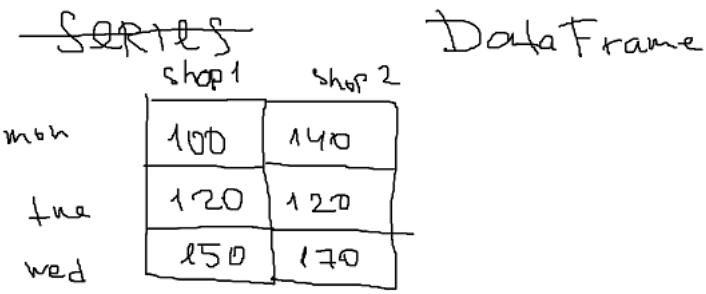
Зимова школа 2024

- df, series
- df to json|csv
- read csv, json
- GAS YEAR
- agg functions
- head tail info describe
- filtering
- value_counts num, %
- sort_values
- unique, nunique
- plot
- groupby
- dt (extract day), str
- sns countplot, kdeplot, boxplot
- optional: pivot_table, vectorization, slices

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
s1 = pd.Series([100, 120, 150], index=['mon', 'tue', 'wed'])
s1
```

```
mon    100
tue    120
wed    150
dtype: int64
```




```
df1 = pd.DataFrame(
    data={
    'shop1': [100, 120, 150],
    'shop2': [140, 120, 170],
    },
    index=['mon', 'tue', 'wed']
)
```

```
df1
```

	shop1	shop2
mon	100	140
tue	120	120
wed	150	170

```
df1.to_csv('df1.csv')
```

```
df1.to_json('df1.json', indent=3)
```

```
df11 = pd.read_csv('df1.csv', index_col=0)
df11
```

	shop1	shop2
mon	100	140
tue	120	120
wed	150	170

```
df12 = pd.read_json('df1.json')
df12
```

	shop1	shop2
mon	100	140
tue	120	120
wed	150	170

```
gas = pd.read_csv('gas_year.csv')
gas['date'] = pd.to_datetime(gas['date'])
gas['month'] = gas['date'].dt.month
gas.head()
```

	date	station	volume	mark	price	payment	month
0	2020-01-02 06:21:32	C	20.9	Premium	1.83	38.25	1
1	2020-01-02 08:42:45	A	16.4	Regular	1.76	28.86	1
2	2020-01-02 11:03:58	C	23.7	Premium	1.83	43.37	1
3	2020-01-02 13:25:11	C	12.2	Regular	1.76	21.47	1
4	2020-01-02 15:46:24	A	11.0	Premium	1.83	20.13	1

```
pd.to_datetime(gas['date'], format='%Y-%m-%d %H:%M:%S')
```

```
gas.tail(3)
```

	date	station	volume	mark	price	payment	month
3676	2020-12-27 18:14:00	C	19.0	Regular	1.87	35.53	12
3677	2020-12-27 20:35:13	A	18.3	Premium	1.94	35.50	12
3678	2020-12-27 22:56:26	C	17.6	Premium	1.94	34.14	12

```
gas.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3679 entries, 0 to 3678
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype         
---  ------   --------------  -----         
 0   date     3679 non-null   datetime64[ns]
 1   station  3679 non-null   object        
 2   volume   3679 non-null   float64       
 3   mark     3679 non-null   object        
 4   price    3679 non-null   float64       
 5   payment  3679 non-null   float64       
 6   month    3679 non-null   int64         
dtypes: datetime64[ns](1), float64(3), int64(1), object(2)
memory usage: 201.3+ KB
```

```
gas.describe()
```

	volume	price	payment	month
count	3679.000000	3679.000000	3679.000000	3679.000000
mean	22.519516	1.920690	43.281264	6.471052
std	11.111304	0.106962	21.557766	3.412651
min	5.000000	1.760000	8.800000	1.000000
25%	14.150000	1.830000	27.070000	4.000000
50%	21.800000	1.890000	41.810000	6.000000
75%	29.800000	2.030000	57.385000	9.000000
max	72.200000	2.110000	142.350000	12.000000

```
gas['date'].dt.day
```

```
gas[['station', 'payment']]
```

```
gas['station'].unique(), gas['station'].nunique()
```

```
(array(['C', 'A', 'B'], dtype=object), 3)
```

```
gas[(gas['station'] == 'C') & (gas['mark'] == 'Premium') & (gas['payment'] > 50)]
```

	date	station	volume	mark	price	payment	month
6	2020-01-02 20:28:50	C	27.6	Premium	1.83	50.51	1
42	2020-01-06 09:12:38	C	31.7	Premium	1.83	58.01	1
45	2020-01-06 16:16:17	C	30.5	Premium	1.83	55.82	1
117	2020-01-13 17:43:53	C	32.4	Premium	1.83	59.29	1
187	2020-01-20 14:29:03	C	32.1	Premium	1.83	58.74	1
...	...	...	...	...	...	...	...
3420	2020-12-02 15:42:32	C	30.2	Premium	1.94	58.59	12
3451	2020-12-05 16:40:15	C	27.4	Premium	1.94	53.16	12
3463	2020-12-06 20:54:51	C	33.1	Premium	1.94	64.21	12
3496	2020-12-10 02:35:00	C	28.4	Premium	1.94	55.10	12
3673	2020-12-27 11:10:21	C	32.0	Premium	1.94	62.08	12

81 rows × 7 columns

```
gas[(gas['station'] == 'C') & (gas['mark'] == 'Premium')][['station', 'payment', 'mark']]
```

	station	payment	mark
0	C	38.25	Premium
2	C	43.37	Premium
6	C	50.51	Premium
10	C	34.22	Premium
11	C	31.66	Premium
...	...	...	...
3661	C	9.70	Premium
3665	C	31.82	Premium
3673	C	62.08	Premium
3674	C	42.10	Premium
3678	C	34.14	Premium

372 rows × 3 columns

```
gas.sort_values(by='volume', ascending=False)[:3]
```

	date	station	volume	mark	price	payment
2448	2020-08-29 07:59:56	B	72.2	Premium	1.90	137.18
2313	2020-08-16 02:15:41	B	71.3	Regular	1.83	130.48
2097	2020-07-25 21:52:53	B	69.1	Super	2.06	142.35

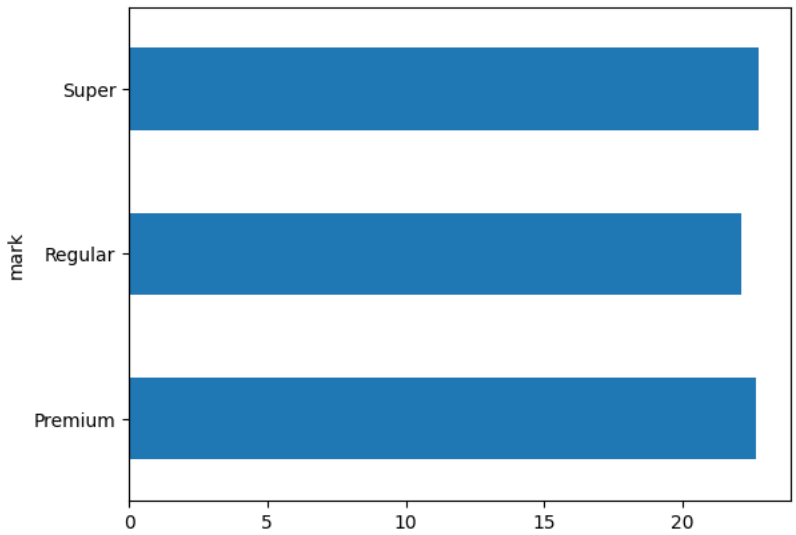
```
# gas['station'].value_counts()
gas['mark'].value_counts()
```

```
Super      1281
Regular    1220
Premium    1178
Name: mark, dtype: int64
```

```
# gas.groupby('station')['payment'].sum()
gas.groupby('mark')['volume'].mean()
```

```
mark
Premium    22.652462
Regular    22.133525
Super      22.764871
Name: volume, dtype: float64
```

```
gas.groupby('mark')['volume'].mean().plot(kind='barh');
```



```
gas.groupby(['station', 'mark'])['volume'].mean()
```

```
station  mark
A        Premium    21.941136
         Regular    21.309774
         Super      21.564220
B        Premium    27.143716
         Regular    26.000952
         Super      26.812442
C        Premium    19.075000
         Regular    18.902494
         Super      19.764477
Name: volume, dtype: float64
```

```
gas.groupby('mark')['volume'].agg(['sum', 'mean', 'min', 'max'])
```

mark	sum	mean	min	max
Premium	26684.6	22.652462	5.0	72.2
Regular	27002.9	22.133525	5.0	71.3
Super	29161.8	22.764871	5.0	69.1

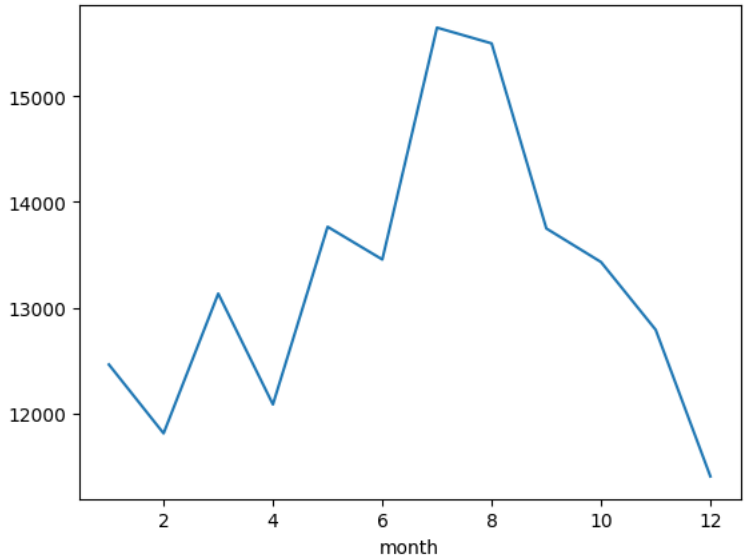
```
gas.pivot_table(index='station', columns='mark', values='payment', aggfunc='sum', margins=True)
```

station	mark Premium	Regular	Super	All
A	18202.71	15444.86	19309.55	52957.12
B	18741.59	19812.24	23892.58	62446.41
C	13391.02	13749.02	16688.20	43828.24
All	50335.32	49006.12	59890.33	159231.77

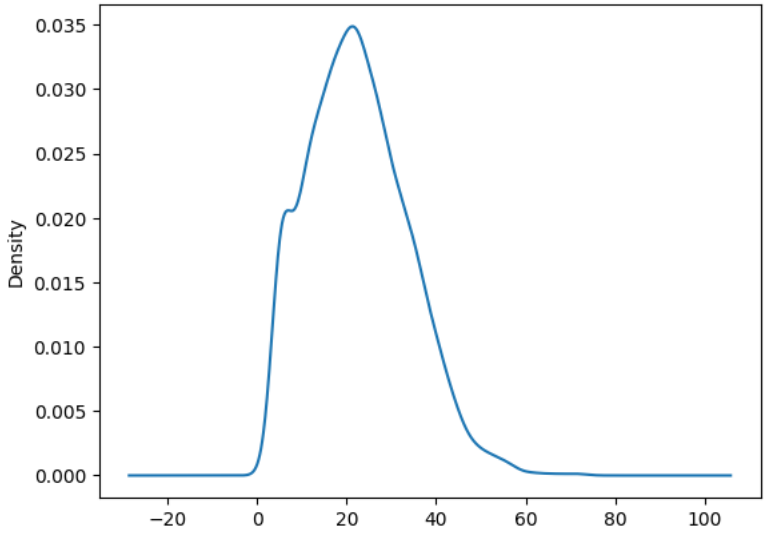
```
gas.groupby(gas['date'].dt.hour)['payment'].sum()
```

```
date
0     6352.97
1     6430.51
2     6931.07
3     6661.25
4     6582.38
5     6531.03
6     6225.15
7     6972.69
8     6560.64
9     6617.85
10    6662.77
11    6707.62
12    6774.65
13    6674.24
14    6789.02
15    6826.09
16    6696.46
17    7136.06
18    6333.85
19    6314.64
20    6560.96
21    6324.14
22    6851.06
23    6714.67
Name: payment, dtype: float64
```

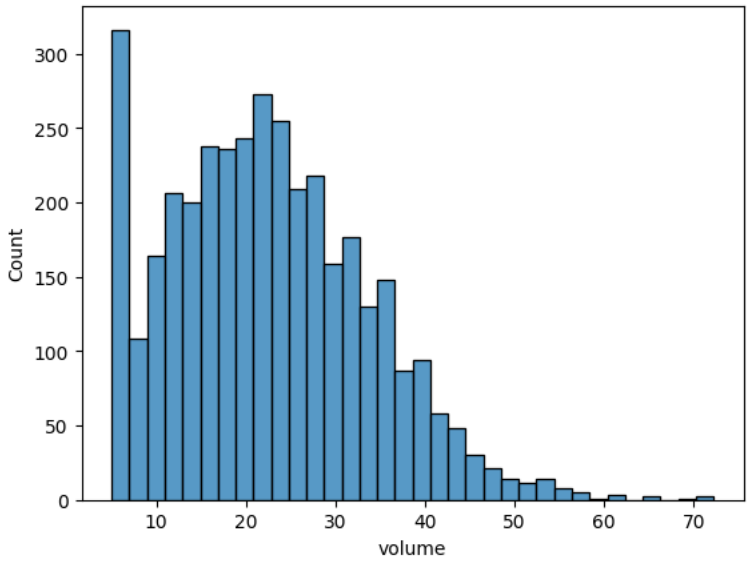
```
gas.groupby('month')['payment'].sum().plot();
```



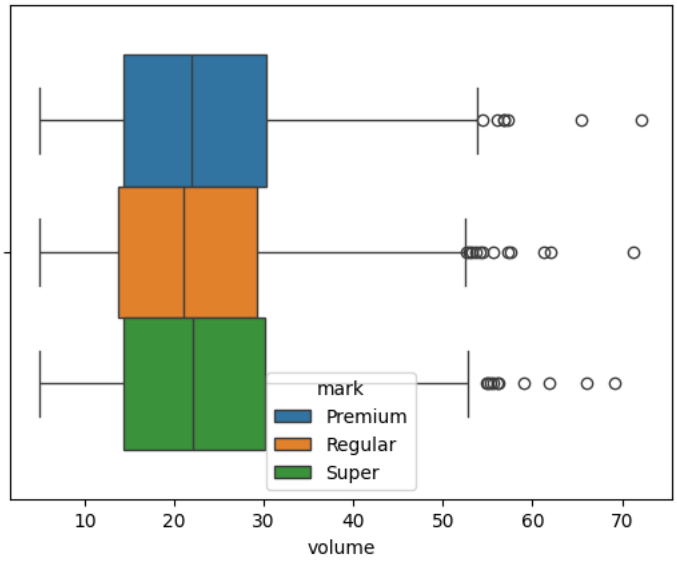

```
gas['volume'].plot(kind='kde');
```




```
sns.histplot(data=gas, x='volume');
```



```
sns.boxplot(data=gas, x='volume', hue='mark');
```




```
sns.countplot(data=gas, x='station', hue='mark');
```