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

```
df=pd.read_csv("/content/laptop...csv")
df
```

	Company	TypeName	Inches	ScreenResolution	Cpu	HDD Memory	SSD Memory	RAM	GPU	Operating System	Weight	Price
0	Apple	NaN	13.3	NaN	IntelCorei5	NaN	128GB	8GB	Intel	macOS	1.37kg	71378.6832
1	Apple	NaN	13.3	1440x900	IntelCorei5	NaN	NaN	8GB	Intel	macOS	1.34kg	47895.5232
2	HP	Notebook	15.6	1920x1080	IntelCorei5	NaN	256GB	8GB	Intel	NaN	1.86kg	30636.0000
3	Apple	NaN	15.4	2880x1800	IntelCorei7	NaN	512GB	16GB	AMD	macOS	1.83kg	135195.3360
4	Apple	NaN	13.3	NaN	IntelCorei5	NaN	256GB	8GB	Intel	macOS	1.37kg	96095.8080
...	...	...	...	...	...	...	...	...	...	...	...	...
1298	Lenovo	2in1Convertible	14.0	1920x1080	IntelCorei7	NaN	128GB	4GB	Intel	Windows10	1.8kg	33992.6400
1299	Lenovo	2in1Convertible	13.3	3200x1800	IntelCorei7	NaN	512GB	16GB	Intel	Windows10	1.3kg	79866.7200
1300	Lenovo	Notebook	14.0	1366x768	IntelCeleronD	NaN	NaN	2GB	Intel	Windows10	1.5kg	12201.1200
1301	HP	Notebook	15.6	1366x768	IntelCorei7	102GB	NaN	6GB	AMD	Windows10	2.19kg	40705.9200
1302	As	Notebook	15.6	1366x768	IntelCeleronD	500GB	NaN	4GB	Intel	Windows10	2.2kg	19660.3200

1303 rows × 12 columns

```
df.size
```

```
15636
```

```
row,columns=df.shape
print("The number of rows are",row)
print("The number of columns are",columns)
```

```
The number of rows are 1303
The number of columns are 12
```

```
df.columns
```

```
Index(['Company', 'TypeName', 'Inches', 'ScreenResolution', 'Cpu',
       'HDD Memory', 'SSD Memory', 'RAM', 'GPU', 'Operating System', 'Weight',
       'Price'],
      dtype='object')
```

```
df.head()
```

	Company	TypeName	Inches	ScreenResolution	Cpu	HDD Memory	SSD Memory	RAM	GPU	Operating System	Weight	Price
0	Apple	NaN	13.3	NaN	IntelCorei5	NaN	128GB	8GB	Intel	macOS	1.37kg	71378.6832
1	Apple	NaN	13.3	1440x900	IntelCorei5	NaN	NaN	8GB	Intel	macOS	1.34kg	47895.5232
2	HP	Notebook	15.6	1920x1080	IntelCorei5	NaN	256GB	8GB	Intel	NaN	1.86kg	30636.0000
3	Apple	NaN	15.4	2880x1800	IntelCorei7	NaN	512GB	16GB	AMD	macOS	1.83kg	135195.3360
4	Apple	NaN	13.3	NaN	IntelCorei5	NaN	256GB	8GB	Intel	macOS	1.37kg	96095.8080

```
df.tail()
```

	Company	TypeName	Inches	ScreenResolution	Cpu	HDD Memory	SSD Memory	RAM	GPU	Operating System	Weight	Price
1298	Lenovo	2in1Convertible	14.0	1920x1080	IntelCorei7	NaN	128GB	4GB	Intel	Windows10	1.8kg	33992.64
1299	Lenovo	2in1Convertible	13.3	3200x1800	IntelCorei7	NaN	512GB	16GB	Intel	Windows10	1.3kg	79866.72
1300	Lenovo	Notebook	14.0	1366x768	IntelCeleronD	NaN	NaN	2GB	Intel	Windows10	1.5kg	12201.12
1301	HP	Notebook	15.6	1366x768	IntelCorei7	102GB	NaN	6GB	AMD	Windows10	2.19kg	40705.92
1302	As	Notebook	15.6	1366x768	IntelCeleronD	500GB	NaN	4GB	Intel	Windows10	2.2kg	19660.32

DEALING WITH MISSING VALUES

```
df.isna().sum()
```

```
Company               7
TypeName            196
Inches               70
ScreenResolution     55
Cpu                   0
HDD Memory          715
SSD Memory          460
RAM                   0
GPU                   0
Operating System     66
Weight                0
Price                 0
dtype: int64
```

```
df["Company"]=df["Company"].fillna(df["Company"].mode().iloc[0])
```

```
df["TypeName"]=df["TypeName"].fillna(df["TypeName"].mode().iloc[0])
```

```
df["Inches"]=df["Inches"].fillna(df["Inches"].mode().iloc[0])
```

```
df["ScreenResolution"]=df["ScreenResolution"].fillna(df["ScreenResolution"].mode().iloc[0])
```

```
df["HDD Memory"]=df["HDD Memory"].fillna("0GB")
```

```
df["SSD Memory"]=df["SSD Memory"].fillna("0GB")
```

```
df["Operating System"]=df["Operating System"].fillna(df["Operating System"].mode().iloc[0])
```

```
df.isna().sum()
```

```
Company             0
TypeName            0
```

```
Inches              0
ScreenResolution    0
Cpu                 0
HDD Memory          0
SSD Memory          0
RAM                 0
GPU                 0
Operating System    0
Weight              0
Price               0
dtype: int64
```

```
df["Price"]=df["Price"].round()
```

```
df
```

	Company	TypeName	Inches	ScreenResolution	Cpu	HDD Memory	SSD Memory	RAM	GPU	Operating System	Weight	Price
0	Apple	Notebook	13.3	1920x1080	IntelCorei5	0GB	128GB	8GB	Intel	macOS	1.37kg	71379.0
1	Apple	Notebook	13.3	1440x900	IntelCorei5	0GB	0GB	8GB	Intel	macOS	1.34kg	47896.0
2	HP	Notebook	15.6	1920x1080	IntelCorei5	0GB	256GB	8GB	Intel	Windows10	1.86kg	30636.0
3	Apple	Notebook	15.4	2880x1800	IntelCorei7	0GB	512GB	16GB	AMD	macOS	1.83kg	135195.0
4	Apple	Notebook	13.3	1920x1080	IntelCorei5	0GB	256GB	8GB	Intel	macOS	1.37kg	96096.0
...	...	...	...	...	...	...	...	...	...	...	...	...
1298	Lenovo	2in1Convertible	14.0	1920x1080	IntelCorei7	0GB	128GB	4GB	Intel	Windows10	1.8kg	33993.0
1299	Lenovo	2in1Convertible	13.3	3200x1800	IntelCorei7	0GB	512GB	16GB	Intel	Windows10	1.3kg	79867.0
1300	Lenovo	Notebook	14.0	1366x768	IntelCeleronD	0GB	0GB	2GB	Intel	Windows10	1.5kg	12201.0
1301	HP	Notebook	15.6	1366x768	IntelCorei7	102GB	0GB	6GB	AMD	Windows10	2.19kg	40706.0
1302	As	Notebook	15.6	1366x768	IntelCeleronD	500GB	0GB	4GB	Intel	Windows10	2.2kg	19660.0

1303 rows × 12 columns

```
df.isna().sum()
```

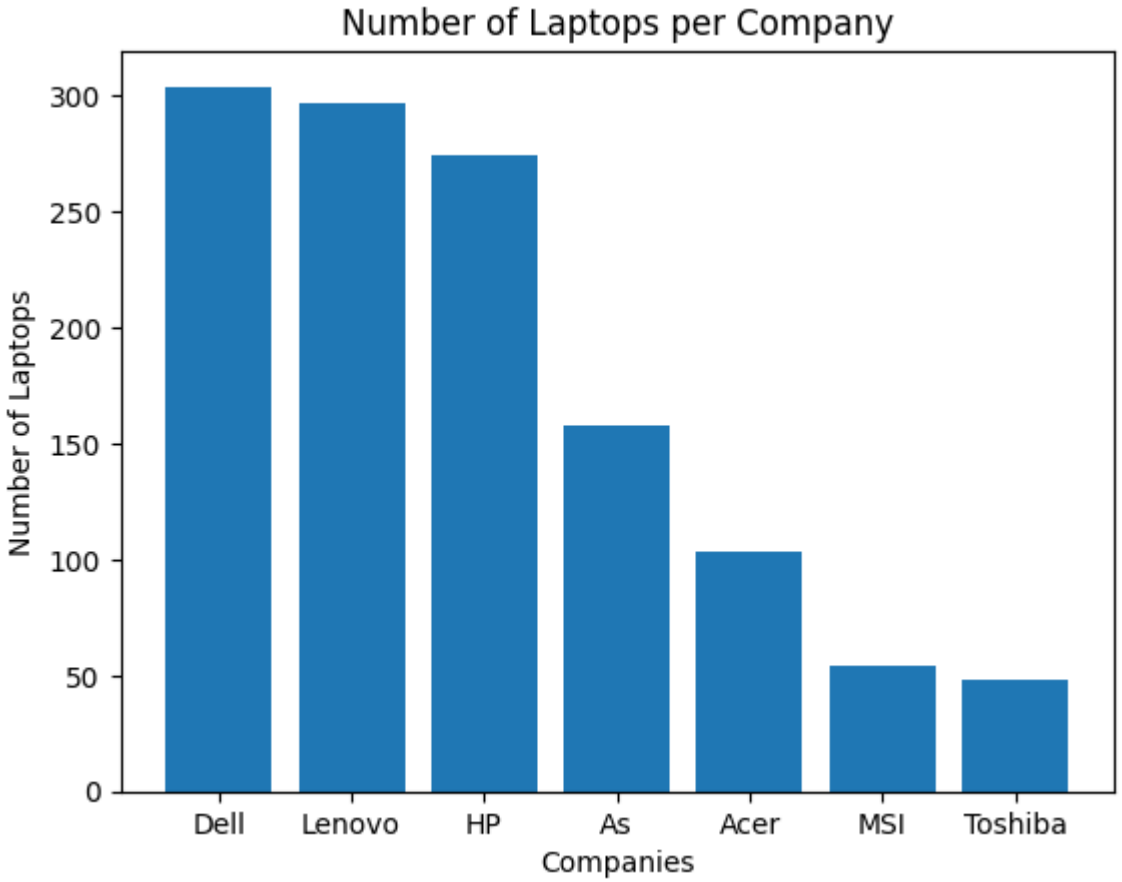
```
Company             0
TypeName            0
Inches              0
ScreenResolution    0
Cpu                 0
HDD Memory          0
SSD Memory          0
RAM                 0
GPU                 0
Operating System    0
Weight              0
Price               0
dtype: int64
```

Data Visualisation

```
df.value_counts("Company")
```

```
Company
Dell         304
Lenovo       297
HP           274
As           158
Acer         103
MSI           54
Toshiba       48
Apple         21
Sams           9
Mediacom       7
Microsoft      6
Vero           4
Xiaomi         4
Google         3
F              3
Ch             3
LG             3
H              2
dtype: int64
```

```
import matplotlib.pyplot as plt
company_counts = df["Company"].value_counts().head(7)
plt.bar(company_counts.index, company_counts.values)
plt.xlabel("Companies")
plt.ylabel("Number of Laptops")
plt.title("Number of Laptops per Company")
plt.show()
```



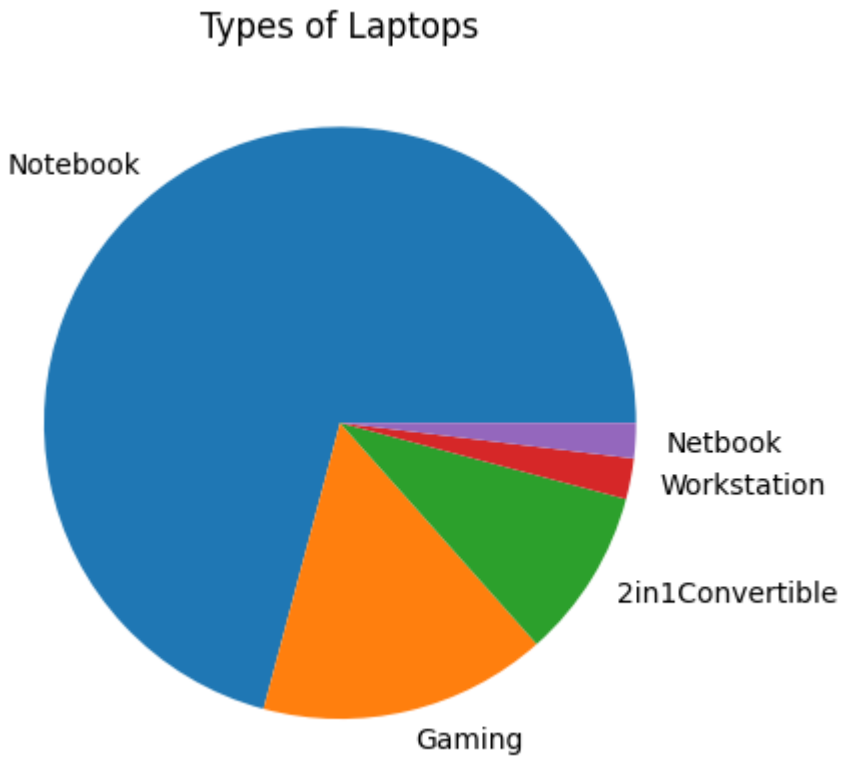

```
type_counts=df["TypeName"].value_counts()
type_counts
```

```
Notebook            923
Gaming              205
2in1Convertible     121
Workstation          29
```

```
Netbook             25
Name: TypeName, dtype: int64
```

```
plt.pie(type_counts.values,labels=type_counts.index)
plt.title("Types of Laptops")
```

```
Text(0.5, 1.0, 'Types of Laptops')
```

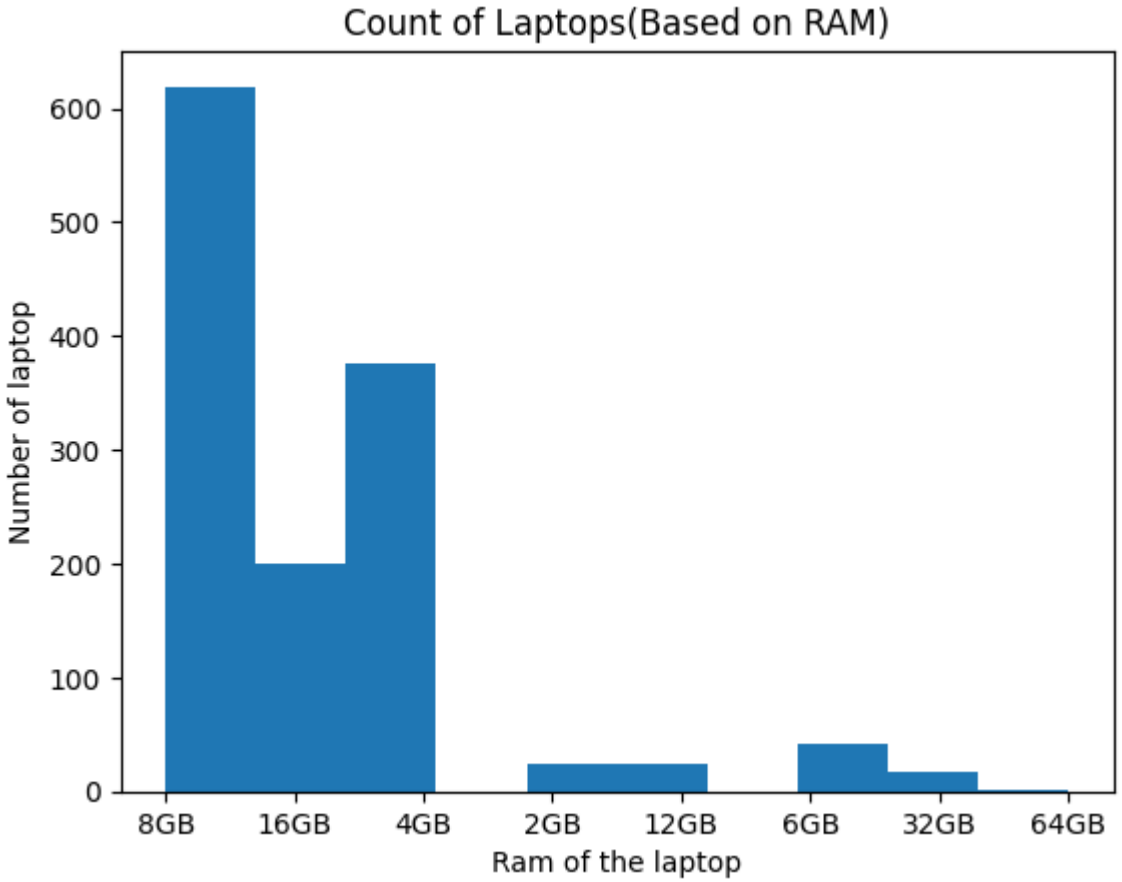



```
df.value_counts("RAM")
```

```
RAM
8GB     619
4GB     375
16GB    200
6GB      41
12GB     25
2GB      25
32GB     17
64GB      1
dtype: int64
```

```
plt.hist(df["RAM"])
plt.xlabel("Ram of the laptop")
plt.ylabel("Number of laptop")
plt.title("Count of Laptops(Based on RAM)")
```

```
Text(0.5, 1.0, 'Count of Laptops(Based on RAM)')
```



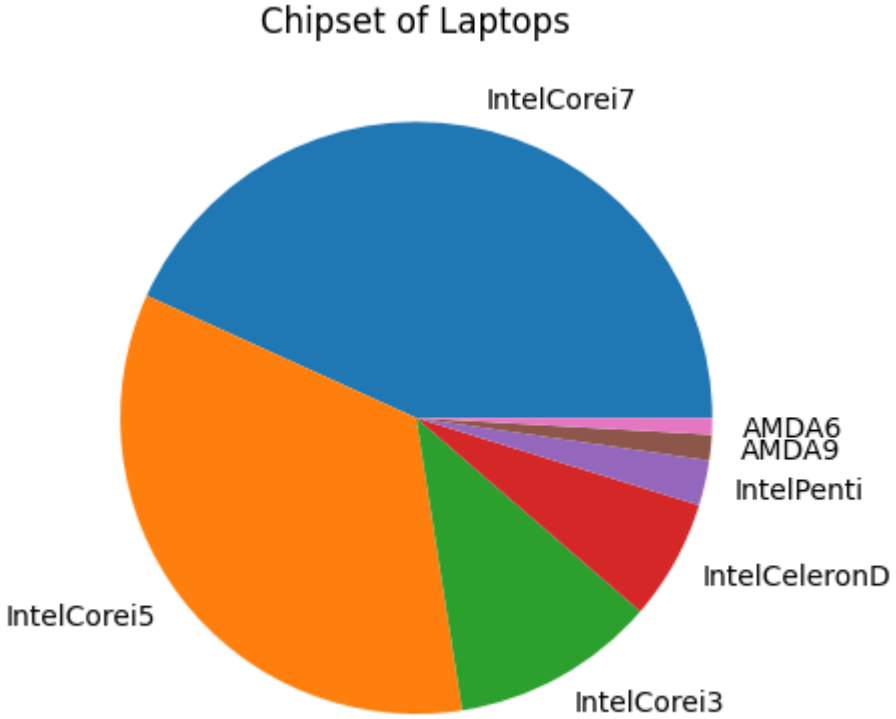

```
Cpu_type=df["Cpu"].value_counts().head(7)
Cpu_type
```

```
IntelCorei7      526
IntelCorei5      417
IntelCorei3      136
IntelCeleronD     80
IntelPenti        30
AMDA9             17
```

```
AMDA6              11
Name: Cpu, dtype: int64
```

```
plt.pie(Cpu_type.values,labels=Cpu_type.index)
plt.title("Chipset of Laptops")
```

```
Text(0.5, 1.0, 'Chipset of Laptops')
```

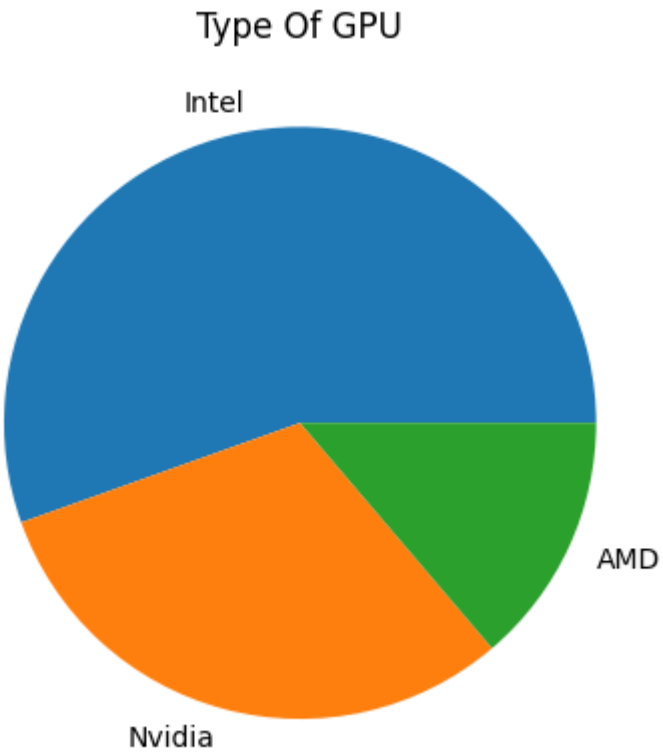



```
gpu_type=df["GPU"].value_counts().head(3)
gpu_type
```

```
Intel     721
Nvidia    400
AMD       179
Name: GPU, dtype: int64
```

```
plt.pie(gpu_type.values,labels=gpu_type.index)
plt.title("Type Of GPU")
```

```
Text(0.5, 1.0, 'Type Of GPU')
```

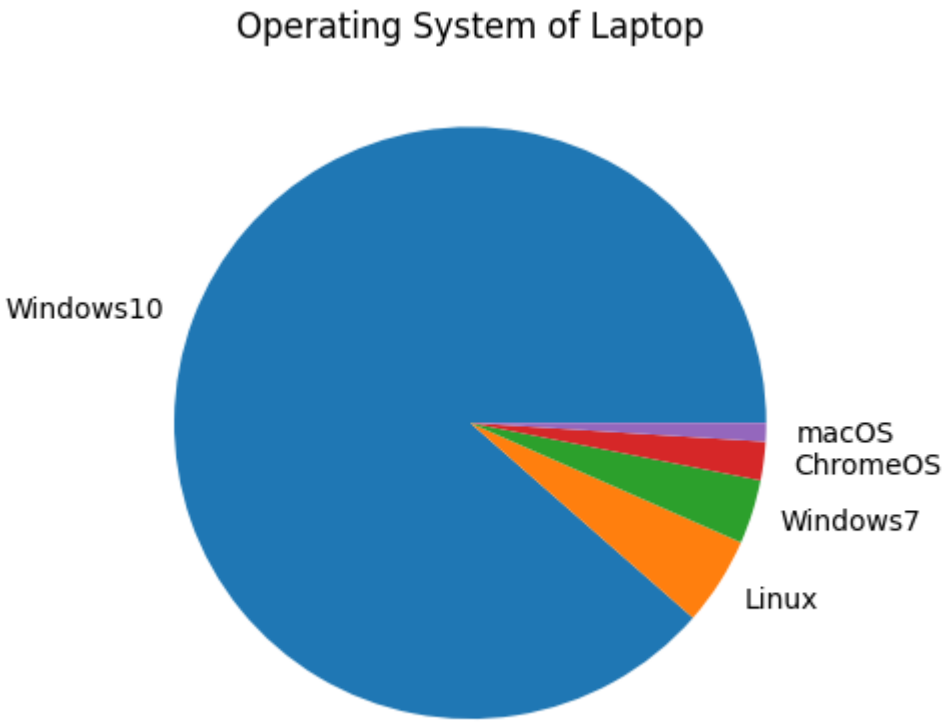



```
op_type=df["Operating System"].value_counts().head()
op_type
```

```
Windows10    1138
Linux          62
Windows7       45
ChromeOS       27
macOS          13
Name: Operating System, dtype: int64
```

```
plt.pie(op_type.values,labels=op_type.index)
plt.title("Operating System of Laptop")
```

```
Text(0.5, 1.0, 'Operating System of Laptop')
```




```
df["Inches"]=df["Inches"].astype(int)
```

```
df["HDD Memory"]=df["HDD Memory"].str.replace("GB","")
```

```
df["SSD Memory"]=df["SSD Memory"].str.replace("GB","")
```

```
df["RAM"]=df["RAM"].str.replace("GB","")
```

```
df=df.drop("Weight",axis=1)
```

```
dummy=pd.get_dummies(df[["Company","TypeName","ScreenResolution","Cpu","GPU","Operating System"]],drop_first=True)
dummy
```

	Company_Apple	Company_As	Company_Ch	Company_Dell	Company_F	Company_Google	Company_H	Company_HP	Company_LG	Company_Lenovo	.
0	1	0	0	0	0	0	0	0	0	0	
1	1	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	1	0	0	
3	1	0	0	0	0	0	0	0	0	0	
4	1	0	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	
1298	0	0	0	0	0	0	0	0	0	1	
1299	0	0	0	0	0	0	0	0	0	1	
1300	0	0	0	0	0	0	0	0	0	1	
1301	0	0	0	0	0	0	0	1	0	0	
1302	0	1	0	0	0	0	0	0	0	0	

1303 rows × 72 columns

```
df1=pd.concat([df,dummy],axis=1)
```

```
df1=df1.drop(["Company","TypeName","ScreenResolution","Cpu","GPU","Operating System"],axis=1)
df1
```

	Inches	HDD Memory	SSD Memory	RAM	Price	Company_Apple	Company_As	Company_Ch	Company_Dell	Company_F	...	GPU_Intel	GPU_IntelG	GP
0	13	0	128	8	71379.0	1	0	0	0	0	...	1	0	
1	13	0	0	8	47896.0	1	0	0	0	0	...	1	0	
2	15	0	256	8	30636.0	0	0	0	0	0	...	1	0	
3	15	0	512	16	135195.0	1	0	0	0	0	...	0	0	
4	13	0	256	8	96096.0	1	0	0	0	0	...	1	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1298	14	0	128	4	33993.0	0	0	0	0	0	...	1	0	
1299	13	0	512	16	79867.0	0	0	0	0	0	...	1	0	
1300	14	0	0	2	12201.0	0	0	0	0	0	...	1	0	
1301	15	102	0	6	40706.0	0	0	0	0	0	...	0	0	
1302	15	500	0	4	19660.0	0	1	0	0	0	...	1	0	

1303 rows × 77 columns

```
x=df1.drop("Price",axis=1)
```

```
y=df1.iloc[:,5]
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30)
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
y_pred
```

```
 1.54390389e-16, -4.68375339e-17, -8.32667268e-17,  7.28583860e-17,
-1.70002901e-16,  4.68375339e-17, -7.80625564e-17, -5.72458747e-17,
 1.57859836e-16,  8.50014503e-17, -1.56125113e-17,  6.41847686e-17,
-2.42861287e-16, -1.37043155e-16, -2.60208521e-17, -6.24500451e-17,
-3.12250226e-17, -5.20417043e-17,  7.45931095e-17,  1.17155942e-06,
-6.24500451e-17,  8.84708973e-17, -9.19403442e-17, -1.04083409e-16,
-1.04083409e-17,  7.80625564e-17, -2.56739074e-16,  6.76542156e-17,
 9.02056208e-17, -4.96130914e-16,  2.48065457e-16,  3.12250226e-17,
 6.07153217e-17, -1.71737624e-16, -2.51534904e-16, -6.07153217e-17,
-5.89805982e-17,  8.67361738e-17, -1.24900090e-16, -3.64291930e-17,
 7.80625564e-17,  1.42247325e-16,  2.30718222e-16, -6.76542156e-17,
-2.08166817e-17, -5.20417043e-18,  5.03069808e-17,  5.37764278e-17,
 7.45931095e-17, -6.07153217e-17,  6.07153217e-17,  2.32452946e-16,
-1.90819582e-17,  8.50014503e-17,  8.15320034e-17,  7.63278329e-17,
 4.85722573e-17,  4.68375339e-17, -1.33573708e-16,  4.33680869e-17,
 1.49186219e-16,  6.07153217e-17,  6.93889390e-17,  3.55618313e-16,
 9.88792381e-17, -1.04083409e-17, -1.57859836e-16,  2.94902991e-17,
 7.63278329e-17, -1.26634814e-16,  9.99974732e-01, -1.89084859e-16,
-1.45716772e-16, -5.72458747e-17, -1.66533454e-16])
```

```
print(f"Slope is {model.coef_}")
```

```
Slope is [ 6.72534889e-17  3.12250226e-17  5.93736215e-17 -3.81639165e-17
  6.55307894e-02  6.93889390e-18 -3.12250226e-17  3.46944695e-17
 -3.46944695e-18 -1.84545925e-03  1.04083409e-17  3.12250226e-17
  6.72205347e-18  2.53703308e-17 -7.80625564e-18  1.30104261e-17
 -1.23470270e-03  0.00000000e+00  1.34441069e-17  4.23923049e-17
 -2.27682456e-18 -5.37764278e-17 -4.33680869e-18 -2.25514052e-17
 -1.90819582e-17 -2.52679253e-05 -2.94902991e-17  0.00000000e+00
  2.12978033e-03  1.70562927e-03  1.23470270e-03 -6.24500451e-17
  6.07153217e-18  1.56125113e-17 -6.07153217e-18 -1.21430643e-17
 -1.73472348e-17 -1.34441069e-17 -3.46944695e-18 -3.90312782e-18
  1.08420217e-18  1.17155942e-06 -1.25767452e-17 -2.08166817e-17
 -8.67361738e-18 -1.70761842e-18 -2.42861287e-17 -2.86229374e-17
 -1.70562927e-03  3.90312782e-18 -5.20417043e-18 -6.75191673e-12
 -1.86482774e-17 -2.55871713e-17 -1.13841228e-17 -4.11996826e-17
 -5.89805982e-17 -2.60208521e-18 -1.56125113e-17 -9.80118764e-17
 -1.90819582e-17 -3.03576608e-17 -2.39879731e-18 -5.33323947e-04
 -1.51788304e-18 -5.33323947e-04  2.34187669e-17 -1.08420217e-19
  2.21177243e-17  1.69135539e-17  5.68121938e-17  3.23116603e-02
  2.25514052e-17  1.06251813e-17  1.43656788e-17  2.95126492e-02]
```

```
print(f"Constant is {model.intercept_}")
```

```
Constant is 0.01206140350877193
```

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
print("Mean Absolute Error is :",mean_absolute_error(y_pred,y_test))
```

```
Mean Absolute Error is : 0.00013228759264700114
```

```
print("Mean Squared Error is : ",mean_squared_error(y_pred, y_test))
```

```
Mean Squared Error is :  3.4077457887508363e-06
```

```
print("Root Mean Squared Error is : ",np.sqrt(mean_squared_error(y_pred, y_test)))
```

```
Root Mean Squared Error is :  0.0018460080684414239
```

```
print("R2 Score is :",r2_score(y_pred,y_test))
```

```
R2 Score is : 0.9998646728033724
```