```
import numpy as np
import pandas as pd
```

```
df = pd.read_csv('/content/laptop_data.csv')
```

```
df.head(100)
```

	Unnamed: 0	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price
0	0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg	71378.6832
1	1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg	47895.5232
2	2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg	30636.0000
3	3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455	macOS	1.83kg	135195.3360
4	4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37kg	96095.8080
...	...	...	...	...	...	...	...	...	...	...	...	...
95	95	Acer	2 in 1 Convertible	13.3	IPS Panel Full HD / Touchscreen 1920x1080	Intel Core i5 8250U 1.6GHz	8GB	256GB SSD	Intel UHD Graphics 620	Windows 10	1.5kg	45128.1600
96	96	Dell	Notebook	15.6	Full HD 1920x1080	Intel Core i7 7500U	8GB	1TB HDD	AMD Radeon R5 M430	Linux	2.2kg	31962.6720

Next steps: Generate code with df New interactive sheet

```
df.shape
```

```
(1303, 12)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Unnamed: 0        1303 non-null   int64  
 1   Company           1303 non-null   object 
 2   TypeName          1303 non-null   object 
 3   Inches            1303 non-null   float64
 4   ScreenResolution  1303 non-null   object 
 5   Cpu               1303 non-null   object 
 6   Ram               1303 non-null   object 
 7   Memory            1303 non-null   object 
 8   Gpu               1303 non-null   object 
 9   OpSys             1303 non-null   object 
 10  Weight            1303 non-null   object 
 11  Price             1303 non-null   float64
dtypes: float64(2), int64(1), object(9)
memory usage: 122.3+ KB
```

```
df.duplicated().sum()
```

```
np.int64(0)
```

```
df.isnull().sum()
```

	0
Unnamed: 0	0
Company	0
TypeName	0
Inches	0
ScreenResolution	0
Cpu	0
Ram	0
Memory	0
Gpu	0
OpSys	0
Weight	0
Price	0

dtype: int64

```
df.drop(columns=['Unnamed: 0'],inplace=True)
```

```
df.head()
```

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg	71378.6832
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg	47895.5232
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg	30636.0000
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455	macOS	1.83kg	135195.3360
4	Apple	Ultrabook	13.3	IPS Panel Retina Display	Intel Core i5	8GB	256GB SSD	Intel Iris Plus	macOS	1.37kg	96095.8080

Next steps: Generate code with df New interactive sheet

```
df['Ram'] = df['Ram'].str.replace('GB','')
df['Weight'] = df['Weight'].str.replace('kg','')
```

```
df.head()
```

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	135195.3360
4	Apple	Ultrabook	13.3	IPS Panel Retina Display	Intel Core i5	8	256GB SSD	Intel Iris Plus	macOS	1.37	96095.8080

Next steps: Generate code with df New interactive sheet

```
df['Ram'] = df['Ram'].astype('int32')
```

```
df['Weight'] = df['Weight'].astype('float32')
```

```
df.info()
```

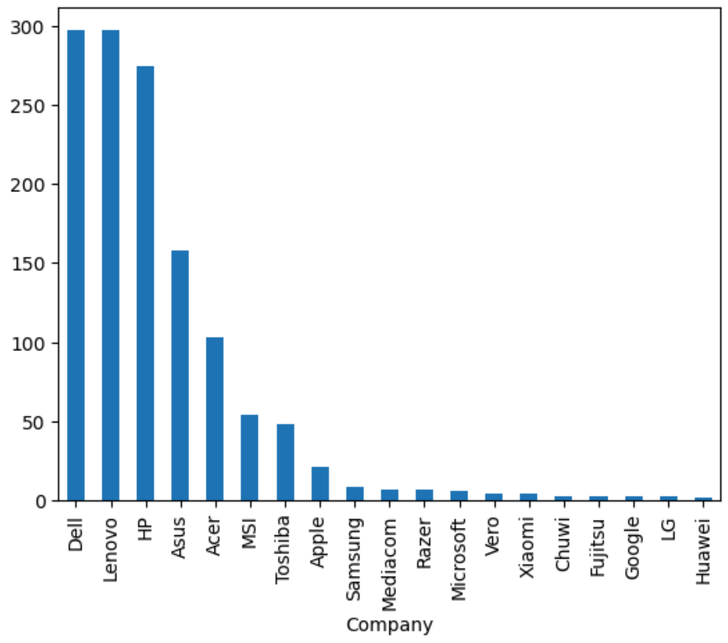
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1303 non-null   object 
 1   TypeName          1303 non-null   object 
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object 
 4   Cpu               1303 non-null   object 
 5   Ram               1303 non-null   int32  
 6   Memory            1303 non-null   object 
 7   Gpu               1303 non-null   object 
 8   OpSys             1303 non-null   object 
 9   Weight            1303 non-null   float32
 10  Price             1303 non-null   float64
dtypes: float32(1), float64(2), int32(1), object(7)
memory usage: 101.9+ KB
```

```
import seaborn as sns
import matplotlib.pyplot as plt
```

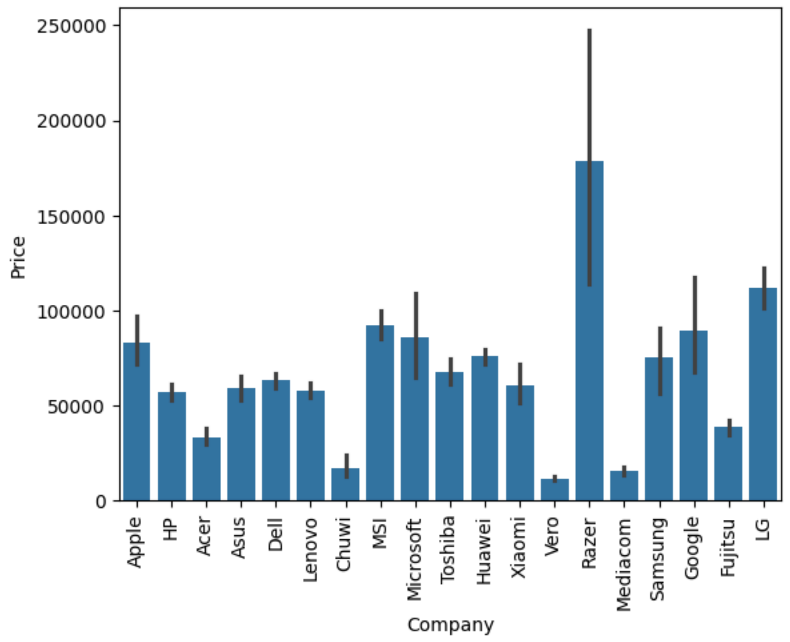
```
sns.distplot(df['Price'])
```

```
df['Company'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Company'>
```

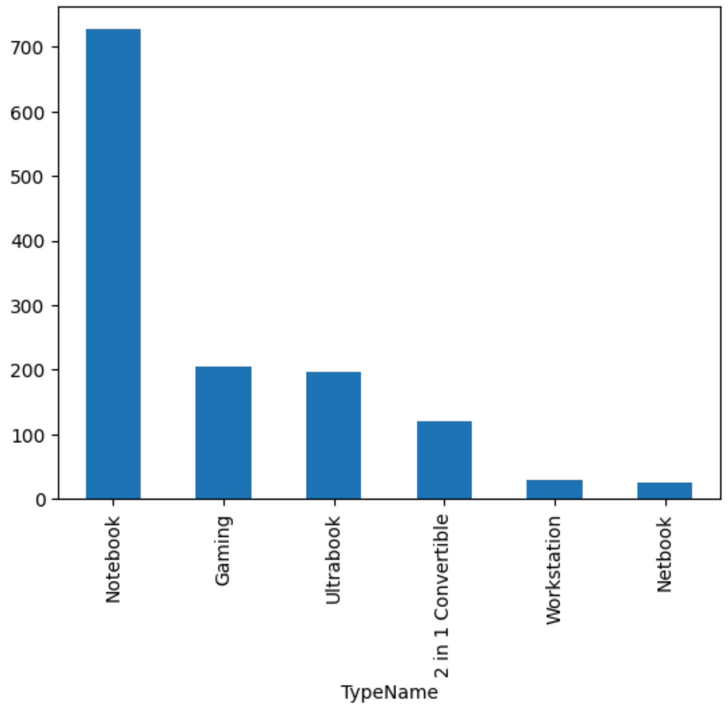


```
sns.barplot(x=df['Company'],y=df['Price'])
plt.xticks(rotation='vertical')
plt.show()
```

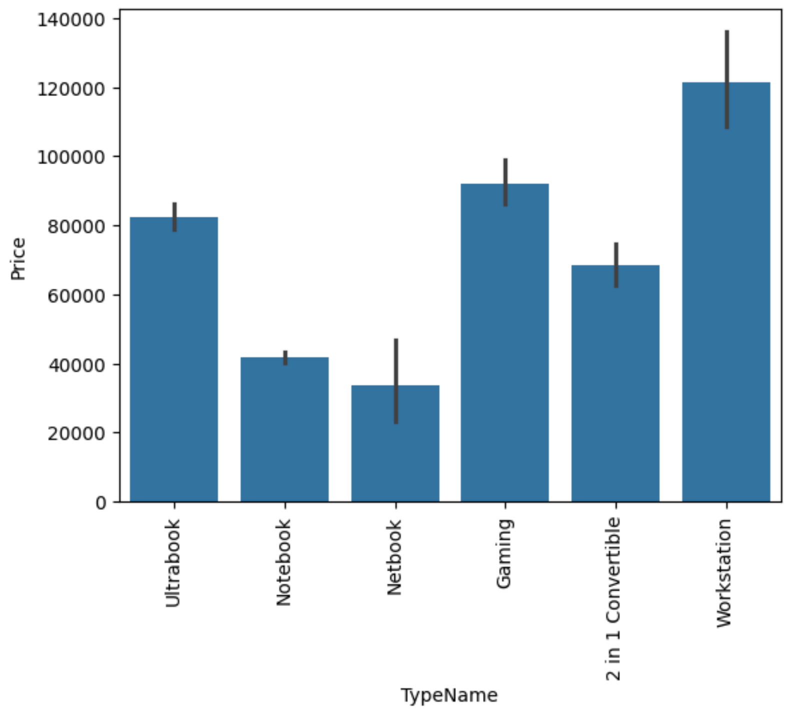


```
df['TypeName'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='TypeName'>
```

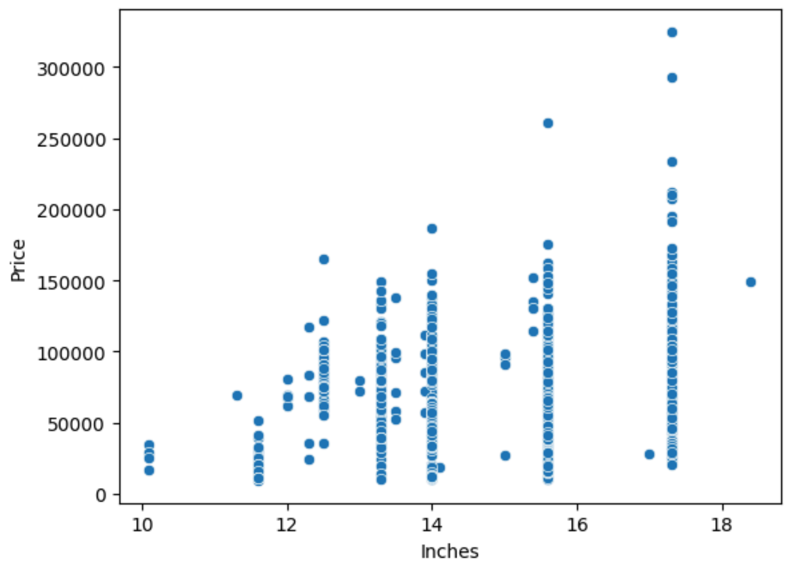


```
sns.barplot(x=df['TypeName'],y=df['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



```
sns.scatterplot(x=df['Inches'],y=df['Price'])
```

```
<Axes: xlabel='Inches', ylabel='Price'>
```



```
df['ScreenResolution'].value_counts()
```

ScreenResolution	count
Full HD 1920x1080	507
1366x768	281
IPS Panel Full HD 1920x1080	230
IPS Panel Full HD / Touchscreen 1920x1080	53
Full HD / Touchscreen 1920x1080	47
1600x900	23
Touchscreen 1366x768	16
Quad HD+ / Touchscreen 3200x1800	15
IPS Panel 4K Ultra HD 3840x2160	12
IPS Panel 4K Ultra HD / Touchscreen 3840x2160	11
4K Ultra HD / Touchscreen 3840x2160	10
IPS Panel 1366x768	7
Touchscreen 2560x1440	7
4K Ultra HD 3840x2160	7
IPS Panel Retina Display 2304x1440	6
IPS Panel Retina Display 2560x1600	6
Touchscreen 2256x1504	6
IPS Panel Quad HD+ / Touchscreen 3200x1800	6
IPS Panel Touchscreen 2560x1440	5
IPS Panel Retina Display 2880x1800	4
1440x900	4
IPS Panel Touchscreen 1920x1200	4
IPS Panel 2560x1440	4
IPS Panel Quad HD+ 2560x1440	3
IPS Panel Touchscreen 1366x768	3
Quad HD+ 3200x1800	3
1920x1080	3
2560x1440	3
Touchscreen 2400x1600	3
IPS Panel Quad HD+ 3200x1800	2
IPS Panel Full HD 2160x1440	2
IPS Panel Full HD 1366x768	1
IPS Panel Retina Display 2736x1824	1
Touchscreen / Full HD 1920x1080	1
Touchscreen / 4K Ultra HD 3840x2160	1

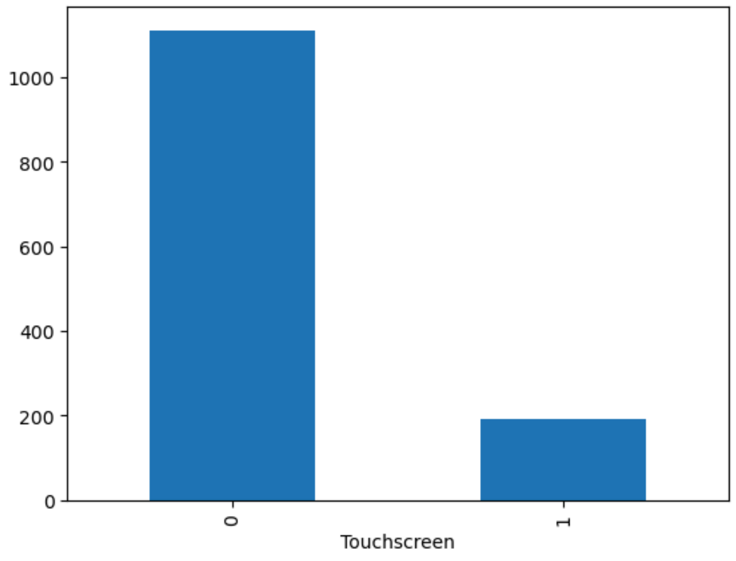
```
df['Touchscreen'] = df['ScreenResolution'].apply(lambda x:1 if 'Touchscreen' in x else 0)
```

```
df.sample(5)
```

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen
1021	Toshiba	Ultrabook	13.3	Full HD 1920x1080	Intel Core i5 6200U 2.3GHz	8	256GB SSD	Intel HD Graphics 520	Windows 10	1.20	84715.200	0
7	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	256GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	61735.536	0
261	Lenovo	Notebook	15.6	Full HD 1920x1080	Intel Core i3 6006U 2GHz	4	256GB SSD	Intel HD Graphics 520	No OS	2.20	23656.320	0
1029	HP	Notebook	17.3	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Nvidia GeForce 930MX	Windows 10	2.63	57542.400	0

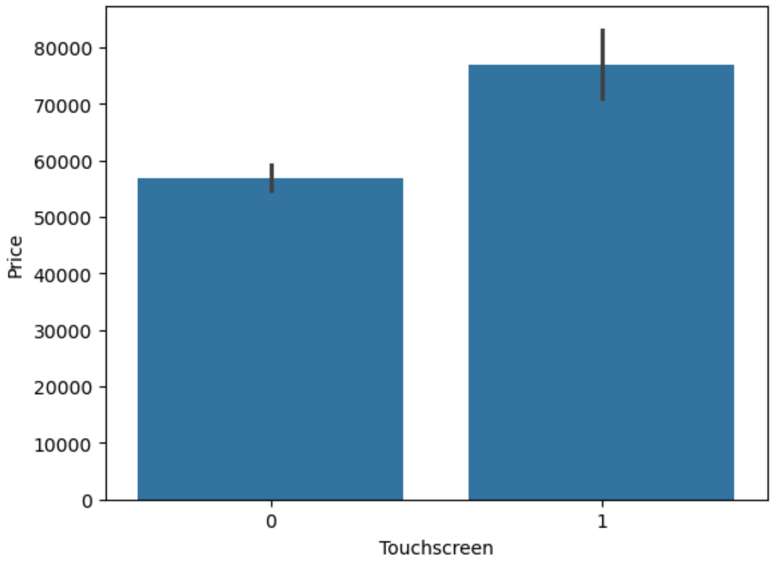
```
df['Touchscreen'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Touchscreen'>
```



```
sns.barplot(x=df['Touchscreen'],y=df['Price'])
```

```
<Axes: xlabel='Touchscreen', ylabel='Price'>
```



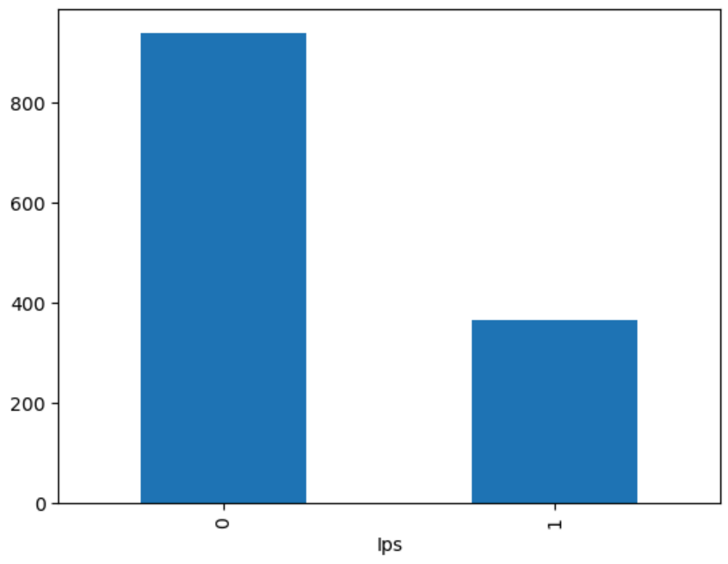
```
df['Ips'] = df['ScreenResolution'].apply(lambda x:1 if 'IPS' in x else 0)
```

```
df.head()
```

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0
3	Apple	Ultrabook	15.4	IPS Panel Retina	Intel Core	16	512GB	AMD Radeon	macOS	1.83	135195.3360	0	1

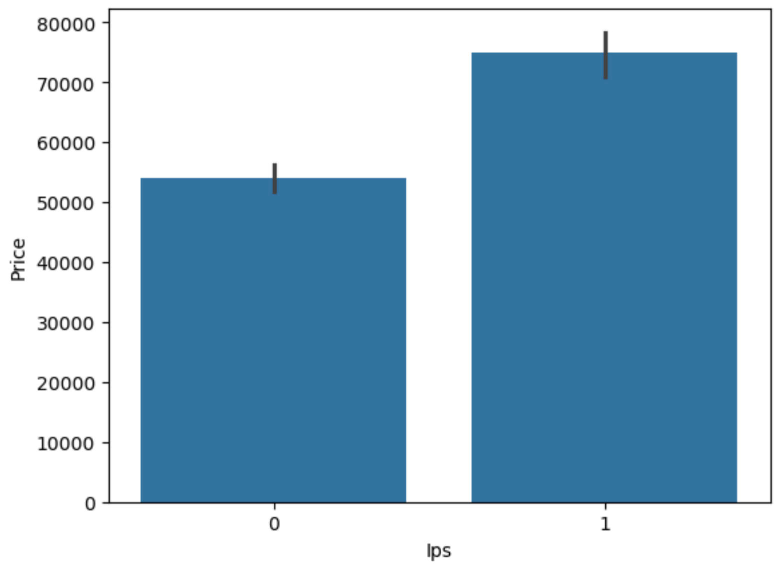
```
df['Ips'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Ips'>
```



```
sns.barplot(x=df['Ips'],y=df['Price'])
```

```
<Axes: xlabel='Ips', ylabel='Price'>
```



```
new = df['ScreenResolution'].str.split('x',n=1,expand=True)
```

```
df['X_res'] = new[0]
df['Y_res'] = new[1]
```

```
df.sample(5)
```

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips
1207	Lenovo	Notebook	15.6	1366x768	AMD E-Series 9000 2.2GHz	4	500GB HDD	AMD Radeon R2 Graphics	Windows 10	2.20	15930.7200	0	0
1070	Dell	Notebook	15.6	1366x768	Intel Core i3 6100U 2.3GHz	4	500GB HDD	Intel HD Graphics 520	Windows 10	2.06	25679.8944	0	0
853	Lenovo	Ultrabook	14.0	IPS Panel Full HD 1920x1080	Intel Core i7 7700HQ 2.8GHz	8	256GB SSD	Nvidia GeForce GT 940MX	Windows 10	1.96	101391.8400	0	1
731	Dell	Notebook	15.6	1366x768	Intel Core i5 7200U 2.5GHz	12	1TB HDD	Intel HD Graphics 620	Windows 10	2.25	34578.7200	0	0

```
df['X_res'] = df['X_res'].str.replace(',','').str.findall(r'(\d+\.?\d+)').apply(lambda x:x[0])
```

```
df.head()
```

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips	X_res
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	2560
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	1440
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	1920

```
df['X_res'] = df['X_res'].astype('int')
df['Y_res'] = df['Y_res'].astype('int')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1303 non-null   object 
 1   TypeName          1303 non-null   object 
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object 
 4   Cpu               1303 non-null   object 
 5   Ram               1303 non-null   int32  
 6   Memory            1303 non-null   object 
 7   Gpu               1303 non-null   object 
 8   OpSys             1303 non-null   object 
 9   Weight            1303 non-null   float32
 10  Price             1303 non-null   float64
 11  Touchscreen       1303 non-null   int64  
 12  Ips               1303 non-null   int64  
 13  X_res             1303 non-null   int64  
 14  Y_res             1303 non-null   int64  
dtypes: float32(1), float64(2), int32(1), int64(4), object(7)
memory usage: 142.6+ KB
```

```
df.corr()['Price']
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[41], line 1
----> 1 df.corr()['Price']

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:11049, in DataFrame.corr(self, method, min_periods,
numeric_only)
  11047 cols = data.columns
  11048 idx = cols.copy()
> 11049 mat = data.to_numpy(dtype=float, na_value=np.nan, copy=False)
  11051 if method == "pearson":
  11052     correl = libalgos.nancorr(mat, minp=min_periods)

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:1993, in DataFrame.to_numpy(self, dtype, copy, na_value)
   1991 if dtype is not None:
   1992     dtype = np.dtype(dtype)
-> 1993 result = self._mgr.as_array(dtype=dtype, copy=copy, na_value=na_value)
   1994 if result.dtype is not dtype:
   1995     result = np.asarray(result, dtype=dtype)

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:1694, in BlockManager.as_array(self, dtype,
copy, na_value)
   1692         arr.flags.writeable = False
   1693 else:
-> 1694     arr = self._interleave(dtype=dtype, na_value=na_value)
   1695     # The underlying data was copied within _interleave, so no need
   1696     # to further copy if copy=True or setting na_value
   1698 if na_value is lib.no_default:

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:1753, in BlockManager._interleave(self, dtype,
na_value)
   1751     else:
   1752         arr = blk.get_values(dtype)
-> 1753     result[rl.indexer] = arr
   1754     itemmask[rl.indexer] = 1
   1756 if not itemmask.all():
```

```
df['ppi'] = (((df['X_res']**2) + (df['Y_res']**2))**0.5/df['Inches']).astype('float')
```

```
df.corr()['Price']
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[43], line 1
----> 1 df.corr()['Price']

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:11049, in DataFrame.corr(self, method, min_periods,
numeric_only)
  11047 cols = data.columns
  11048 idx = cols.copy()
> 11049 mat = data.to_numpy(dtype=float, na_value=np.nan, copy=False)
  11051 if method == "pearson":
  11052     correl = libalgos.nancorr(mat, minp=min_periods)

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:1993, in DataFrame.to_numpy(self, dtype, copy, na_value)
   1991 if dtype is not None:
   1992     dtype = np.dtype(dtype)
-> 1993 result = self._mgr.as_array(dtype=dtype, copy=copy, na_value=na_value)
   1994 if result.dtype is not dtype:
   1995     result = np.asarray(result, dtype=dtype)

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:1694, in BlockManager.as_array(self, dtype,
copy, na_value)
   1692         arr.flags.writeable = False
   1693 else:
-> 1694     arr = self._interleave(dtype=dtype, na_value=na_value)
   1695     # The underlying data was copied within _interleave, so no need
   1696     # to further copy if copy=True or setting na_value
   1698 if na_value is lib.no_default:

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:1753, in BlockManager._interleave(self, dtype,
na_value)
   1751     else:
   1752         arr = blk.get_values(dtype)
-> 1753     result[rl.indexer] = arr
   1754     itemmask[rl.indexer] = 1
   1756 if not itemmask.all():
```

```
df.drop(columns=['ScreenResolution'],inplace=True)
df.head()
```

	Company	TypeName	Inches	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips	X_res	Y_res	ppi
0	Apple	Ultrabook	13.3	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	2560	1600	226.983005
1	Apple	Ultrabook	13.3	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	1440	900	127.677940
2	HP	Notebook	15.6	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	1920	1080	141.211998

```
df.drop(columns=['Inches','X_res','Y_res'],inplace=True)
```

```
df.head()
```

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips	ppi
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	226.983005
1	Apple	Ultrabook	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	127.677940
2	HP	Notebook	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	141.211998
			Intel Core i7			AMD Radeon Pro						

```
df['Cpu'].value_counts()
```

```
Cpu
Intel Core i5 7200U 2.5GHz       190
Intel Core i7 7700HQ 2.8GHz      146
Intel Core i7 7500U 2.7GHz       134
Intel Core i7 8550U 1.8GHz        73
Intel Core i5 8250U 1.6GHz        72
                                ... 
Intel Core i5 7200U 2.70GHz        1
Intel Core M M7-6Y75 1.2GHz        1
Intel Core M 6Y54 1.1GHz           1
AMD E-Series 9000 2.2GHz           1
Samsung Cortex A72&A53 2.0GHz      1
Name: count, Length: 118, dtype: int64
```

```
df['Cpu Name'] = df['Cpu'].apply(lambda x:" ".join(x.split()[0:3]))
```

```
df.head()
```

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips	ppi	Cpu Name
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	226.983005	Intel Core i5
1	Apple	Ultrabook	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	127.677940	Intel Core i5
2	HP	Notebook	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	141.211998	Intel Core i5
3	Apple	Ultrabook	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	135195.3360	0	1	220.534624	Intel Core i7

```
def fetch_processor(text):
    if text == 'Intel Core i7' or text == 'Intel Core i5' or text == 'Intel Core i3':
        return text
    else:
        if text.split()[0] == 'Intel':
            return 'Other Intel Processor'
        else:
            return 'AMD Processor'
```

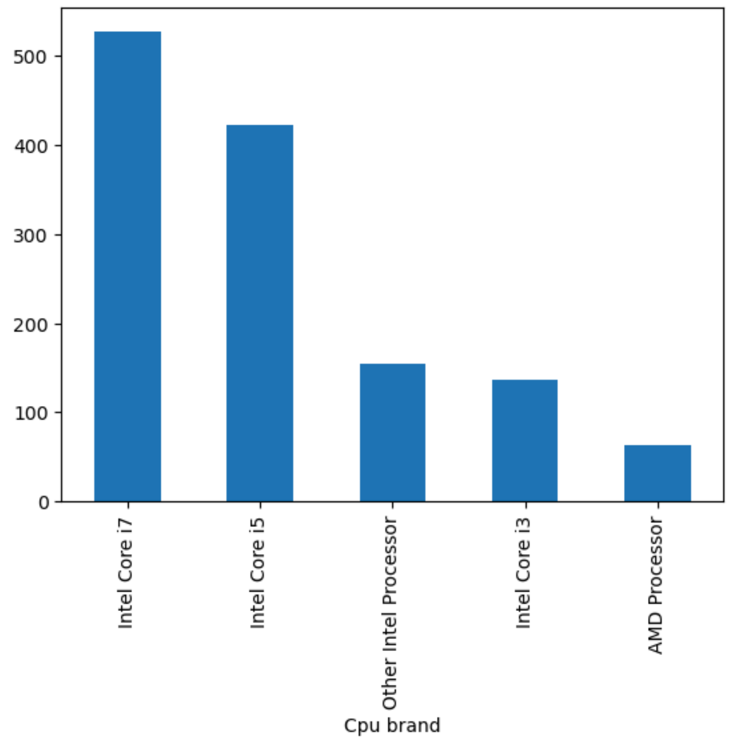
```
df['Cpu brand'] = df['Cpu Name'].apply(fetch_processor)
```

```
df.head()
```

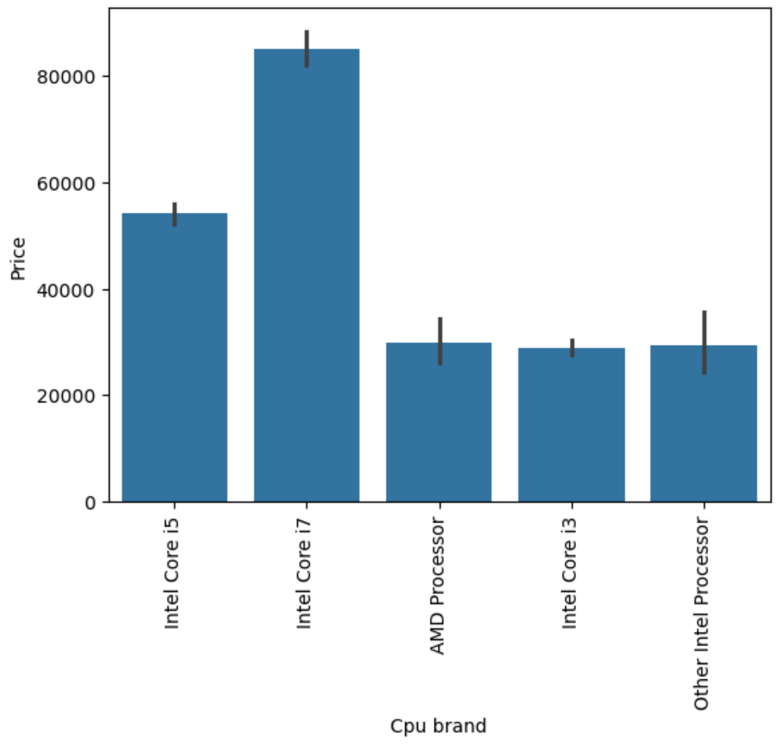
	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips	ppi	Cpu Name	Cpu brand
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	226.983005	Intel Core i5	Intel Core i5
1	Apple	Ultrabook	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	127.677940	Intel Core i5	Intel Core i5
2	HP	Notebook	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	141.211998	Intel Core i5	Intel Core i5
3	Apple	Ultrabook	Intel Core	16	512GB	AMD Radeon Pro	macOS	1.83	135195.3360	0	1	220.534624	Intel Core	Intel Core

```
df['Cpu brand'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Cpu brand'>
```



```
sns.barplot(x=df['Cpu brand'],y=df['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



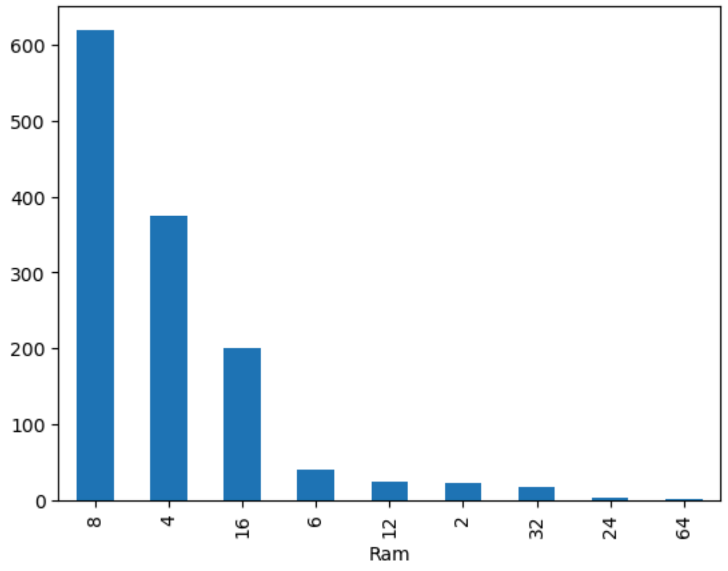
```
df.drop(columns=['Cpu','Cpu Name'],inplace=True)
```

```
df.head()
```

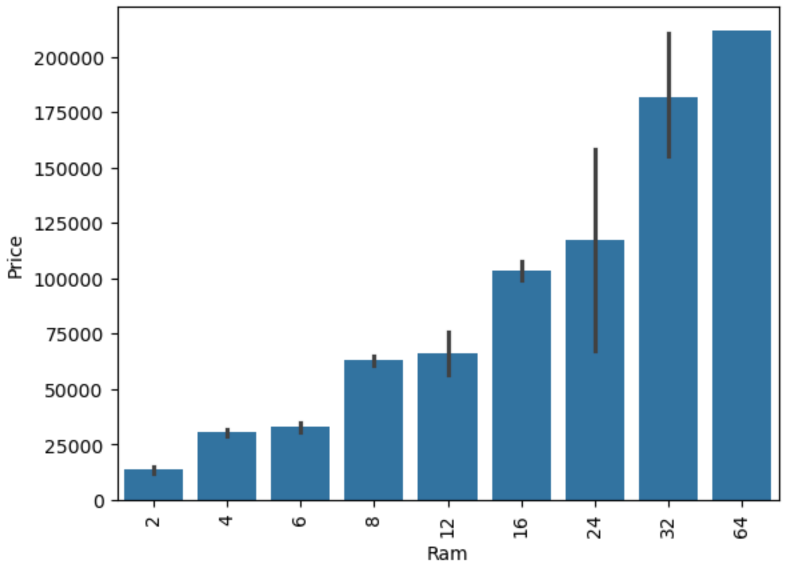
	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips	ppi	Cpu brand
0	Apple	Ultrabook	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	226.983005	Intel Core i5
1	Apple	Ultrabook	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	127.677940	Intel Core i5
2	HP	Notebook	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	141.211998	Intel Core i5

```
df['Ram'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Ram'>
```



```
sns.barplot(x=df['Ram'],y=df['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



```
df['Memory'].value_counts()
```

```
Memory
256GB SSD                      412
1TB HDD                        223
500GB HDD                      132
512GB SSD                      118
128GB SSD +  1TB HDD            94
128GB SSD                       76
256GB SSD +  1TB HDD            73
32GB Flash Storage              38
2TB HDD                         16
64GB Flash Storage              15
1TB SSD                         14
512GB SSD +  1TB HDD            14
256GB SSD +  2TB HDD            10
1.0TB Hybrid                     9
256GB Flash Storage              8
16GB Flash Storage               7
32GB SSD                         6
180GB SSD                        5
128GB Flash Storage              4
16GB SSD                         3
512GB SSD +  2TB HDD             3
128GB SSD +  2TB HDD             2
256GB SSD +  256GB SSD           2
512GB Flash Storage              2
1TB SSD +  1TB HDD               2
256GB SSD +  500GB HDD           2
64GB SSD                         1
512GB SSD +  512GB SSD           1
64GB Flash Storage +  1TB HDD    1
1TB HDD +  1TB HDD               1
512GB SSD +  256GB SSD           1
32GB HDD                         1
128GB HDD                        1
240GB SSD                        1
8GB SSD                          1
508GB Hybrid                     1
1.0TB HDD                        1
512GB SSD +  1.0TB Hybrid        1
256GB SSD +  1.0TB Hybrid        1
Name: count, dtype: int64
```

```
df['Memory'] = df['Memory'].astype(str).replace('\.0', '', regex=True)
df["Memory"] = df["Memory"].str.replace('GB', '')
df["Memory"] = df["Memory"].str.replace('TB', '000')
new = df["Memory"].str.split("+", n = 1, expand = True)

df["first"]= new[0]
df["first"]=df["first"].str.strip()
```

```
df["second"]= new[1]

df["Layer1HDD"] = df["first"].apply(lambda x: 1 if "HDD" in x else 0)
df["Layer1SSD"] = df["first"].apply(lambda x: 1 if "SSD" in x else 0)
df["Layer1Hybrid"] = df["first"].apply(lambda x: 1 if "Hybrid" in x else 0)
df["Layer1Flash_Storage"] = df["first"].apply(lambda x: 1 if "Flash Storage" in x else 0)

df['first'] = df['first'].str.replace(r'\D', '')

df["second"].fillna("0", inplace = True)

df["Layer2HDD"] = df["second"].apply(lambda x: 1 if "HDD" in x else 0)
df["Layer2SSD"] = df["second"].apply(lambda x: 1 if "SSD" in x else 0)
df["Layer2Hybrid"] = df["second"].apply(lambda x: 1 if "Hybrid" in x else 0)
df["Layer2Flash_Storage"] = df["second"].apply(lambda x: 1 if "Flash Storage" in x else 0)

df['second'] = df['second'].str.replace(r'\D', '')

df["first"] = df["first"].astype(int)
df["second"] = df["second"].astype(int)

df["HDD"]=(df["first"]*df["Layer1HDD"]+df["second"]*df["Layer2HDD"])
df["SSD"]=(df["first"]*df["Layer1SSD"]+df["second"]*df["Layer2SSD"])
df["Hybrid"]=(df["first"]*df["Layer1Hybrid"]+df["second"]*df["Layer2Hybrid"])
df["Flash_Storage"]=(df["first"]*df["Layer1Flash_Storage"]+df["second"]*df["Layer2Flash_Storage"])

df.drop(columns=['first', 'second', 'Layer1HDD', 'Layer1SSD', 'Layer1Hybrid',
       'Layer1Flash_Storage', 'Layer2HDD', 'Layer2SSD', 'Layer2Hybrid',
       'Layer2Flash_Storage'],inplace=True)
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_25308\4023190604.py:18: FutureWarning: A value is trying to be set on a copy of a Dat
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setti

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[


  df["second"].fillna("0", inplace = True)
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[61], line 27
     23 df["Layer2Flash_Storage"] = df["second"].apply(lambda x: 1 if "Flash Storage" in x else 0)
     25 df['second'] = df['second'].str.replace(r'\D', '')
---> 27 df["first"] = df["first"].astype(int)
     28 df["second"] = df["second"].astype(int)
     30 df["HDD"]=(df["first"]*df["Layer1HDD"]+df["second"]*df["Layer2HDD"])

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\generic.py:6643, in NDFrame.astype(self, dtype, copy, errors)
   6637     results = [
   6638         ser.astype(dtype, copy=copy, errors=errors) for _, ser in self.items()
   6639     ]
   6641 else:
   6642     # else, only a single dtype is given
-> 6643     new_data = self._mgr.astype(dtype=dtype, copy=copy, errors=errors)
   6644     res = self._constructor_from_mgr(new_data, axes=new_data.axes)
   6645     return res.__finalize__(self, method="astype")

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:430, in BaseBlockManager.astype(self, dtype,
copy, errors)
    427 elif using_copy_on_write():
    428     copy = False
--> 430 return self.apply(
    431     "astype",
    432     dtype=dtype,
    433     copy=copy,
    434     errors=errors,
    435     using_cow=using_copy_on_write(),
    436 )

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:363, in BaseBlockManager.apply(self, f,
align_keys, **kwargs)
    361         applied = b.apply(f, **kwargs)
    362     else:
--> 363         applied = getattr(b, f)(**kwargs)
    364     result_blocks = extend_blocks(applied, result_blocks)
    366 out = type(self).from_blocks(result_blocks, self.axes)

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\internals\blocks.py:758, in Block.astype(self, dtype, copy, errors,
using_cow, squeeze)
    755         raise ValueError("Can not squeeze with more than one column.")
    756     values = values[0, :]  # type: ignore[call-overload]
--> 758 new_values = astype_array_safe(values, dtype, copy=copy, errors=errors)
    760 new_values = maybe_coerce_values(new_values)
    762 refs = None

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:237, in astype_array_safe(values, dtype, copy,
errors)
    234     dtype = dtype.numpy_dtype
    236 try:
--> 237     new_values = astype_array(values, dtype, copy=copy)
    238 except (ValueError, TypeError):
    239     # e.g. _astype_nansafe can fail on object-dtype of strings
    240     #  trying to convert to float
    241     if errors == "ignore":

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:182, in astype_array(values, dtype, copy)
    179     values = values.astype(dtype, copy=copy)
    181 else:
--> 182     values = _astype_nansafe(values, dtype, copy=copy)
    184 # in pandas we don't store numpy str dtypes, so convert to object
    185 if isinstance(dtype, np.dtype) and issubclass(values.dtype.type, str):

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\dtypes\astype.py:133, in _astype_nansafe(arr, dtype, copy, skipna)
    129     raise ValueError(msg)
    131 if copy or arr.dtype == object or dtype == object:
    132     # Explicit copy, or required since NumPy can't view from / to object.
--> 133     return arr.astype(dtype, copy=True)
    135 return arr.astype(dtype, copy=copy)
```

```
df['Memory'] = df['Memory'].astype(str).replace('\.0', '', regex=True)
df["Memory"] = df["Memory"].str.replace('GB', '')
df["Memory"] = df["Memory"].str.replace('TB', '000')
new = df["Memory"].str.split("+", n = 1, expand = True)
```