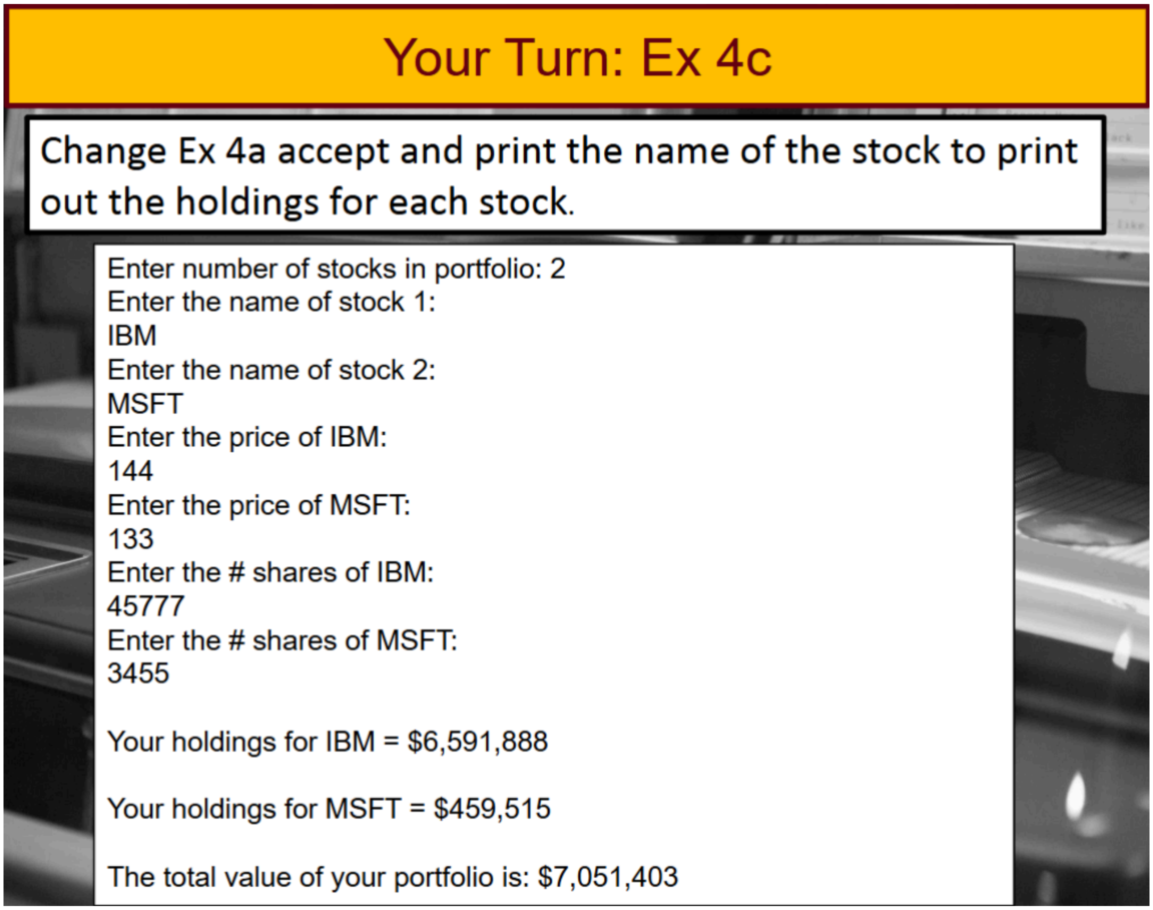
Change Ex 4 to print out the holdings for each stock and protfolio value

```
Enter number of stocks in portfolio: 2
Enter the price of stock 1:
4
Enter the price of stock 2:
5
Enter the # shares of stock 1:
7
Enter the # shares of stock 2:
8
Your holdings for stock 1 = $28
Your holdings for stock 2 = $40
The total value of your portfolio is: $68
```

In [ ]:

```python
stock_num = int(input("Enter number of stocks in portfolio: "))
stock_prices = []
stock_numbers = []
total_list = []

for i in range(stock_num):
    stock_price = float(input('Enter the price of stock {}: '.format(i+1)))
    stock_prices.append(stock_price)

for i in range(stock_num):
    stock_number = float(input('Enter the # shares of stock {}: '.format(i+1)))
    stock_numbers.append(stock_number)

for i in range(len(stock_prices)):
    total = stock_prices[i]*stock_numbers[i]
    total_list.append(total)
    print("Your holding for stock {} = ${}".format(i+1, total_list[i]))

x = sum(total_list)
print("The total value of your portfolio is: ${}".format(x))
```

```
Your holding for stock 1 = $28.0
Your holding for stock 2 = $40.0
The total value of your portfolio is: $68.0
```



Your Turn: Ex 4c

Change Ex 4a accept and print the name of the stock to print out the holdings for each stock.

```
Enter number of stocks in portfolio: 2
Enter the name of stock 1:
IBM
Enter the name of stock 2:
MSFT
Enter the price of IBM:
144
Enter the price of MSFT:
133
Enter the # shares of IBM:
45777
Enter the # shares of MSFT:
3455

Your holdings for IBM = $6,591,888

Your holdings for MSFT = $459,515

The total value of your portfolio is: $7,051,403
```

In [ ]:

```python
stock_num = int(input("Enter number of stocks in portfolio: "))
stock_names = []
stock_prices = []
stock_numbers = []
total_list = []

for i in range(stock_num):
    stock_name = input('Enter the name of stock {}: '.format(i+1))
    stock_names.append(stock_name)

for i in range(stock_num):
    stock_price = float(input('Enter the price of {}: '.format(stock_names[i])))
    stock_prices.append(stock_price)

for i in range(stock_num):
    stock_number = float(input('Enter the # shares of {}: '.format(stock_names[i]))
    stock_numbers.append(stock_number)

for i in range(len(stock_prices)):
    total = stock_prices[i]*stock_numbers[i]
    total_list.append(total)
    print("Your holding for stock {} = ${:,.0f}".format(stock_names[i], total_list[

x = sum(total_list)
print("The total value of your portfolio is: ${:,.0f}".format(x))
```

```
Your holding for stock IBM = $6,591,888
Your holding for stock MSFT = $459,515
The total value of your portfolio is: $7,051,403
```

Practise

In [ ]:

```
import pandas as pd
```

In [ ]:

```
###Panda Demos
import pandas as pd
fivek = pd.read_csv('FiveK.csv')

###is there a null?
print(pd.isnull(fivek).any()) # or use fivek.isnull().any()
display(fivek)
```

```
Event
False
Year
False
Registered
False
Gender
False
Event Age
False
Zip
False
Estimated Finish Time
False
Would you like to upgrade to the VIP Experience for the Red, White & Boom TC 5K?
True
Interested in Volunteering
True
Number of Dependents 18 and Younger Living with You
True
How did you hear about us?
False
How did you hear about us?  Other
True
Occupation
True
Highest Level of Education
True
Annual Household Income
True
I would like to Sponsor a Child for an additional $10.00
True
dtype: bool
```

	Event	Year	Registered	Gender	Event Age	Zip	Estimated Finish Time	Would you like to upgrade to the VIP Experience for the Red, White & Boom TC 5K?	Interested in Volunteering
0	TC5K	2015	2/4/2015 11:38 PM	Female	43	55379	00:00:00	NaN	No Thank You
1	TC5K	2015	2/4/2015 9:58 PM	Female	61	55104	00:00:00	NaN	No Thank You
2	TC5K	2015	2/4/2015 1:16 PM	Female	24	55128	00:00:00	NaN	No Thank You
3	TC5K	2015	2/3/2015 8:25 PM	Female	32	55431	00:00:00	NaN	Yes- I would like to be informed of volunteer ...
4	TC5K	2015	2/3/2015 8:01 PM	Female	29	55068	00:00:00	NaN	No Thank You
...	...	...	...	...	...	...	...	...	...
339	TC5K	2017	10/25/2016 3:28 PM	Male	57	55306	00:00:00	NaN	No Thank You
340	TC5K	2017	10/25/2016 2:29 PM	Female	41	55379	00:00:00	NaN	No Thank You
341	TC5K	2017	10/25/2016 12:32 PM	Female	36	54812	00:00:00	NaN	No Thank You
342	TC5K	2017	10/25/2016 12:16 PM	Male	35	55318	00:00:00	NaN	No Thank You
343	TC5K	2017	10/25/2016 12:12 PM	Female	26	55123	00:00:00	NaN	No Thank You

344 rows × 16 columns

In [ ]:

```
##drop null
# Saving in a new dataframe but not doing inplace
dropnull = fivek.dropna(subset=['Interested in Volunteering']) # this drops the row
print(pd.isnull(dropnull).any())
display(dropnull)
###notice the rows are dropped that are null.
```

```
Event
False
Year
False
Registered
False
Gender
False
Event Age
False
Zip
False
Estimated Finish Time
False
Would you like to upgrade to the VIP Experience for the Red, White & Boom TC 5K?
True
Interested in Volunteering
False
Number of Dependents 18 and Younger Living with You
True
How did you hear about us?
False
How did you hear about us?  Other
True
Occupation
True
Highest Level of Education
True
Annual Household Income
True
I would like to Sponsor a Child for an additional $10.00
True
dtype: bool
```

	Event	Year	Registered	Gender	Event Age	Zip	Estimated Finish Time	Would you like to upgrade to the VIP Experience for the Red, White & Boom TC 5K?	Interested in Volunteering
0	TC5K	2015	2/4/2015 11:38 PM	Female	43	55379	00:00:00	NaN	No Thank You
1	TC5K	2015	2/4/2015 9:58 PM	Female	61	55104	00:00:00	NaN	No Thank You
2	TC5K	2015	2/4/2015 1:16 PM	Female	24	55128	00:00:00	NaN	No Thank You
3	TC5K	2015	2/3/2015 8:25 PM	Female	32	55431	00:00:00	NaN	Yes- I would like to be informed of volunteer ...
4	TC5K	2015	2/3/2015 8:01 PM	Female	29	55068	00:00:00	NaN	No Thank You
...	...	...	...	...	...	...	...	...	...
339	TC5K	2017	10/25/2016 3:28 PM	Male	57	55306	00:00:00	NaN	No Thank You
340	TC5K	2017	10/25/2016 2:29 PM	Female	41	55379	00:00:00	NaN	No Thank You
341	TC5K	2017	10/25/2016 12:32 PM	Female	36	54812	00:00:00	NaN	No Thank You
342	TC5K	2017	10/25/2016 12:16 PM	Male	35	55318	00:00:00	NaN	No Thank You
343	TC5K	2017	10/25/2016 12:12 PM	Female	26	55123	00:00:00	NaN	No Thank You

307 rows × 16 columns

In [ ]:

```
###replace null
fillOcc=fivek #saving into a new dataframe and doing inplace in it
# Fills all the null values in Highest level of eductaion
fillOcc.fillna({'Highest Level of Education': 'Prefer not to answer'}, inplace=True
display(fillOcc)
```

	Event	Year	Registered	Gender	Event Age	Zip	Estimated Finish Time	Would you like to upgrade to the VIP Experience for the Red, White & Boom TC 5K?	Interested in Volunteering
0	TC5K	2015	2/4/2015 11:38 PM	Female	43	55379	00:00:00	NaN	No Thank You
1	TC5K	2015	2/4/2015 9:58 PM	Female	61	55104	00:00:00	NaN	No Thank You
2	TC5K	2015	2/4/2015 1:16 PM	Female	24	55128	00:00:00	NaN	No Thank You
3	TC5K	2015	2/3/2015 8:25 PM	Female	32	55431	00:00:00	NaN	Yes- I would like to be informed of volunteer ...
4	TC5K	2015	2/3/2015 8:01 PM	Female	29	55068	00:00:00	NaN	No Thank You
...	...	...	...	...	...	...	...	...	...
339	TC5K	2017	10/25/2016 3:28 PM	Male	57	55306	00:00:00	NaN	No Thank You
340	TC5K	2017	10/25/2016 2:29 PM	Female	41	55379	00:00:00	NaN	No Thank You
341	TC5K	2017	10/25/2016 12:32 PM	Female	36	54812	00:00:00	NaN	No Thank You
342	TC5K	2017	10/25/2016 12:16 PM	Male	35	55318	00:00:00	NaN	No Thank You
343	TC5K	2017	10/25/2016 12:12 PM	Female	26	55123	00:00:00	NaN	No Thank You

344 rows × 16 columns

In [ ]:

```
###Filtering and Saving
fivek1= fivek[(fivek['Event Age']>45)]
fivek2= fivek[(fivek['Event Age']>45) & (fivek['Zip']==55417)]

fivek1.to_csv("fivek1.csv")
fivek2.to_excel("fivek1.xlsx")
```

In [ ]:

```python
#Groupby Function
###Mean for all categories
fivek_avg_all= fivek.groupby("Gender").mean(numeric_only=True)
display(fivek_avg_all)
```

	Year	Event Age	Zip	Number of Dependents 18 and Younger Living with You
Gender				
Female	2016.085366	39.674797	55309.695122	0.627329
Male	2016.153061	40.173469	55288.642857	0.625000

In [ ]:

```python
###Only show mean for Event Age
fivek_age_gen= fivek.groupby("Gender")['Event Age'].mean()
display(fivek_age_gen)
```

```
Gender
Female    39.674797
Male      40.173469
Name: Event Age, dtype: float64
```

In [ ]:

```python
#Ex 1 Create and display two data frames
import pandas as pd
loan = pd.read_csv("loan default.csv")
# display(loan)
## borrowers over 35 years of age:

borrowers_over_35 = loan[(loan['Bo_Age']>35)]
display(borrowers_over_35)

##borrowers only from CA and NY:
borrowers_only_CA_NY = loan[(loan['State']=='CA') | (loan['State']=='NY')]
display(borrowers_only_CA_NY)
```

	Cust_ID	Bo_Age	Ln_Orig	Orig_LTV_Ratio_Pct	Credit_score	First_home	Tot_mthly_d
1	1001	40	168701.0	100	765	N	
4	1004	42	96000.0	100	788	N	
6	1006	53	136000.0	100	661	N	
7	1007	44	101900.0	100	788	N	
9	1009	49	161050.0	97	661	N	
...	...	...	...	...	...	...	
15146	16146	36	72000.0	100	707	N	
15147	16147	37	180000.0	90	711	Y	
15150	16150	49	259900.0	100	641	Y	
15151	16151	42	230400.0	80	682	N	
15152	16152	42	61211.0	95	779	N	

8496 rows × 16 columns

	Cust_ID	Bo_Age	Ln_Orig	Orig_LTV_Ratio_Pct	Credit_score	First_home	Tot_mthly_d
8	1008	30	106500.0	100	708	N	
16	1016	34	116000.0	100	672	N	
18	1018	38	70300.0	95	726	N	
19	1019	33	245000.0	100	615	N	
38	1038	36	269000.0	100	696	N	
...	...	...	...	...	...	...	
15148	16148	32	154850.0	95	740	N	
15149	16149	20	69000.0	100	708	Y	
15150	16150	49	259900.0	100	641	Y	
15151	16151	42	230400.0	80	682	N	
15152	16152	42	61211.0	95	779	N	

2035 rows × 16 columns