Министерство образования Республики Беларусь

Учреждение образования

“Брестский государственный университет”

Кафедра ИИТ

Лабораторная работа №8

По дисциплине “Языки программирования”

Выполнил:

Студент группы ПО-7

Угляница И.Н

Проверил:

Бойко Д.О

Брест 2021

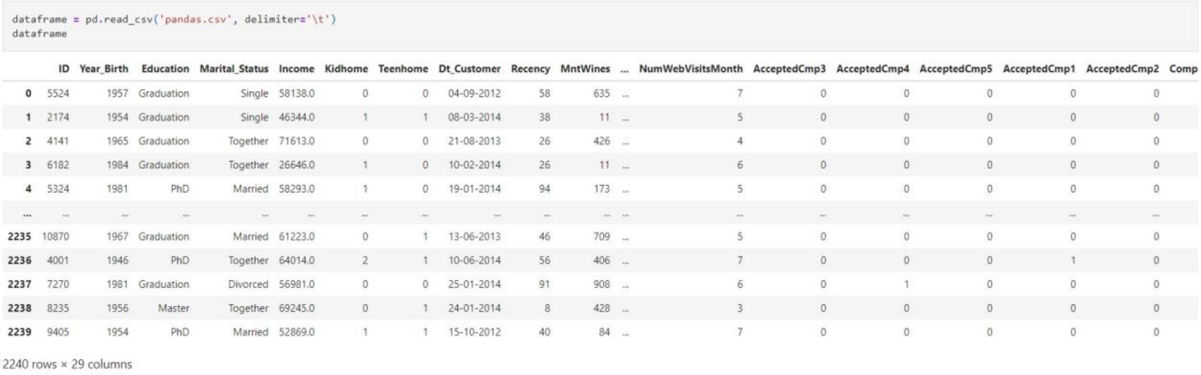
Цель работы: ознакомиться с основами библиотеки pandas и научиться строить графики с использованием библиотек matplotlib.pyplot и seaborn

Ход работы:

1. Загрузить датасет в pandas и проверить на доступность

   Все четко, все доступно.

2. Вывести общую информацию о датасете

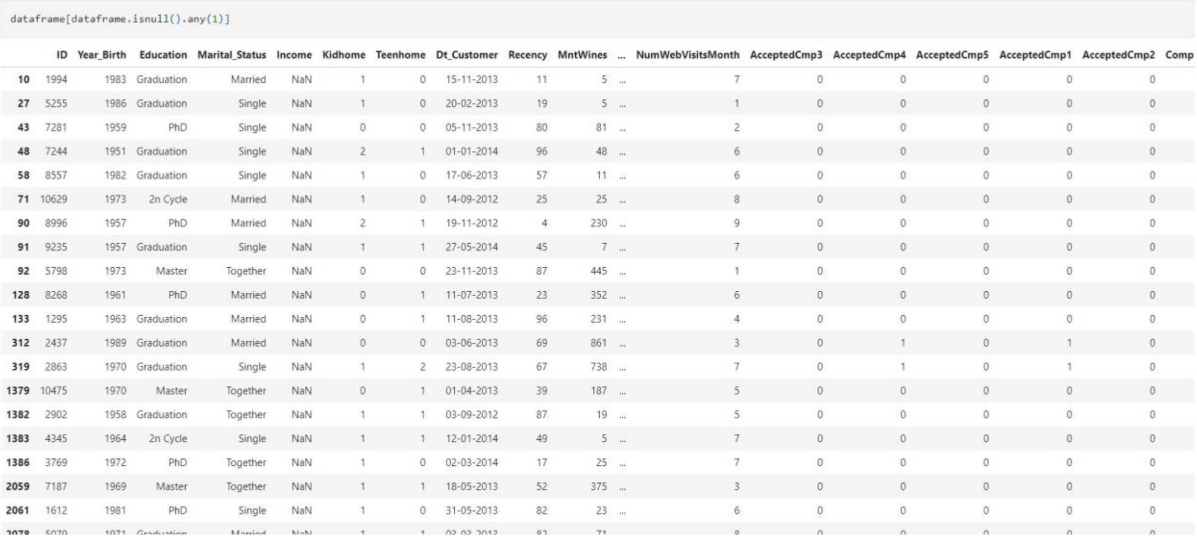


```
dataframe = pd.read_csv('pandas.csv', delimiter='\t')
dataframe
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumWebVisitsMonth	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5	AcceptedCmp1	AcceptedCmp2	Comp
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	58	635	...	7	0	0	0	0	0	
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	38	11	...	5	0	0	0	0	0	
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	26	426	...	4	0	0	0	0	0	
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	26	11	...	6	0	0	0	0	0	
4	5324	1981	PhD	Married	58293.0	1	0	19-01-2014	94	173	...	5	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
2235	10870	1967	Graduation	Married	61223.0	0	1	13-06-2013	46	709	...	5	0	0	0	0	0	
2236	4001	1946	PhD	Together	64014.0	2	1	10-06-2014	56	406	...	7	0	0	0	1	0	
2237	7270	1981	Graduation	Divorced	56981.0	0	0	25-01-2014	91	908	...	6	0	1	0	0	0	
2238	8235	1956	Master	Together	69245.0	0	1	24-01-2014	8	428	...	3	0	0	0	0	0	
2239	9405	1954	PhD	Married	52869.0	1	1	15-10-2012	40	84	...	7	0	0	0	0	0	

2240 rows × 29 columns

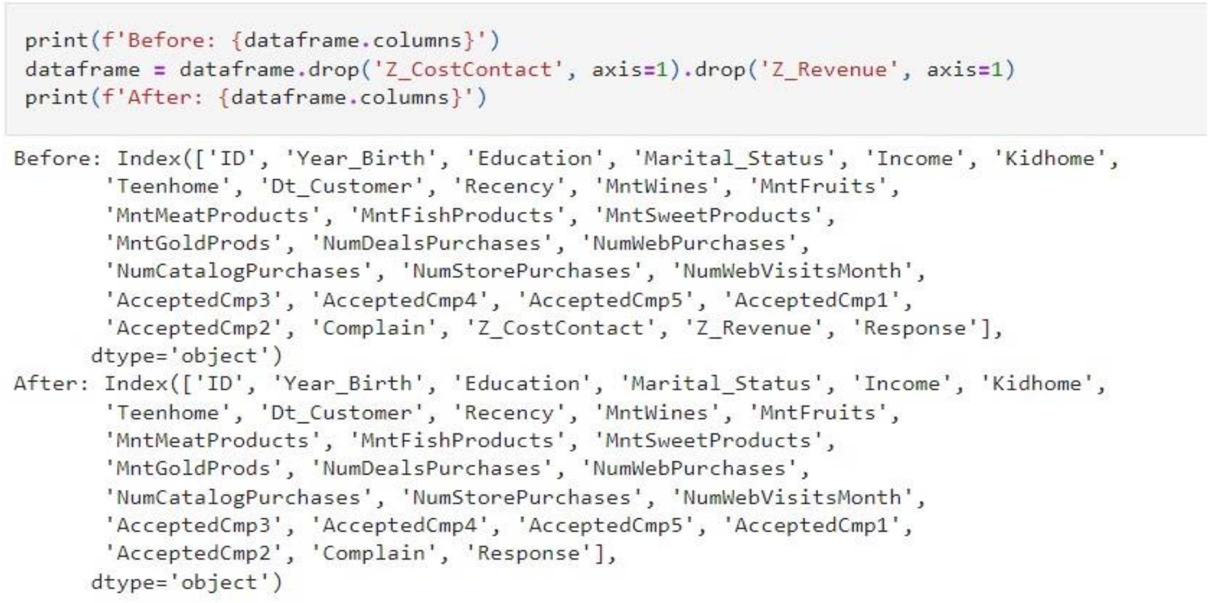
3. Проверка наличия NULL-данных. При их наличии вывести на экран



```
dataframe[dataframe.isnull().any(1)]
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumWebVisitsMonth	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5	AcceptedCmp1	AcceptedCmp2	Comp
10	1994	1983	Graduation	Married	NaN	1	0	15-11-2013	11	5	...	7	0	0	0	0	0	
27	5255	1986	Graduation	Single	NaN	1	0	20-02-2013	19	5	...	1	0	0	0	0	0	
43	7281	1959	PhD	Single	NaN	0	0	05-11-2013	80	81	...	2	0	0	0	0	0	
48	7244	1951	Graduation	Single	NaN	2	1	01-01-2014	96	48	...	6	0	0	0	0	0	
58	8557	1982	Graduation	Single	NaN	1	0	17-06-2013	57	11	...	6	0	0	0	0	0	
71	10629	1973	2n Cycle	Married	NaN	1	0	14-09-2012	25	25	...	8	0	0	0	0	0	
90	8996	1957	PhD	Married	NaN	2	1	19-11-2012	4	230	...	9	0	0	0	0	0	
91	9235	1957	Graduation	Single	NaN	1	1	27-05-2014	45	7	...	7	0	0	0	0	0	
92	5798	1973	Master	Together	NaN	0	0	23-11-2013	87	445	...	1	0	0	0	0	0	
128	8268	1961	PhD	Married	NaN	0	1	11-07-2013	23	352	...	6	0	0	0	0	0	
133	1295	1963	Graduation	Married	NaN	0	1	11-08-2013	96	231	...	4	0	0	0	0	0	
312	2437	1989	Graduation	Married	NaN	0	0	03-06-2013	69	861	...	3	0	1	0	1	0	
319	2863	1970	Graduation	Single	NaN	1	2	23-08-2013	67	738	...	7	0	1	0	1	0	
1379	10475	1970	Master	Together	NaN	0	1	01-04-2013	39	187	...	5	0	0	0	0	0	
1382	2902	1958	Graduation	Together	NaN	1	1	03-09-2012	87	19	...	5	0	0	0	0	0	
1383	4345	1964	2n Cycle	Single	NaN	1	1	12-01-2014	49	5	...	7	0	0	0	0	0	
1386	3769	1972	PhD	Together	NaN	1	0	02-03-2014	17	25	...	7	0	0	0	0	0	
2059	7187	1969	Master	Together	NaN	1	1	18-05-2013	52	375	...	3	0	0	0	0	0	
2061	1612	1981	PhD	Single	NaN	1	0	31-05-2013	82	23	...	6	0	0	0	0	0	
2078	5079	1971	Graduation	Married	NaN	1	1	03-03-2013	83	71	...	8	0	0	0	0	0	

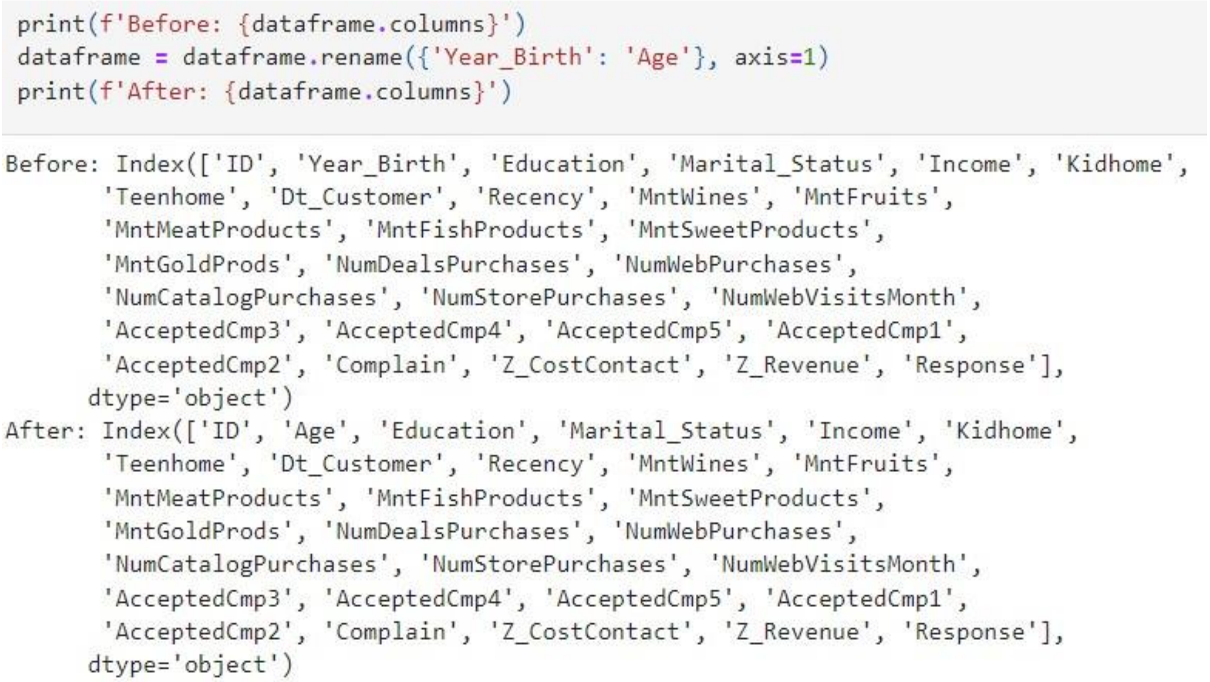
4. Удалить колонки "Z_CostContact", "Z_Revenue"



```
print(f'Before: {dataframe.columns}')
dataframe = dataframe.drop('Z_CostContact', axis=1).drop('Z_Revenue', axis=1)
print(f'After: {dataframe.columns}')
```

```
Before: Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',
       'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
       'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response'],
      dtype='object')
After: Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',
       'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
       'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Response'],
      dtype='object')
```

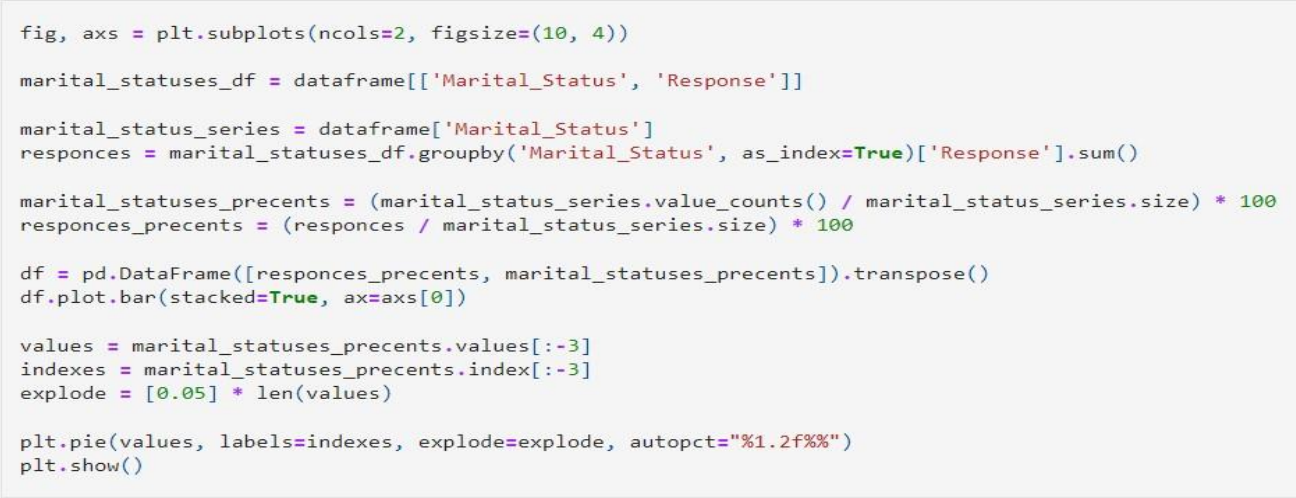
5. Переименовать колонку "Year_Birth" в "Age"



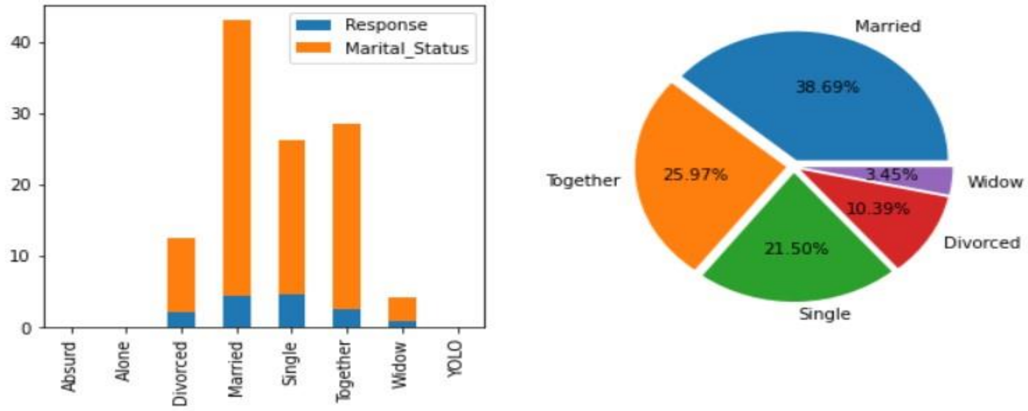
```
print(f'Before: {dataframe.columns}')
dataframe = dataframe.rename({'Year_Birth': 'Age'}, axis=1)
print(f'After: {dataframe.columns}')
```

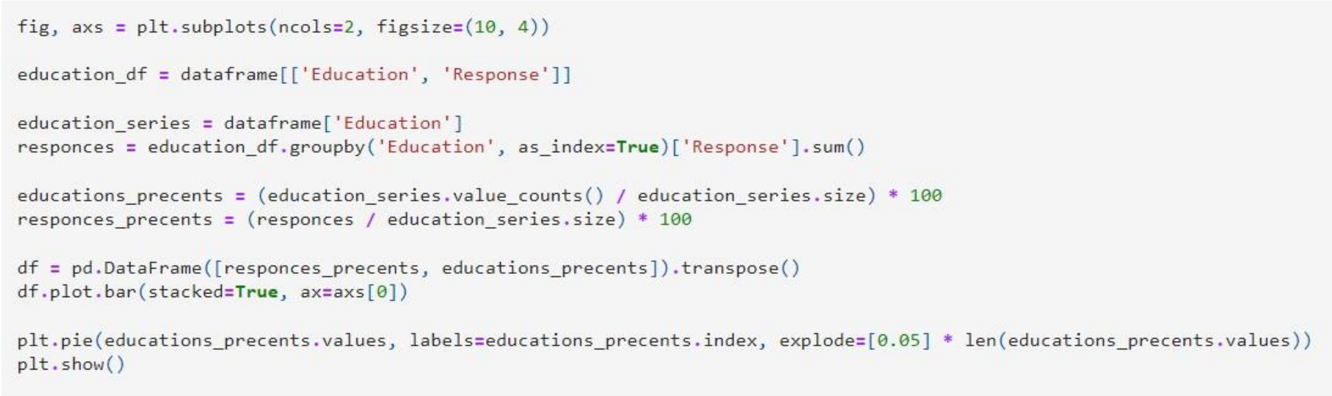
```
Before: Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',
       'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
       'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response'],
      dtype='object')
After: Index(['ID', 'Age', 'Education', 'Marital_Status', 'Income', 'Kidhome',
       'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
       'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response'],
      dtype='object')
```

6. Оценить состояние колонок "Marital_Status", "Education". Построить информативные диаграммы и гистограммы для каждой.

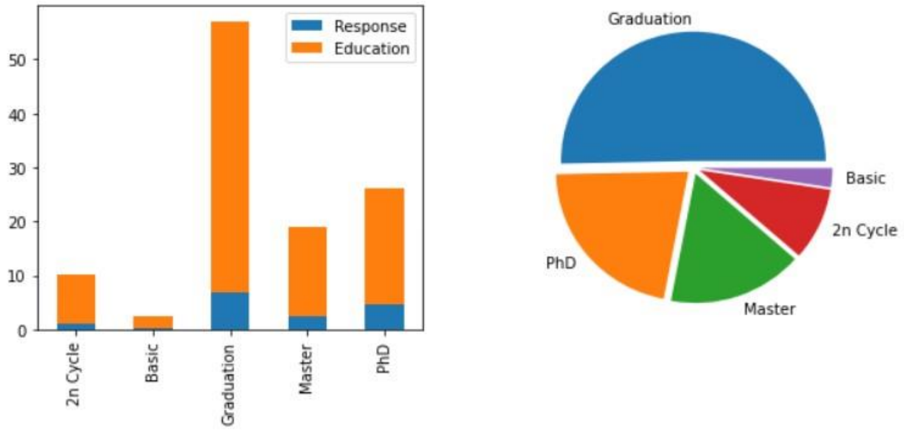


```
fig, axs = plt.subplots(ncols=2, figsize=(10, 4))

marital_statuses_df = dataframe[['Marital_Status', 'Response']]

marital_status_series = dataframe['Marital_Status']
responces = marital_statuses_df.groupby('Marital_Status', as_index=True)['Response'].sum()

marital_statuses_precents = (marital_status_series.value_counts() / marital_status_series.size) * 100
responces_precents = (responces / marital_status_series.size) * 100

df = pd.DataFrame([responces_precents, marital_statuses_precents]).transpose()
df.plot.bar(stacked=True, ax=axs[0])

values = marital_statuses_precents.values[:-3]
indexes = marital_statuses_precents.index[:-3]
explode = [0.05] * len(values)

plt.pie(values, labels=indexes, explode=explode, autopct="%1.2f%%")
plt.show()
```

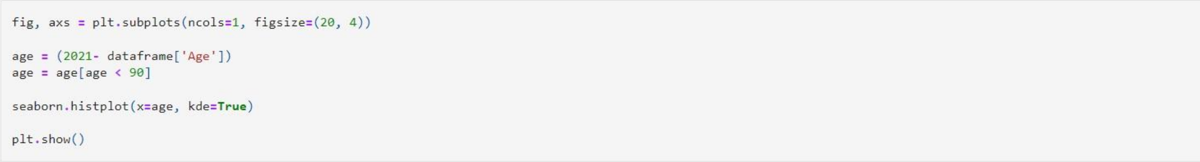




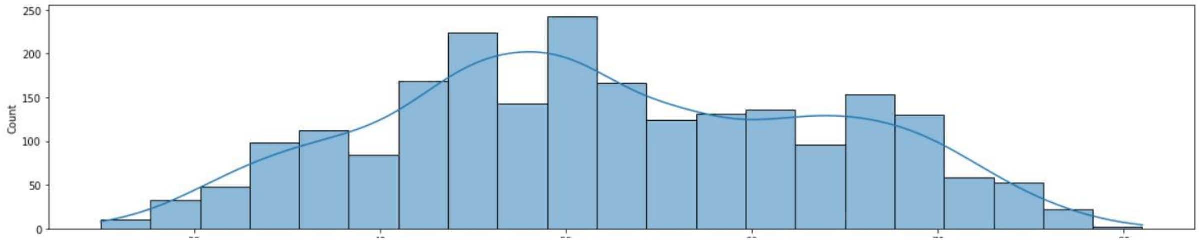
```
fig, axs = plt.subplots(ncols=2, figsize=(10, 4))

education_df = dataframe[['Education', 'Response']]

education_series = dataframe['Education']
responces = education_df.groupby('Education', as_index=True)['Response'].sum()

educations_precents = (education_series.value_counts() / education_series.size) * 100
responces_precents = (responces / education_series.size) * 100

df = pd.DataFrame([responces_precents, educations_precents]).transpose()
df.plot.bar(stacked=True, ax=axs[0])

plt.pie(educations_precents.values, labels=educations_precents.index, explode=[0.05] * len(educations_precents.values))
plt.show()
```



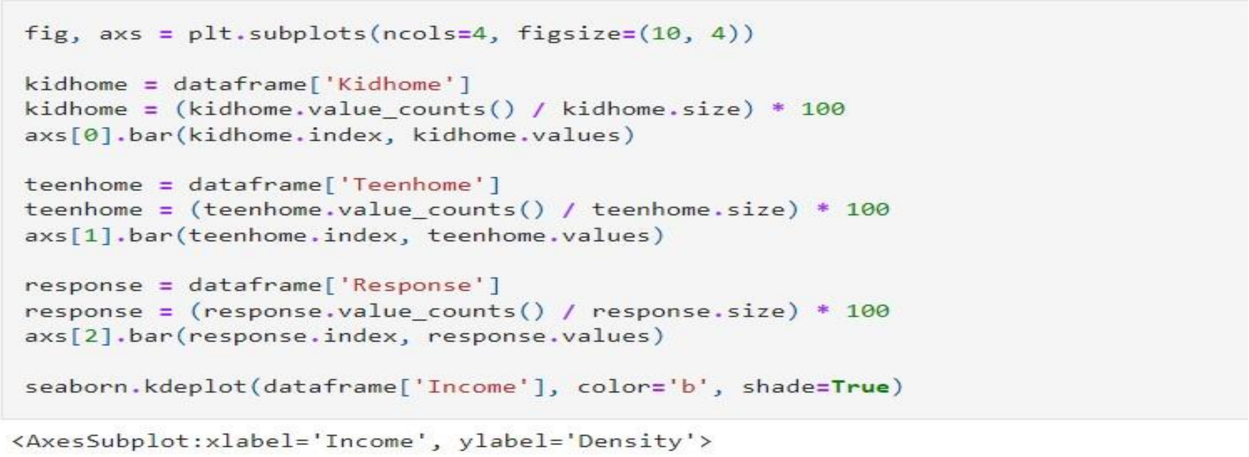
7. Создать гистаграмму по колонке "Age" и оценить на распределение по Гауссу.



```
fig, axs = plt.subplots(ncols=1, figsize=(20, 4))

age = (2021- dataframe['Age'])
age = age[age < 90]

seaborn.histplot(x=age, kde=True)

plt.show()
```

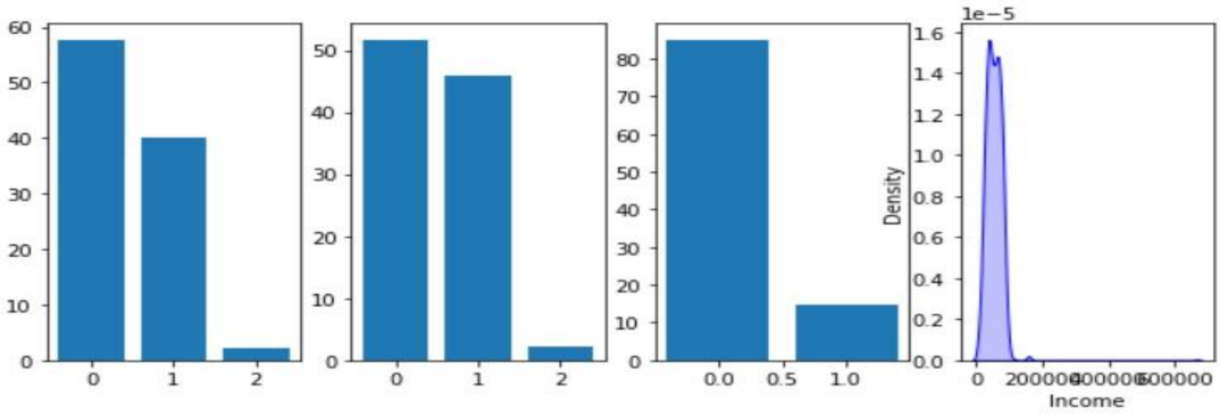


8. Оценка полей "Kidhome" и "Teenhome", "Response" и "Income" (диаграммы и гистограммы)

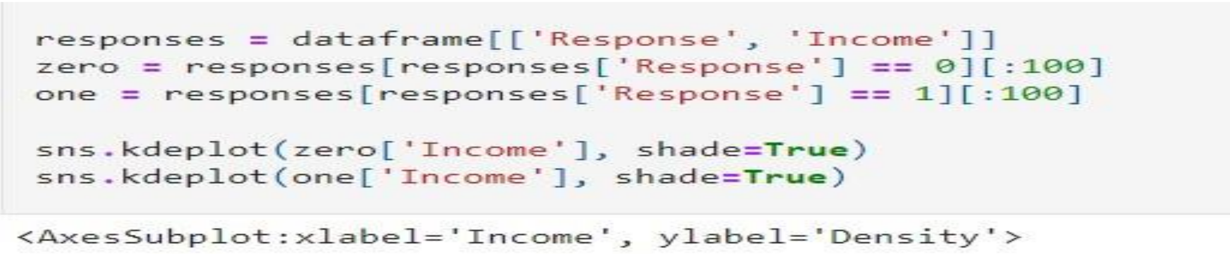


```
fig, axs = plt.subplots(ncols=4, figsize=(10, 4))

kidhome = dataframe['Kidhome']
kidhome = (kidhome.value_counts() / kidhome.size) * 100
axs[0].bar(kidhome.index, kidhome.values)

teenhome = dataframe['Teenhome']
teenhome = (teenhome.value_counts() / teenhome.size) * 100
axs[1].bar(teenhome.index, teenhome.values)

response = dataframe['Response']
response = (response.value_counts() / response.size) * 100
axs[2].bar(response.index, response.values)

seaborn.kdeplot(dataframe['Income'], color='b', shade=True)
```

```
<AxesSubplot:xlabel='Income', ylabel='Density'>
```

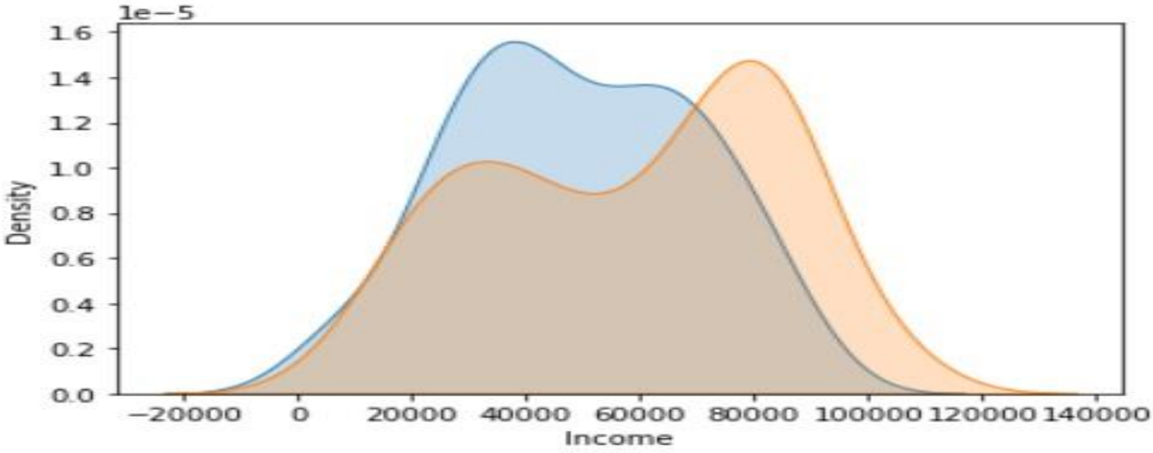


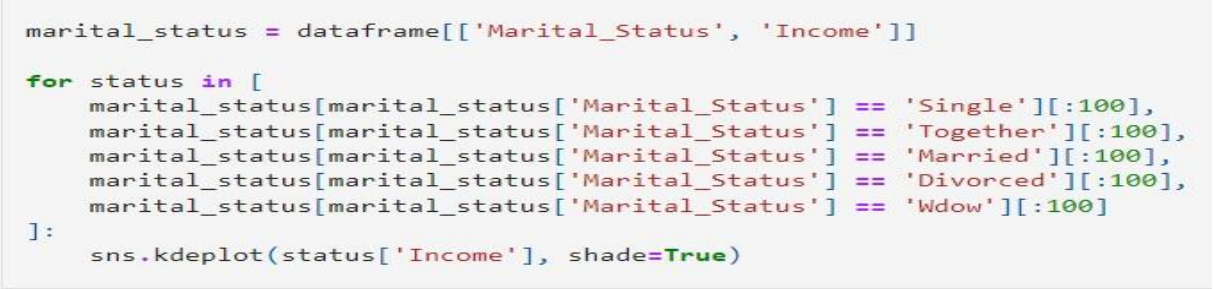
9.	Построить графики "Response", "Marital_Status", "Education" и "Kidhome"



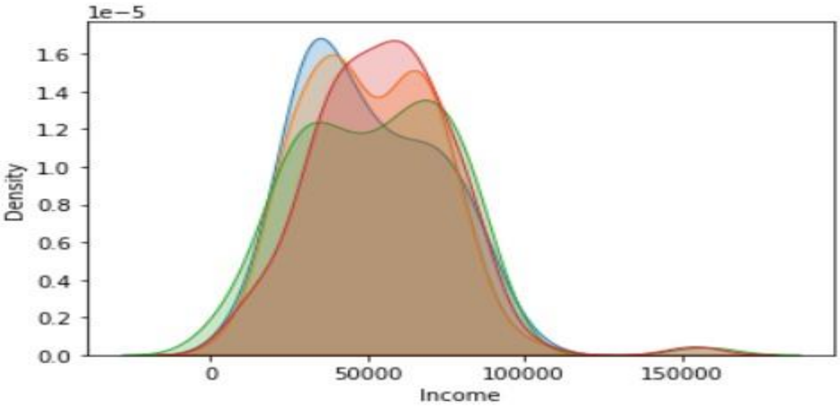
```
responses = dataframe[['Response', 'Income']]
zero = responses[responses['Response'] == 0][:100]
one = responses[responses['Response'] == 1][:100]

sns.kdeplot(zero['Income'], shade=True)
sns.kdeplot(one['Income'], shade=True)
```

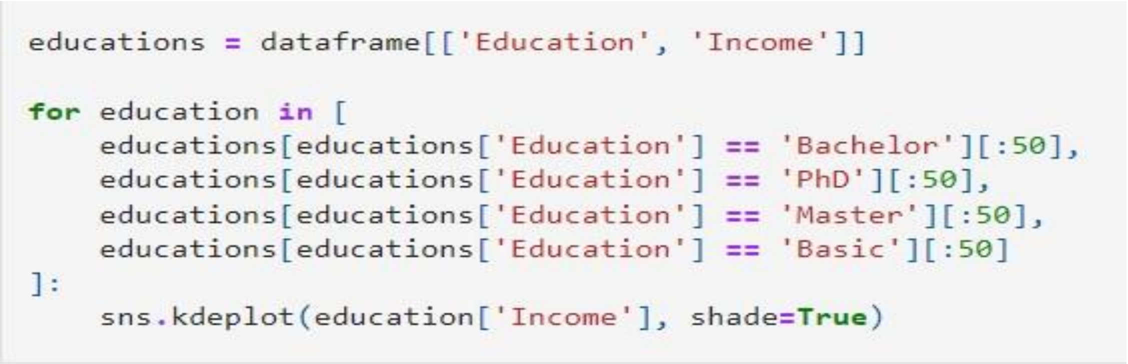
```
<AxesSubplot:xlabel='Income', ylabel='Density'>
```



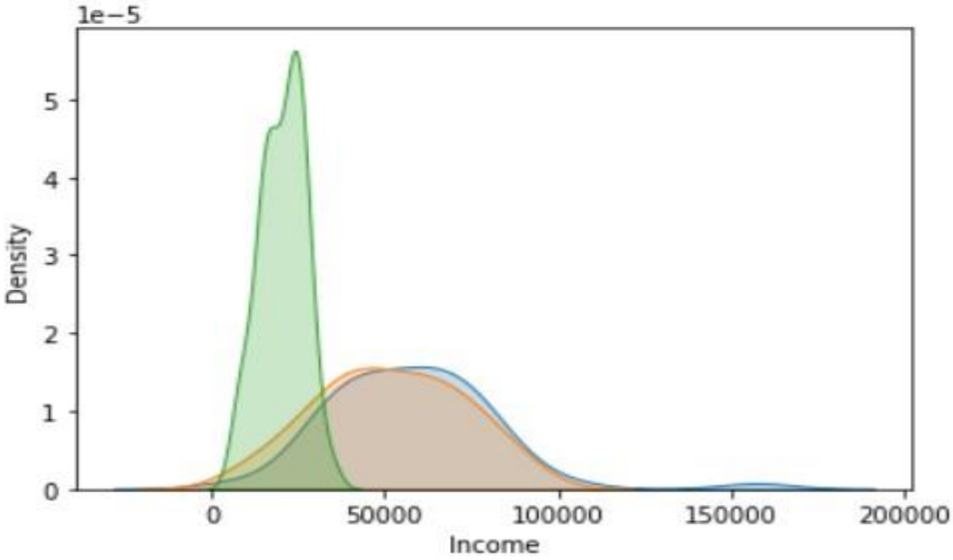


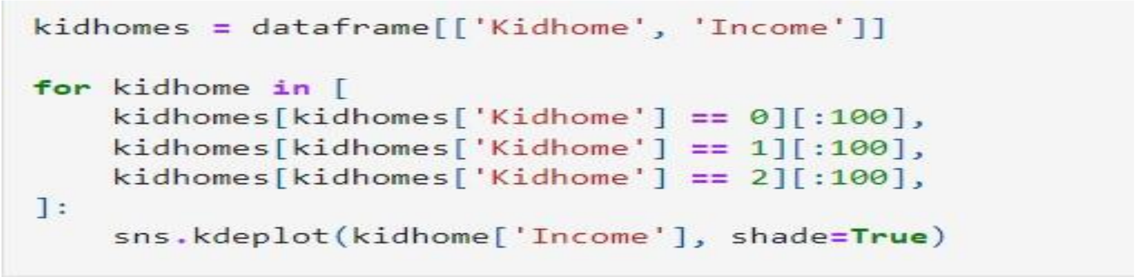
```
marital_status = dataframe[['Marital_Status', 'Income']]

for status in [
    marital_status[marital_status['Marital_Status'] == 'Single'][:100],
    marital_status[marital_status['Marital_Status'] == 'Together'][:100],
    marital_status[marital_status['Marital_Status'] == 'Married'][:100],
    marital_status[marital_status['Marital_Status'] == 'Divorced'][:100],
    marital_status[marital_status['Marital_Status'] == 'Wdow'][:100]
]:
    sns.kdeplot(status['Income'], shade=True)
```



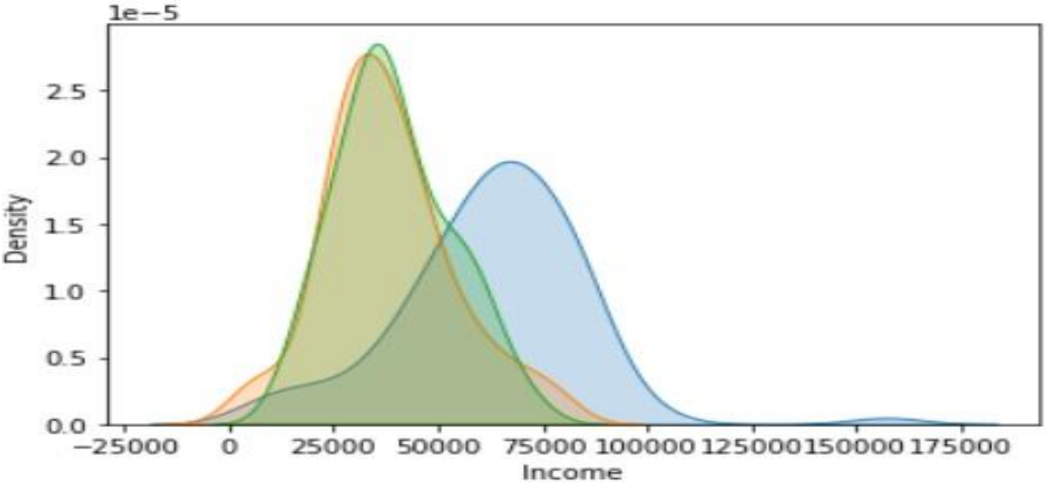


```
educations = dataframe[['Education', 'Income']]

for education in [
    educations[educations['Education'] == 'Bachelor'][:50],
    educations[educations['Education'] == 'PhD'][:50],
    educations[educations['Education'] == 'Master'][:50],
    educations[educations['Education'] == 'Basic'][:50]
]:
    sns.kdeplot(education['Income'], shade=True)
```

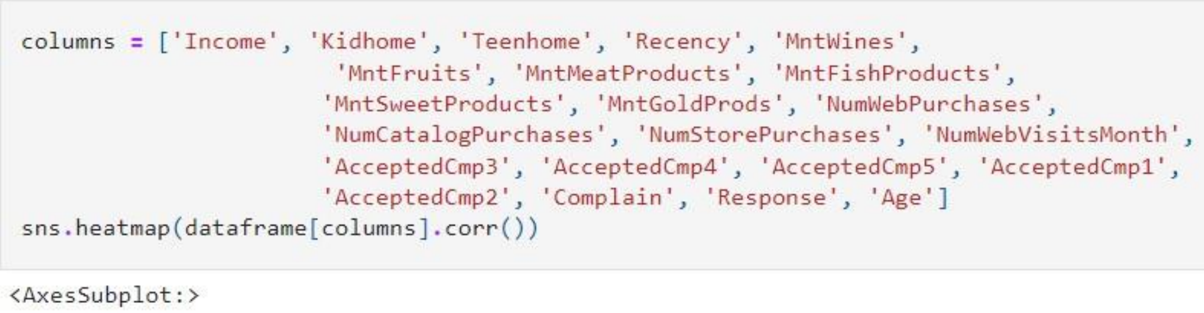




```
kidhomes = dataframe[['Kidhome', 'Income']]

for kidhome in [
    kidhomes[kidhomes['Kidhome'] == 0][:100],
    kidhomes[kidhomes['Kidhome'] == 1][:100],
    kidhomes[kidhomes['Kidhome'] == 2][:100],
]:
    sns.kdeplot(kidhome['Income'], shade=True)
```

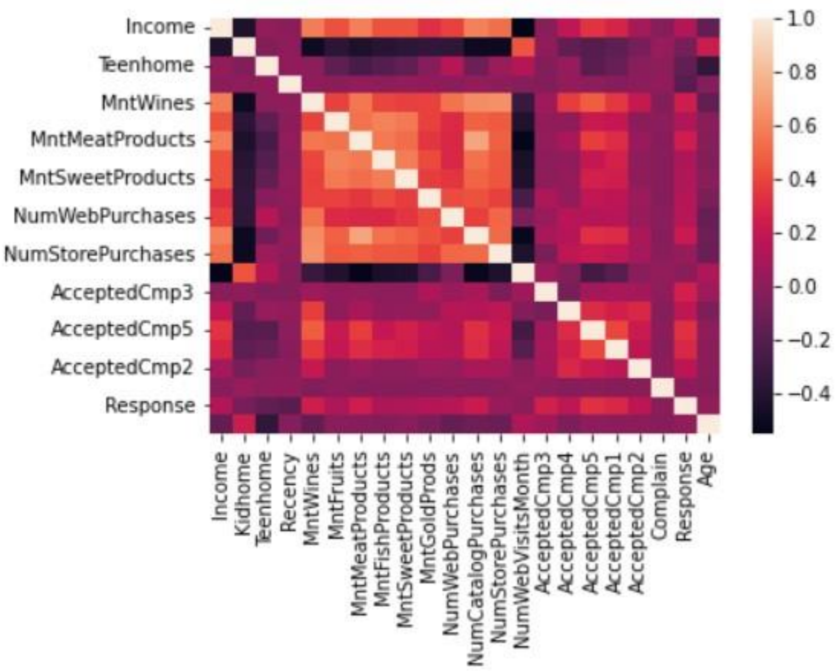


10. Построить heatmap для всех числовых колонок



```
columns = ['Income', 'Kidhome', 'Teenhome', 'Recency', 'MntWines',
                    'MntFruits', 'MntMeatProducts', 'MntFishProducts',
                   'MntSweetProducts', 'MntGoldProds', 'NumWebPurchases',
                   'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
                   'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
                   'AcceptedCmp2', 'Complain', 'Response', 'Age']
sns.heatmap(dataframe[columns].corr())
```

```
<AxesSubplot:>
```



Вывод: Изучил основы пандас и преисполнился в своем познании.