```
"""
1, Upload the data file, view the data file in Colab
2, Read and Assign it to a variable called users and use the 'user_id' as index
3, See the first 25 entries
4, See the last 10 entries
5, What is the number of observations in the dataset?
6, What is the number of columns in the dataset?
7, Print the name of all the columns.
8, How is the dataset indexed?
9, What is the data type of each column?
10, Print only the occupation column
11, How many different occupations are in this dataset?
12, What is the most frequent occupation?
13, Summarize the DataFrame.
14, Summarize all the columns.
15, Summarize only the occupation column?
16, What is the mean age of users?
17, What is the age with least occurrence?
18, Discover what is the mean age per occupation
19, Discover the Male ratio per occupation and sort it from the most to the least
20, For each occupation, calculate the minimum and maximum ages
21, For each combination of occupation and gender, calculate the mean age
22, For each occupation present the percentage of women and men
"""
```

```
'\n1, Upload the data file, view the data file in Colab\n2, Read and Assign it to a
variable called users and use the 'user_id' as index\n3, See the first 25 entries\n
4, See the last 10 entries\n5, What is the number of observations in the dataset?\n
6, What is the number of columns in the dataset?\n7, Print the name of all the colum
ns.\n8, How is the dataset indexed?\n9, What is the data type of each column?\n10, P
rint only the occupation column\n11, How many different occupations are in this data
set?\n12, What is the most frequent occupation?\n13, Summarize the DataFrame.\n14, S
ummarize all the columns.\n15, Summarize only the occupation column?\n16, What is th
```

1, Upload the data file, view the data file in Colab

done

```
import numpy as np
import pandas as pd
occ= pd.read_csv("Occupation.csv")
occ
```

	user_id|age|gender|occupation|zip_code
0	1|24|M|technician|85711
1	2|53|F|other|94043
2	3|23|M|writer|32067
3	4|24|M|technician|43537
4	5|33|F|other|15213
...	...
938	939|26|F|student|33319
940	941|20|M|student|97229

2, Read and Assign it to a variable called users and use the 'user_id' as index

```
users= pd.read_csv("Occupation.csv",sep='|')
users
```

	user_id	age	gender	occupation	zip_code
0	1	24	M	technician	85711
1	2	53	F	other	94043
2	3	23	M	writer	32067
3	4	24	M	technician	43537
4	5	33	F	other	15213
...	...	...	...	...	...
938	939	26	F	student	33319
939	940	32	M	administrator	02215
940	941	20	M	student	97229
941	942	48	F	librarian	78209
942	943	22	M	student	77841

943 rows × 5 columns

```
users.index=users['user_id']
```

```
del users['user_id']
```

```
users
```

	age	gender	occupation	zip_code
user_id				
1	24	M	technician	85711
2	53	F	other	94043
3	23	M	writer	32067
4	24	M	technician	43537
5	33	F	other	15213
...	...	...	...	...
939	26	F	student	33319
940	32	M	administrator	02215
941	20	M	student	97229
942	48	F	librarian	78209

```
users.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 943 entries, 1 to 943
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   age         943 non-null    int64 
 1   gender      943 non-null    object
 2   occupation  943 non-null    object
 3   zip_code    943 non-null    object
dtypes: int64(1), object(3)
memory usage: 36.8+ KB
```

```
users.describe()
```

	age
count	943.000000
mean	34.051962
std	12.192740
min	7.000000
25%	25.000000
50%	31.000000
75%	43.000000
max	73.000000

3, See the first 25 entries

```
users.head(25)
users.head(25)
```

	age	gender	occupation	zip_code
user_id				
1	24	M	technician	85711
2	53	F	other	94043
3	23	M	writer	32067
4	24	M	technician	43537
5	33	F	other	15213
6	42	M	executive	98101
7	57	M	administrator	91344
8	36	M	administrator	05201
9	29	M	student	01002
10	53	M	lawyer	90703
11	39	F	other	30329
12	28	F	other	06405
13	47	M	educator	29206
14	45	M	scientist	55106
15	49	F	educator	97301
16	21	M	entertainment	10309
17	30	M	programmer	06355
18	35	F	other	37212
19	40	M	librarian	02138
20	42	F	homemaker	95660
21	26	M	writer	30068
22	25	M	writer	40206
23	30	F	artist	48197
24	21	F	artist	94533
25	39	M	engineer	55107

4, See the last 10 entries

```
users.tail(10)
```

	age	gender	occupation	zip_code
user_id				
934	61	M	engineer	22902
935	42	M	doctor	66221
936	24	M	other	32789
937	48	M	educator	98072
938	38	F	technician	55038
939	26	F	student	33319
940	32	M	administrator	02215
941	20	M	student	97229
942	48	F	librarian	78209

5, What is the number of observations in the dataset?

```
users.shape
```

```
(943, 4)
```

```
users.shape[0]
```

```
943
```

6, What is the number of columns in the dataset?

```
users.shape[1]
```

```
4
```

7, Print the name of all the columns.

```
users.columns
```

```
Index(['age', 'gender', 'occupation', 'zip_code'], dtype='object')
```

8, How is the dataset indexed?

```
users.index
```

```
Int64Index([  1,   2,   3,   4,   5,   6,   7,   8,   9,  10,
            ...
            934, 935, 936, 937, 938, 939, 940, 941, 942, 943],
           dtype='int64', name='user_id', length=943)
```

9, What is the data type of each column?

```
users.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 943 entries, 1 to 943
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   age         943 non-null    int64 
 1   gender      943 non-null    object
 2   occupation  943 non-null    object
 3   zip_code    943 non-null    object
dtypes: int64(1), object(3)
memory usage: 36.8+ KB
```

10, Print only the occupation column

```
users.occupation
```

```
user_id
1         technician
2              other
3             writer
4         technician
5              other
           ...      
939          student
940    administrator
941          student
942        librarian
943          student
Name: occupation, Length: 943, dtype: object
```

11, How many different occupations are in this dataset?

12, What is the most frequent occupation?

```
users.occupation.describe()
```

```
count         943
unique         21
top       student
freq          196
Name: occupation, dtype: object
```

```
users.occupation.value_counts().count()
```

```
21
```

```
users.occupation.value_counts().count()
```

```
21
```

```
users.occupation.unique()
```

```
array(['technician', 'other', 'writer', 'executive', 'administrator',
       'student', 'lawyer', 'educator', 'scientist', 'entertainment',
       'programmer', 'librarian', 'homemaker', 'artist', 'engineer',
       'marketing', 'none', 'healthcare', 'retired', 'salesman', 'doctor'],
      dtype=object)
```

```
users.occupation.nunique()
```

```
21
```

```
occfreq= users.groupby('occupation')['occupation'].count()
occfreq
```

```
occupation
administrator     79
artist            28
doctor             7
educator          95
engineer          67
entertainment     18
executive         32
healthcare        16
homemaker          7
lawyer            12
librarian         51
marketing         26
none               9
other            105
programmer        66
retired           14
salesman          12
scientist         31
student          196
technician        27
writer            45
Name: occupation, dtype: int64
```

```
  occfreq.sort_values()
```

```
occupation
doctor             7
homemaker          7
none               9
salesman          12
lawyer            12
retired           14
healthcare        16
entertainment     18
marketing         26
technician        27
artist            28
```

```
scientist         31
executive         32
writer            45
librarian         51
programmer        66
engineer          67
administrator     79
educator          95
other            105
student          196
Name: occupation, dtype: int64
```

13, Summarize the DataFrame.

```
users.describe()
```

	age
count	943.000000
mean	34.051962
std	12.192740
min	7.000000
25%	25.000000
50%	31.000000
75%	43.000000
max	73.000000

14, Summarize all the columns.

```
users.describe(include='all')
```

	age	gender	occupation	zip_code
count	943.000000	943	943	943

15, Summarize only the occupation column?

top NaN M student 55414

```
users.occupation.describe()
```

```
count         943
unique         21
top       student
freq          196
Name: occupation, dtype: object
```

25% 25.000000 NaN NaN NaN

16, What is the mean age of users?

75% 43.000000 NaN NaN NaN

```
users.age.mean()
```

```
34.05196182396607
```

17, What is the age with least occurrence?

```
users.age.describe()
```

```
count    943.000000
mean      34.051962
std       12.192740
min        7.000000
25%       25.000000
50%       31.000000
75%       43.000000
max       73.000000
Name: age, dtype: float64
```

```
agegroup = users.groupby('age')['age'].count()
agegroup
```

```
age
7     1
10    1
11    1
13    5
14    3
     ..
66    1
68    2
69    2
70    3
73    1
Name: age, Length: 61, dtype: int64
```

```
agegroup[agegroup==agegroup.min()]
```

```
age
7     1
10    1
11    1
66    1
73    1
Name: age, dtype: int64
```

8, Discover what is the mean age per occupation

```
users.groupby('occupation')['age'].mean().sort_index()
```

```
occupation
administrator    38.746835
artist           31.392857
doctor           43.571429
educator         42.010526
engineer         36.388060
entertainment    29.222222
executive        38.718750
healthcare       41.562500
homemaker        32.571429
lawyer           36.750000
librarian        40.000000
marketing        37.615385
none             26.555556
other            34.523810
programmer       33.121212
retired          63.071429
salesman         35.666667
scientist        35.548387
student          22.081633
technician       33.148148
writer           36.311111
Name: age, dtype: float64
```

19, Discover the Male ratio per occupation and sort it from the most to the least

```
pd.get_dummies(users.gender)
```

	F	M
user_id		
1	0	1
2	1	0
3	0	1
4	0	1
5	1	0

```
usersc=users.copy()
usersc
```

	age	gender	occupation	zip_code
user_id				
1	24	M	technician	85711
2	53	F	other	94043
3	23	M	writer	32067
4	24	M	technician	43537
5	33	F	other	15213
...	...	...	...	...
939	26	F	student	33319
940	32	M	administrator	02215
941	20	M	student	97229
942	48	F	librarian	78209
943	22	M	student	77841

943 rows × 4 columns

```
usersc=pd.concat([usersc, pd.get_dummies(users.gender)], axis='columns')
usersc
```

	age	gender	occupation	zip_code	F	M
user_id						
1	24	M	technician	85711	0	1
2	53	F	other	94043	1	0
3	23	M	writer	32067	0	1
4	24	M	technician	43537	0	1
5	33	F	other	15213	1	0

```
mratio=usersc.groupby('occupation')['M'].sum()/users.groupby('occupation')['gender'].c
mratio
```

```
occupation
administrator    0.544304
artist           0.535714
doctor           1.000000
educator         0.726316
engineer         0.970149
entertainment    0.888889
executive        0.906250
healthcare       0.312500
homemaker        0.142857
lawyer           0.833333
librarian        0.431373
marketing        0.615385
none             0.555556
other            0.657143
programmer       0.909091
retired          0.928571
salesman         0.750000
scientist        0.903226
student          0.693878
technician       0.962963
writer           0.577778
dtype: float64
```

```
mratio.sort_values(ascending=False)
```

```
occupation
doctor           1.000000
engineer         0.970149
technician       0.962963
retired          0.928571
programmer       0.909091
executive        0.906250
scientist        0.903226
entertainment    0.888889
lawyer           0.833333
salesman         0.750000
educator         0.726316
student          0.693878
other            0.657143
marketing        0.615385
writer           0.577778
none             0.555556
```

```
administrator    0.544304
artist           0.535714
librarian        0.431373
healthcare       0.312500
homemaker        0.142857
dtype: float64
```

alternate

```

```

20, For each occupation, calculate the minimum and maximum ages

```
occMinMax=pd.concat([
                     usersc.groupby('occupation')['age'].min(),
                     usersc.groupby('occupation')['age'].max()],
                    axis='columns')
occMinMax
```

	age	age
occupation		
administrator	21	70

```
occMinMax.columns=['Min_Age','Max_Age']
occMinMax
```

	Min_Age	Max_Age
occupation		
administrator	21	70
artist	19	48
doctor	28	64
educator	23	63
engineer	22	70
entertainment	15	50
executive	22	69
healthcare	22	62
homemaker	20	50
lawyer	21	53
librarian	23	69
marketing	24	55
none	11	55
other	13	64
programmer	20	63
retired	51	73
salesman	18	66
scientist	23	55
student	7	42
technician	21	55
writer	18	60

```
#alternate
users.groupby('occupation')['age'].agg(['min','max'])
```

	min	max
occupation		
administrator	21	70
artist	19	48
doctor	28	64
educator	23	63
engineer	22	70
entertainment	15	50
executive	22	69
healthcare	22	62
homemaker	20	50
lawyer	21	53
librarian	23	69
marketing	24	55
none	11	55
other	13	64
programmer	20	63
retired	51	73
salesman	18	66
scientist	23	55
student	7	42
technician	21	55

21, For each combination of occupation and gender, calculate the mean age

```
round(usersc.groupby(['occupation','gender'])['age'].mean())
```

```
occupation     gender
administrator  F         41.0
               M         37.0
artist         F         30.0
               M         32.0
doctor         M         44.0
educator       F         39.0
               M         43.0
engineer       F         30.0
               M         37.0
entertainment  F         31.0
               M         29.0
executive      F         44.0
               M         38.0
healthcare     F         40.0
```

```
               M         45.0
homemaker      F         34.0
               M         23.0
lawyer         F         40.0
               M         36.0
librarian      F         40.0
               M         40.0
marketing      F         37.0
               M         38.0
none           F         36.0
               M         19.0
other          F         35.0
               M         34.0
programmer     F         32.0
               M         33.0
retired        F         70.0
               M         63.0
salesman       F         27.0
               M         39.0
scientist      F         28.0
               M         36.0
student        F         21.0
               M         23.0
technician     F         38.0
               M         33.0
writer         F         38.0
               M         35.0
Name: age, dtype: float64
```

22, For each occupation present the percentage of women and men

```
occGen=pd.concat([
                  usersc.groupby('occupation')['M'].sum(),
                  usersc.groupby('occupation')['F'].sum()],
                  axis='columns')
occGen
```

	M	F
occupation		
administrator	43	36
artist	15	13
doctor	7	0
educator	69	26
engineer	65	2
entertainment	16	2
executive	29	3
healthcare	5	11
homemaker	1	6
lawyer	10	2
librarian	22	29

```
occGen['Men%']=round(occGen['M']/(occGen['M']+occGen['F']),2)*100
occGen['WoMen%']=round(occGen['F']/(occGen['M']+occGen['F']),2)*100
occGen
```

	M	F	Men%	WoMen%
occupation				
administrator	43	36	54.0	46.0
artist	15	13	54.0	46.0
doctor	7	0	100.0	0.0
educator	69	26	73.0	27.0
engineer	65	2	97.0	3.0
entertainment	16	2	89.0	11.0
executive	29	3	91.0	9.0
healthcare	5	11	31.0	69.0
homemaker	1	6	14.0	86.0
lawyer	10	2	83.0	17.0
librarian	22	29	43.0	57.0
marketing	16	10	62.0	38.0
none	5	4	56.0	44.0
other	69	36	66.0	34.0
programmer	60	6	91.0	9.0
retired	13	1	93.0	7.0
salesman	9	3	75.0	25.0