In [1]:

```
import pandas as pd
```

In [2]:

```
imoveis = pd.DataFrame([['Apartamento', None, 970, 68],
                        ['Apartamento', 2000, 878, 112],
                        ['Casa', 5000, None, 500],
                        ['Apartamento', None, 1010, 170],
                        ['Apartamento', 1500, 850, None],
                        ['Casa', None, None, None],
                        ['Apartamento', 2000, 878, None],
                        ['Apartamento', 1550, None, 228],
                        ['Apartamento', 2500, 880, 195]],
                        columns = ['Tipo', 'Valor', 'Condominio', 'IPTU'])
```

In [3]:

```
imoveis
```

Out[3]:

	Tipo	Valor	Condominio	IPTU
0	Apartamento	NaN	970.0	68.0
1	Apartamento	2000.0	878.0	112.0
2	Casa	5000.0	NaN	500.0
3	Apartamento	NaN	1010.0	170.0
4	Apartamento	1500.0	850.0	NaN
5	Casa	NaN	NaN	NaN
6	Apartamento	2000.0	878.0	NaN
7	Apartamento	1550.0	NaN	228.0
8	Apartamento	2500.0	880.0	195.0

In [4]:

```
imoveis.shape[0]
```

Out[4]:

```
9
```

In [5]:

```
imoveis.dropna(subset = ['Valor'], inplace = True)
imoveis
```

Out[5]:

	Tipo	Valor	Condominio	IPTU
1	Apartamento	2000.0	878.0	112.0
2	Casa	5000.0	NaN	500.0
4	Apartamento	1500.0	850.0	NaN
6	Apartamento	2000.0	878.0	NaN
7	Apartamento	1550.0	NaN	228.0
8	Apartamento	2500.0	880.0	195.0

In [6]:

```
selecao = (imoveis['Tipo'] == 'Apartamento') & (imoveis['Condominio'].isnull())
print(f"valores boleanos com condição de seleção: \n\n{selecao}")
print()
print(f"inversão dos valores boleanos: \n\n{~selecao}")
```

```
valores boleanos com condição de seleção:

1    False
2    False
4    False
6    False
7     True
8    False
dtype: bool

inversão dos valores boleanos:

1     True
2     True
4     True
6     True
7    False
8     True
dtype: bool
```

o caractere ~ inverte o valor boelano onde, oque oque era falso vira verdadeiro, e oque era verdadeiro vira falso.

In [7]:

```
imoveis
```

Out[7]:

	Tipo	Valor	Condominio	IPTU
1	Apartamento	2000.0	878.0	112.0
2	Casa	5000.0	NaN	500.0
4	Apartamento	1500.0	850.0	NaN
6	Apartamento	2000.0	878.0	NaN
7	Apartamento	1550.0	NaN	228.0
8	Apartamento	2500.0	880.0	195.0

In [8]:

```
imoveis2 = imoveis[~selecao]
imoveis2
```

Out[8]:

	Tipo	Valor	Condominio	IPTU
1	Apartamento	2000.0	878.0	112.0
2	Casa	5000.0	NaN	500.0
4	Apartamento	1500.0	850.0	NaN
6	Apartamento	2000.0	878.0	NaN
8	Apartamento	2500.0	880.0	195.0

substitui os valores na variavel pelos valores novos da seleção invertida.

In [9]:

```
imoveis = imoveis2
imoveis
```

Out[9]:

	Tipo	Valor	Condominio	IPTU
1	Apartamento	2000.0	878.0	112.0
2	Casa	5000.0	NaN	500.0
4	Apartamento	1500.0	850.0	NaN
6	Apartamento	2000.0	878.0	NaN
8	Apartamento	2500.0	880.0	195.0

In [10]:

```
imoveis = imoveis.fillna({'Condominio': 0, 'IPTU': 0})
imoveis
```

Out[10]:

	Tipo	Valor	Condominio	IPTU
1	Apartamento	2000.0	878.0	112.0
2	Casa	5000.0	0.0	500.0
4	Apartamento	1500.0	850.0	0.0
6	Apartamento	2000.0	878.0	0.0
8	Apartamento	2500.0	880.0	195.0

In [11]:

```
imoveis.index = range(imoveis.shape[0])
imoveis
```

Out[11]:

	Tipo	Valor	Condominio	IPTU
0	Apartamento	2000.0	878.0	112.0
1	Casa	5000.0	0.0	500.0
2	Apartamento	1500.0	850.0	0.0
3	Apartamento	2000.0	878.0	0.0
4	Apartamento	2500.0	880.0	195.0

In [12]:

```
atletas = pd.DataFrame([['Marcos', 9.62], ['Pedro', None], ['João', 9.69],
                        ['Beto', 9.72], ['Sandro', None], ['Denis', 9.69],
                        ['Ary', None], ['Carlos', 9.74]],
                        columns = ['Corredor', 'Melhor Tempo'])
atletas
```

Out[12]:

	Corredor	Melhor Tempo
0	Marcos	9.62
1	Pedro	NaN
2	João	9.69
3	Beto	9.72
4	Sandro	NaN
5	Denis	9.69
6	Ary	NaN
7	Carlos	9.74

In [13]:

```
atletas.fillna(atletas.mean(), inplace = True)
```

In [14]:

```
atletas
```

Out[14]:

	Corredor	Melhor Tempo
0	Marcos	9.620
1	Pedro	9.692
2	João	9.690
3	Beto	9.720
4	Sandro	9.692
5	Denis	9.690
6	Ary	9.692
7	Carlos	9.740

In [15]:

```
alunos = pd.DataFrame({'Nome': ['Ary', 'Cátia', 'Denis', 'Beto', 'Bruna', 'Dara', 'Carlos',
                        'Sexo': ['M', 'F', 'M', 'M', 'F', 'F', 'M', 'F'],
                        'Idade': [15, 27, 56, 32, 42, 21, 19, 35],
                        'Notas': [7.5, 2.5, 5.0, 10, 8.2, 7, 6, 5.6]},
                        columns = ['Nome', 'Idade', 'Sexo', 'Notas'])
```

In [16]:

```
alunos
```

Out[16]:

	Nome	Idade	Sexo	Notas
0	Ary	15	M	7.5
1	Cátia	27	F	2.5
2	Denis	56	M	5.0
3	Beto	32	M	10.0
4	Bruna	42	F	8.2
5	Dara	21	F	7.0
6	Carlos	19	M	6.0
7	Alice	35	F	5.6

In [17]:

```
alunos['Notas-Média(Notas)'] = alunos['Notas'].mean()
alunos
```

Out[17]:

	Nome	Idade	Sexo	Notas	Notas-Média(Notas)
0	Ary	15	M	7.5	6.475
1	Cátia	27	F	2.5	6.475
2	Denis	56	M	5.0	6.475
3	Beto	32	M	10.0	6.475
4	Bruna	42	F	8.2	6.475
5	Dara	21	F	7.0	6.475
6	Carlos	19	M	6.0	6.475
7	Alice	35	F	5.6	6.475

In [18]:

```
alunos['Notas-Média(Notas)'] = alunos['Notas'].apply(lambda x: x - alunos['Notas'].mean())
```

In [19]:

```
alunos
```

Out[19]:

	Nome	Idade	Sexo	Notas	Notas-Média(Notas)
0	Ary	15	M	7.5	1.025
1	Cátia	27	F	2.5	-3.975
2	Denis	56	M	5.0	-1.475
3	Beto	32	M	10.0	3.525
4	Bruna	42	F	8.2	1.725
5	Dara	21	F	7.0	0.525
6	Carlos	19	M	6.0	-0.475
7	Alice	35	F	5.6	-0.875

In [20]:

```
alunos['Faixa Etária'] = alunos['Idade'].apply(lambda x: 'Menor que 20 anos' if x < 20
        else ('Entre 20 e 40 anos' if (x >= 20 and x <= 40)
        else 'Maior que 40 anos'))
```

In [21]:

```
alunos
```

Out[21]:

	Nome	Idade	Sexo	Notas	Notas-Média(Notas)	Faixa Etária
0	Ary	15	M	7.5	1.025	Menor que 20 anos
1	Cátia	27	F	2.5	-3.975	Entre 20 e 40 anos
2	Denis	56	M	5.0	-1.475	Maior que 40 anos
3	Beto	32	M	10.0	3.525	Entre 20 e 40 anos
4	Bruna	42	F	8.2	1.725	Maior que 40 anos
5	Dara	21	F	7.0	0.525	Entre 20 e 40 anos
6	Carlos	19	M	6.0	-0.475	Menor que 20 anos
7	Alice	35	F	5.6	-0.875	Entre 20 e 40 anos

In [22]:

```
alunos['Notas-Média(Notas)'] = alunos.Notas - alunos.Notas.mean()
```

In [23]:

```
alunos
```

Out[23]:

	Nome	Idade	Sexo	Notas	Notas-Média(Notas)	Faixa Etária
0	Ary	15	M	7.5	1.025	Menor que 20 anos
1	Cátia	27	F	2.5	-3.975	Entre 20 e 40 anos
2	Denis	56	M	5.0	-1.475	Maior que 40 anos
3	Beto	32	M	10.0	3.525	Entre 20 e 40 anos
4	Bruna	42	F	8.2	1.725	Maior que 40 anos
5	Dara	21	F	7.0	0.525	Entre 20 e 40 anos
6	Carlos	19	M	6.0	-0.475	Menor que 20 anos
7	Alice	35	F	5.6	-0.875	Entre 20 e 40 anos

Erro: Alternativa correta! O único erro nesta linha de código está na declaração elif, trocando por else o código roda perfeitamente.

In [24]:

```
alunos['Faixa Etária'] = alunos['Idade']
    .apply(lambda x: 'Menor que 20 anos' if x < 20
        elif ('Entre 20 e 40 anos' if (x >= 20 and x <= 40)
            else 'Maior que 40 anos'))
```

```
  File "<ipython-input-24-b1663ea618f4>", line 2
    .apply(lambda x: 'Menor que 20 anos' if x < 20
    ^
IndentationError: unexpected indent
```

In [27]:

```
m1 = 'CCcCCccCCCccCcCccCcCcCCCcCCccccCCcCCcCcCccccCCcCcccCc'
m2 = 'CCCCCccCccCcCCCCccCccccCccCccCCcCccCcCcCCcCccCccCc'
m3 = 'CccCCccCcCCCCCCCCCCccccCccCCCCCCccCCCccccCCCcCCccCC'
m4 = 'cCCccCCccCCccCCccccCcCcCcCcCcCcCCCCccccCCCcCCcCCCC'
m5 = 'CCCcCcCcCcCCCcCcCcCcCCccCcCCcccCccCCcCcCcCcCcccccCc'
```

In [33]:

```
eventos = { 'm1': list(m1),
            'm2': list(m2),
            'm3': list(m3),
            'm4': list(m4),
            'm5': list(m5)}
moedas = pd.DataFrame(eventos)
print(moedas)
df = pd.DataFrame(data = ['Coroa', 'Cara'],
                  index = ['C', 'c'],
                  columns = ['Faces'])
print(df)
for item in moedas:
    df = pd.concat([df, moedas[item].value_counts()], axis = 1)
df
```

```
   m1 m2 m3 m4 m5
0   C  C  C  c  C
1   C  C  c  C  C
2   c  C  c  C  C
3   C  C  C  c  c
4   C  C  C  c  C
5   c  c  c  C  c
6   c  c  c  C  C
7   C  C  C  c  c
8   C  c  c  c  C
9   C  c  C  C  c
10  c  C  C  C  C
11  c  c  C  c  C
12  C  C  C  c  C
13  c  C  C  C  c
14  C  C  C  C  C
15  c  C  C  c  C
16  c  c  C  c  c
17  C  c  C  c  C
18  c  C  C  c  c
19  C  c  c  C  C
20  c  c  c  c  C
21  C  c  c  C  c
22  C  c  C  c  c
23  C  C  c  C  C
24  c  c  c  c  c
25  C  c  C  C  C
26  C  C  C  c  C
27  c  c  C  C  c
28  c  c  C  c  c
29  c  C  C  C  c
30  C  C  C  c  C
31  C  c  c  C  c
32  c  C  c  C  c
33  C  c  C  C  C
34  c  c  C  C  C
35  C  C  C  c  c
36  c  c  c  c  C
37  C  C  c  c  c
38  c  c  c  c  C
39  c  C  C  C  c
40  c  C  C  C  C
```

```
41  C   c   C   C   c
42  C   C   c   c   C
43  c   c   C   C   c
44  C   c   C   C   c
45  c   C   c   c   c
46  c   c   c   C   c
47  c   c   c   C   c
48  C   C   C   C   C
49  c   c   C   C   c
   Faces
C  Coroa
```

Out[33]:

	Faces	m1	m2	m3	m4	m5
C	Coroa	25	24	31	27	25
c	Cara	25	26	19	23	25

In [25]:

```
alunos = pd.DataFrame({'Nome': ['Ary', 'Cátia', 'Denis', 'Beto', 'Bruna', 'Dara', 'Carlos',
                        'Sexo': ['M', 'F', 'M', 'M', 'F', 'F', 'M', 'F'],
                        'Idade': [15, 27, 56, 32, 42, 21, 19, 35],
                        'Notas': [7.5, 2.5, 5.0, 10, 8.2, 7, 6, 5.6],
                        'Aprovado': [True, False, False, True, True, True, False, False]},
                        columns = ['Nome', 'Idade', 'Sexo', 'Notas', 'Aprovado'])
```

In [26]:

```
alunos
```

Out[26]:

	Nome	Idade	Sexo	Notas	Aprovado
0	Ary	15	M	7.5	True
1	Cátia	27	F	2.5	False
2	Denis	56	M	5.0	False
3	Beto	32	M	10.0	True
4	Bruna	42	F	8.2	True
5	Dara	21	F	7.0	True
6	Carlos	19	M	6.0	False
7	Alice	35	F	5.6	False

In [27]:

```
sexo = alunos.groupby('Sexo')
sexo = pd.DataFrame(sexo['Notas'].mean().round(2))
sexo.columns = ['Notas Médias']
sexo
```

Out[27]:

Sexo	Notas Médias
F	5.82
M	7.12

In [25]:

```
precos = pd.DataFrame([['Feira', 'Cebola', 2.5],
                       ['Mercado', 'Cebola', 1.99],
                       ['Supermercado', 'Cebola', 1.69],
                       ['Feira', 'Tomate', 4],
                       ['Mercado', 'Tomate', 3.29],
                       ['Supermercado', 'Tomate', 2.99],
                       ['Feira', 'Batata', 4.2],
                       ['Mercado', 'Batata', 3.99],
                       ['Supermercado', 'Batata', 3.69]],
                       columns = ['Local', 'Produto', 'Preço'])
precos
```

Out[25]:

	Local	Produto	Preço
0	Feira	Cebola	2.50
1	Mercado	Cebola	1.99
2	Supermercado	Cebola	1.69
3	Feira	Tomate	4.00
4	Mercado	Tomate	3.29
5	Supermercado	Tomate	2.99
6	Feira	Batata	4.20
7	Mercado	Batata	3.99
8	Supermercado	Batata	3.69

In [28]:

```
produtos = precos.groupby('Produto')
produtos.describe().round(2)
```

Out[28]:

	Preço							
	count	mean	std	min	25%	50%	75%	max
Produto								
Batata	3.0	3.96	0.26	3.69	3.84	3.99	4.10	4.2
Cebola	3.0	2.06	0.41	1.69	1.84	1.99	2.24	2.5
Tomate	3.0	3.43	0.52	2.99	3.14	3.29	3.64	4.0

In [30]:

```
estatisticas = ['mean', 'std', 'min', 'max']
nomes = {'mean': 'Média', 'std': 'Desvio Padrão', 'min': 'Mínimo', 'max': 'Máximo'}
produtos['Preço'].aggregate(estatisticas).rename(columns = nomes).round(2)
```

Out[30]:

	Média	Desvio Padrão	Mínimo	Máximo
Produto				
Batata	3.96	0.26	3.69	4.2
Cebola	2.06	0.41	1.69	2.5
Tomate	3.43	0.52	2.99	4.0

In [ ]:

```

```