analyze_computer_prices_PDF_code

April 13, 2025

1 Phân tích doanh số bán máy tính sách tay trong năm 2024

GIẢ SỬ BẠN ĐANG LÀ NHÂN VIÊN PHÂN TÍCH DỮ LIỆU CỦA MỘT CÔNG TY BÁN MÁY TÍNH SÁCH TAY VÀ BẠN ĐANG PHÂN TÍCH VỀ DỮ LIỆU BÁN HÀNG CỦA CÔNG TY TRONG NĂM 2024

1.0.1 B1 : xác định vấn đề

[1]:
```
# import thư viện
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from random import randint
```

1.0.2 B2: Thu thập dữ liệu

[2]:
```
# load dataset
data = pd.read_csv('C:\DATA\laptop_prices.csv')
data['LY price'] = data['Price ($)'] - 100
months = ["January", "February", "March", "April", "May", "June",
          "July", "August", "September", "October", "November", "December"]
months_of_df = []
for i in range(len(data)):
    months_of_df.append(months[randint(0,len(months)-1)])
data['months'] = months_of_df
data
```

[2]:

	Brand	Processor	RAM (GB)	Storage	GPU	\
0	Apple	AMD Ryzen 3	64.0	512GB SSD	Nvidia GTX 1650	
1	Razer	AMD Ryzen 7	4.0	1TB SSD	Nvidia RTX 3080	
2	Asus	Intel i5	32.0	2TB SSD	Nvidia RTX 3060	
3	Lenovo	Intel i5	4.0	256GB SSD	Nvidia RTX 3080	
4	Razer	Intel i3	4.0	256GB SSD	AMD Radeon RX 6600	
…	…	…	…	…	…	
11766	Samsung	AMD Ryzen 7	16.0	512GB SSD	Integrated	
11767	Samsung	Intel i7	8.0	256GB SSD	Nvidia RTX 3080	
11768	Samsung	Intel i7	8.0	256GB SSD	Nvidia RTX 3080	

```
11769  Samsung     Intel i7       8.0  256GB SSD     Nvidia RTX 3080
11770  Samsung     Intel i7       8.0  256GB SSD     Nvidia RTX 3080

       Screen Size (inch) Resolution  Battery Life (hours)  Weight (kg)  \
0                    17.3  2560x1440                   8.9         1.42
1                    14.0   1366x768                   9.4         2.57
2                    13.3  3840x2160                   8.5         1.74
3                    13.3   1366x768                  10.5         3.10
4                    16.0  3840x2160                   5.7         3.38
…                     …        …                      …           …
11766                13.3  1920x1080                   7.5         1.48
11767                17.3  2560x1440                   6.4         2.45
11768                17.3  2560x1440                   6.4         2.45
11769                17.3  2560x1440                   6.4         2.45
11770                17.3  2560x1440                   6.4         2.45

      Operating System  Price ($)  LY price    months
0              FreeDOS    3997.07   3897.07   October
1                Linux    1355.78   1255.78  December
2              FreeDOS    2673.07   2573.07   October
3              Windows     751.17    651.17    August
4                Linux    2059.83   1959.83   October
…                  …          …         …         …
11766            macOS    1067.13    967.13   January
11767          FreeDOS    1579.55   1479.55  November
11768          FreeDOS    1579.55   1479.55       May
11769          FreeDOS    1579.55   1479.55      July
11770          FreeDOS    1579.55   1479.55   October

[11771 rows x 13 columns]
```

[3]:
```
# from sqlalchemy import create_engine
# engine = create_engine('postgresql://postgres:VIVIAN1982@localhost:5432/CS số␣
 ↪6 Phạm Văn Đồng')
# data.to_sql(name='doanh_so', con=engine, if_exists='replace', index=False)
```

[4]:
```
# in ra 5 dòng đầu
data.head(5)
```

[4]:
```
    Brand    Processor  RAM (GB)    Storage                GPU  \
0   Apple  AMD Ryzen 3      64.0  512GB SSD     Nvidia GTX 1650
1   Razer  AMD Ryzen 7       4.0    1TB SSD     Nvidia RTX 3080
2    Asus     Intel i5      32.0    2TB SSD     Nvidia RTX 3060
3  Lenovo     Intel i5       4.0  256GB SSD     Nvidia RTX 3080
4   Razer     Intel i3       4.0  256GB SSD  AMD Radeon RX 6600

   Screen Size (inch) Resolution  Battery Life (hours)  Weight (kg)  \
```

```
0                17.3  2560x1440                  8.9         1.42
1                14.0   1366x768                  9.4         2.57
2                13.3  3840x2160                  8.5         1.74
3                13.3   1366x768                 10.5         3.10
4                16.0  3840x2160                  5.7         3.38

  Operating System  Price ($)  LY price    months
0          FreeDOS    3997.07   3897.07   October
1            Linux    1355.78   1255.78  December
2          FreeDOS    2673.07   2573.07   October
3          Windows     751.17    651.17    August
4            Linux    2059.83   1959.83   October
```

[5]:
```
# in ra 5 dòng cuối
data.tail(5)
```

[5]:
```
          Brand    Processor  RAM (GB)    Storage              GPU  \
11766  Samsung  AMD Ryzen 7      16.0  512GB SSD       Integrated
11767  Samsung     Intel i7       8.0  256GB SSD  Nvidia RTX 3080
11768  Samsung     Intel i7       8.0  256GB SSD  Nvidia RTX 3080
11769  Samsung     Intel i7       8.0  256GB SSD  Nvidia RTX 3080
11770  Samsung     Intel i7       8.0  256GB SSD  Nvidia RTX 3080

       Screen Size (inch) Resolution  Battery Life (hours)  Weight (kg)  \
11766                13.3  1920x1080                   7.5         1.48
11767                17.3  2560x1440                   6.4         2.45
11768                17.3  2560x1440                   6.4         2.45
11769                17.3  2560x1440                   6.4         2.45
11770                17.3  2560x1440                   6.4         2.45

      Operating System  Price ($)  LY price    months
11766            macOS    1067.13    967.13   January
11767          FreeDOS    1579.55   1479.55  November
11768          FreeDOS    1579.55   1479.55       May
11769          FreeDOS    1579.55   1479.55      July
11770          FreeDOS    1579.55   1479.55   October
```

[6]:
```
# in ra thông tin tổng quát của dataframe
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11771 entries, 0 to 11770
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Brand                 11686 non-null  object
 1   Processor             11727 non-null  object
 2   RAM (GB)              11756 non-null  float64
```

```
 3   Storage               11771 non-null  object
 4   GPU                   11771 non-null  object
 5   Screen Size (inch)    11771 non-null  float64
 6   Resolution            11771 non-null  object
 7   Battery Life (hours)  11767 non-null  float64
 8   Weight (kg)           11771 non-null  float64
 9   Operating System      11766 non-null  object
 10  Price ($)             11771 non-null  float64
 11  LY price              11771 non-null  float64
 12  months                11771 non-null  object
dtypes: float64(6), object(7)
memory usage: 1.2+ MB
```

Tổng quá về dữ liệu *- Dữ liệu được thu tập từ cơ sở dữ liệu về doanh số bán hàng của công ty trong năm 2024 bằng SQL - Tập dữ liệu nói về nhu cầu mua , sử dụng và xu hướng chọn mua máy laptop cá nhân của khách hàng trong 1 năm qua - Tập dữ liệu bao gồm những thông tin cơ bản sau: + dữ liệu có tổng cộng 11771 sample + dữ liệu có 13 feature (columns) bao gồm :*

```
*Brand : Nói về các hãng máy tính như Apple , MSI , asus , ....*
*Preocesser : chứa thông tin về các bộ xử lý của máy tính như AMD Ryzen , intel 3 , 5,
*RAM (GB) : chứa thông thi về dung lượng ram mà máy tính có tính theo đơn vị GB*
*Storage : chứa thông tin về dụng lượng ổ cứng và loại ổ cứng sử dụng SSD và HHD*
*GPU : Chứa thông tin về các loại GPU của máy tính như Nvidia RTX 3080 , Nvidia RTX 31
*Screen Size (inch) : chứa thông tin về kích thước của màn hình tính theo đơn vị inch*
*Resolution : Chứa thông tin về độ phân giải của màn hình , tức là độ nét của màn hình
*Battery Life (hours) : chứa thông tin về vòng đời bin của laptop tức là bin sẽ sử dụng
*Weight (kg) : chứa thông tin về cân nặng của laptop*
*Operating System : nói về hệ điều hành của laptop như window , mas , linux , ..*
*Price ($) : giá tiền bán của chiếc laptop đó*
*LY price : giá của laptop so với cùng kì năm ngoái*
*months : các tháng trong năm*
</div>
+ *trong đó có 6 feature có kiểu dữ liệu float64 và có 7 feature có kiểu dữ liệu là object*
+ *tập dữ liệu sử dụng hết 1.2+ MB của bộ nhớ chính*
```

[7]:
```
# in ra thông tin về chiều và cột
data.shape
```

[7]: (11771, 13)

1.0.3 B3 : Tiền sử lý dữ liệu

- xử lý các giá trị thiếu

[8]:
```
# lấy ra tên các cột có giá trị thiếu
cot_co_gia_tri_null = data.columns[data.isnull().any()]
cot_co_gia_tri_null
```

[8]:
```
Index(['Brand', 'Processor', 'RAM (GB)', 'Battery Life (hours)',
       'Operating System'],
      dtype='object')
```

[9]:
```
# kiểm tra xem mỗi cột có bao nhiêu giá trị thiếu
data.isna().sum()
```

[9]:
```
Brand                  85
Processor              44
RAM (GB)               15
Storage                 0
GPU                     0
Screen Size (inch)      0
Resolution              0
Battery Life (hours)    4
Weight (kg)             0
Operating System        5
Price ($)               0
LY price                0
months                  0
dtype: int64
```

Dữ liệu thiếu - *Các cột có giá trị thiếu bao gồm : + Brand: 85 miss , Processer: 44 miss, RAM : 15 miss,Battery Life : 4 miss,Operating System : 5 miss*

[10]:
```
# xử lý các giá trị thiếu và bị null
for i in range(len(cot_co_gia_tri_null)):
    if(data[cot_co_gia_tri_null[i]].dtype == 'O'):
        data[cot_co_gia_tri_null[i]].fillna(data[cot_co_gia_tri_null[i]].
↪mode()[0],inplace=True)
    elif(data[cot_co_gia_tri_null[i]].dtype == 'float'):
        data[cot_co_gia_tri_null[i]].fillna(data[cot_co_gia_tri_null[i]].
↪mean(),inplace=True)
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\490012484.py:4:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  data[cot_co_gia_tri_null[i]].fillna(data[cot_co_gia_tri_null[i]].mode()[0],inp
lace=True)
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\490012484.py:6:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  data[cot_co_gia_tri_null[i]].fillna(data[cot_co_gia_tri_null[i]].mean(),inplac
e=True)
```

[11]:
```
data.isna().sum()
```

[11]:
```
Brand                   0
Processor               0
RAM (GB)                0
Storage                 0
GPU                     0
Screen Size (inch)      0
Resolution              0
Battery Life (hours)    0
Weight (kg)             0
Operating System        0
Price ($)               0
LY price                0
months                  0
dtype: int64
```

Dữ liệu thiếu đã được xử lý

- Xử lý các giá trị trùng lặp

[12]:
```
# xóa đi các giá trị trùng lặp dữ lại giá trị đầu tiên
data = data.drop_duplicates(keep='first')
# kiểm tra giá trị trùng lặp
data.duplicated().sum()
```

[12]:
```
np.int64(0)
```

- Chuyển đổi kiểu dữ liệu

[13]:
```
# chuyển đổi kiểu dữ liệu
data['RAM (GB)'] = data['RAM (GB)'].astype('int')
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 11771 entries, 0 to 11770
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Brand                 11771 non-null  object
 1   Processor             11771 non-null  object
 2   RAM (GB)              11771 non-null  int64
 3   Storage               11771 non-null  object
 4   GPU                   11771 non-null  object
 5   Screen Size (inch)    11771 non-null  float64
 6   Resolution            11771 non-null  object
 7   Battery Life (hours)  11771 non-null  float64
 8   Weight (kg)           11771 non-null  float64
 9   Operating System      11771 non-null  object
 10  Price ($)             11771 non-null  float64
 11  LY price              11771 non-null  float64
 12  months                11771 non-null  object
dtypes: float64(5), int64(1), object(7)
memory usage: 1.2+ MB
```

Chuyển đổi kiểu dữ liệu

Trong dữ liệu có cột RAM đang mang kiểu dữ liệu float (nhưng trên thực tế thì không có 32.5 RAM) nên phải truyển từ float sang int để tránh sai sót*

- Chuẩn hóa dữ liệu
- Tạo các biến mới nếu cần

1.0.4 B4 : Khám phá dữ liệu

- các hàm dùng để khám phá dữ liệu

[14]:
```
# tạo một list danh sách các màu
color_list = [
    "red", "blue", "green", "yellow", "purple",
    "orange", "pink", "brown", "gray", "cyan",
    "magenta", "lime", "indigo", "violet", "gold",
    "silver", "navy", "teal", "coral", "maroon"
]

# tạo list danh sách các mẫu của biến palette
palettes = [
# Bộ màu mặc định (Categorical Palettes)
"deep", "muted", "bright", "pastel", "dark", "colorblind",

# Bộ màu Gradient (Sequential Palettes)
"Blues", "Reds", "Greens", "coolwarm", "magma", "viridis"
]
```

```
# xây dựng hàm vẽ biểu đồ phân phối cho các biến dữ liệu định lượng
def ve_bieu_do_phan_phoi(so_hang,so_cot,ten_cac_cot,x_label,ten_bieu_do):
    fig,ax = plt.subplots(ncols=so_cot,nrows=so_hang,figsize=(20,10))
    index = 0
    for i in range(so_hang):
        for j in range(so_cot):
            try:
                sns.
 ↪histplot(data[ten_cac_cot[index]],bins=20,ax=ax[i,j],color=color_list[index],kde=True)
            except:
                print(1)
            index+=1
    fig.suptitle(ten_bieu_do)
    fig.supxlabel(x_label)

# Xây dựng hàm vẽ biểu đồ cột về sản lượng bán theo từng hãng
def ve_bieu_do_cot_tong_so_luon_ban(ten_cot,ten_bieu_do):
    df_cot = data[ten_cot].value_counts().reset_index().sort_values(by='count')
    ten = df_cot[ten_cot]
    values = df_cot['count']
    x_coordinate = [i for i in range(len(ten))]
    plt.figure(figsize=(20,5))
    sns.
 ↪barplot(x=ten_cot,y='count',data=df_cot,palette=palettes[randint(0,len(palettes)-1)])
    plt.title(ten_bieu_do + "\n")
    plt.xticks(rotation=45,ha='right');
    for x,y in zip(x_coordinate,values):
        plt.text(x-0.1,y+13,y)

# xây dựng hàm vẽ biểu đồ tròn thể hiện phần trăm số lượng bán hàng và doanh␣
 ↪thu theo từng hãng
def ve_bieu_do_tron(ten_cot,ten_bieu_do):
    df_cot = data[ten_cot].value_counts().reset_index().sort_values(by='count')
    ten = df_cot[ten_cot]
    values = df_cot['count']
    percents = []
    for i in range(len(ten)):
        percent = (values[i]/(values.sum()))*100
        percents.append(float(percent))
    percents = sorted(percents,reverse=False)
    plt.pie(percents,labels=ten,autopct="%1.1f%%",colors=color_list[0:
 ↪len(ten)]);
    plt.legend(loc="center left",bbox_to_anchor=(1.1,0.5))
    plt.title(ten_bieu_do)

# xây dựng hàm vẽ biểu đồ cột tổng doanh thu theo từng loại
def ve_bieu_do_cot_tong_doanh_thu(ten_cot,ten_bieu_do):
```

```
    df_cot = data.groupby(ten_cot)["Price ($)"].sum().reset_index().
 ↪sort_values(by="Price ($)")
    ten = df_cot[ten_cot]
    values = df_cot['Price ($)']
    x_coordinate = [i for i in range(len(ten))]
    plt.figure(figsize=(20,5))
    sns.barplot(x=ten_cot,y="Price␣
 ↪($)",data=df_cot,palette=palettes[randint(0,len(palettes)-1)])
    plt.xticks(rotation=45,ha='right');
    plt.title("Biểu đồ tổng doanh thu theo Brand")
    for x,y in zip(x_coordinate,values):
        plt.text(x-0.25,y+100,str(round(y/1000000,2)) + " triệu đô")
    plt.title(ten_bieu_do)

# xây dựng hàm vẽ biểu đồ tròn thể hiện phần trăm tổng doanh thu theo loại
def ve_bieu_do_phan_tram_tong_doanh_so(ten_cot,ten_bieu_do):
    df_cot = data.groupby(ten_cot)["Price ($)"].sum().reset_index().
 ↪sort_values(by="Price ($)")
    ten = df_cot[ten_cot]
    values = df_cot['Price ($)']
    percents = []
    for i in range(len(ten)):
        percent = (values[i]/(values.sum()))*100
        percents.append(float(percent))
    percents = sorted(percents,reverse=False)
    plt.pie(percents,labels=ten,autopct="%1.1f%%",colors=color_list[0:
 ↪len(ten)]);
    plt.title(ten_bieu_do)
    plt.legend(loc="center left",bbox_to_anchor=(1.1,0.5))

# xây dựng hàm để vẽ biểu đồ cột với nhiều giá trị cung một lúc ( 1 x và nhiều␣
 ↪y)
def␣
 ↪ve_nhieu_bieu_do_cot(danh_sach_ten_cot,ten_bieu_do,ten_cot_can_ve_theo,ds_gia_tri_ve_theo_c
 ↪
    df_new = data[danh_sach_ten_cot]
    df_melted = df_new.melt(id_vars=[ten_cot_can_ve_theo], var_name="Loại",␣
 ↪value_name="Giá trị")
    df_melted = df_melted.sort_values(by='Giá trị')
    plt.figure(figsize=(20, 5))
    loai = df_melted["Loại"].value_counts().index.to_list()
    sns.barplot(x=ten_cot_can_ve_theo, y="Giá trị", hue="Loại",␣
 ↪data=df_melted,palette=palettes[randint(0,len(palettes)-1)],estimator=sum)
    plt.xticks(rotation=45,ha='right')
    plt.title(ten_bieu_do)
    ds_df_tra_ve = []
```

```
    for i in range(len(ds_gia_tri_ve_theo_cot)):
        loc_theo_loai = df_melted[df_melted["Loại"] ==  loai[i]]
        values_count = loc_theo_loai.groupby(ten_cot_can_ve_theo)['Giá trị'].
 ↪sum().reset_index().sort_values(by='Giá trị')
        ds_df_tra_ve.append(values_count)
    return ds_df_tra_ve

# xây dựng hàm để vẽ biểu đồ đường trong nhiều trường hợp
def ve_bieu_do_duong(danh_sach_ten_cot,ten_cot_can_ve_theo,ten_bieu_do,dk):
    if(dk == "%"):
        df_new = data[danh_sach_ten_cot]
        df_melted = df_new.melt(id_vars=[ten_cot_can_ve_theo], var_name="Loại",␣
 ↪value_name="Giá trị")
        plt.figure(figsize=(20, 5))
        df_melted
        loai = df_melted["Loại"].value_counts().index.to_list()
        ds_marker = ['o','p','s','D','X']
        ds_loai = []
        for i in range(2):
            df_loai = df_melted[df_melted['Loại'] == loai[i]]
            group_theo_giaTri = df_loai.groupby(ten_cot_can_ve_theo)["Giá trị"].
 ↪sum()
            ten = group_theo_giaTri.index.tolist()
            values = group_theo_giaTri.values.tolist()
            ds_loai.append(values)
            plt.plot(ten,values,marker=ds_marker[i],ms=10,label=loai[i])
        x = [i for i in range(len(ten))]
        percent = ((np.array(ds_loai[0]) - np.array(ds_loai[1]))/np.
 ↪array(ds_loai[1]))*100
        index=0
        for x,y in zip(x,percent):
            plt.text(x,values[index]+100,str(round(abs(y),2))+"%")
            index+=1
        plt.title(ten_bieu_do)
        plt.legend()
        print(loai)
        return ten,percent
    elif(dk=="1"):
        plt.figure(figsize=(20, 5))
        df_new = data[danh_sach_ten_cot]
        sns.lineplot(x=ten_cot_can_ve_theo,y=danh_sach_ten_cot[1],data=df_new)
        plt.title(ten_bieu_do)
        plt.legend()
    elif(int(dk) > 1):
        df_new = data[danh_sach_ten_cot]
        df_melted = df_new.melt(id_vars=[ten_cot_can_ve_theo], var_name="Loại",␣
 ↪value_name="Giá trị")
```

```
        plt.figure(figsize=(20, 5))
        df_melted
        loai = df_melted["Loại"].value_counts().index.to_list()
        sns.lineplot(x=ten_cot_can_ve_theo,y="Giá␣
 ↪trị",data=df_melted,hue='Loại',markers='D')
        plt.title(ten_bieu_do)

# Hàm vẽ biểu đồ boxplot
def ve_bieu_do_box_plot(danh_sach_ten_cot,ten_bieu_do,hue):
    df = data[danh_sach_ten_cot]
    plt.figure(figsize=(20,5))
    sns.
 ↪boxplot(x=df[danh_sach_ten_cot[0]],y=df[danh_sach_ten_cot[1]],data=df,hue=hue,palette=palet
    plt.title(ten_bieu_do)
    plt.xticks(rotation=45,ha='right')
    plt.legend()

# Hàm vẽ bểu đồ violin plot
def ve_bieu_do_violin_plot(danh_sach_ten_cot,ten_bieu_do,hue):
    df = data[danh_sach_ten_cot]
    plt.figure(figsize=(20,5))
    sns.
 ↪violinplot(x=df[danh_sach_ten_cot[0]],y=df[danh_sach_ten_cot[1]],data=df,hue=hue,palette=pa
    plt.title(ten_bieu_do)
    plt.xticks(rotation=45,ha='right')
    plt.legend()

# Hàm để vẽ biểu đồ poinplot
def ve_bieu_do_poinplot(danh_sach_ten_cot,ten_bieu_do,hue):
    df = data[danh_sach_ten_cot]
    means = df.groupby(danh_sach_ten_cot[0])[danh_sach_ten_cot[1]].mean().
 ↪sort_values()
    order = means.sort_values().index
    plt.figure(figsize=(20,5))
    sns.
 ↪pointplot(x=danh_sach_ten_cot[0],y=danh_sach_ten_cot[1],data=df,hue=hue,␣
 ↪capsize=0.2,order=order)
    for ten,mean in enumerate(means):
        plt.text(ten,mean,round(mean,2))
    plt.title(ten_bieu_do)
    print(means)
```

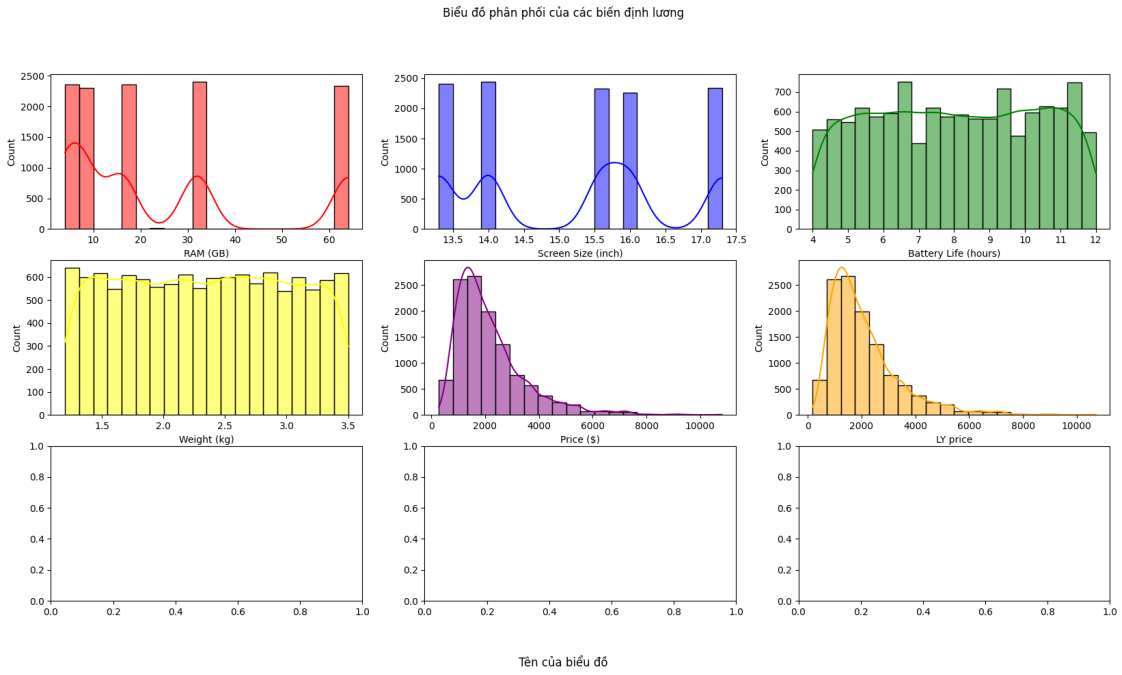
- Khám phá dữ liệu định lượng

[15]:
```
# lấy ra các cột có kiểu dữ liệu là int và float
cot_dinh_luong = data.select_dtypes(include=['int','float']).columns.to_list()
cot_dinh_luong
```

[15]:
```
['RAM (GB)',
 'Screen Size (inch)',
 'Battery Life (hours)',
 'Weight (kg)',
 'Price ($)',
 'LY price']
```

[16]:
```
# vẽ biểu đồ phân phối của các biến định lượng
ve_bieu_do_phan_phoi(3,3,cot_dinh_luong,"Tên của biểu đồ","Biểu đồ phân phối␣
 ↪của các biến định lương")
```

```
1
1
1
```



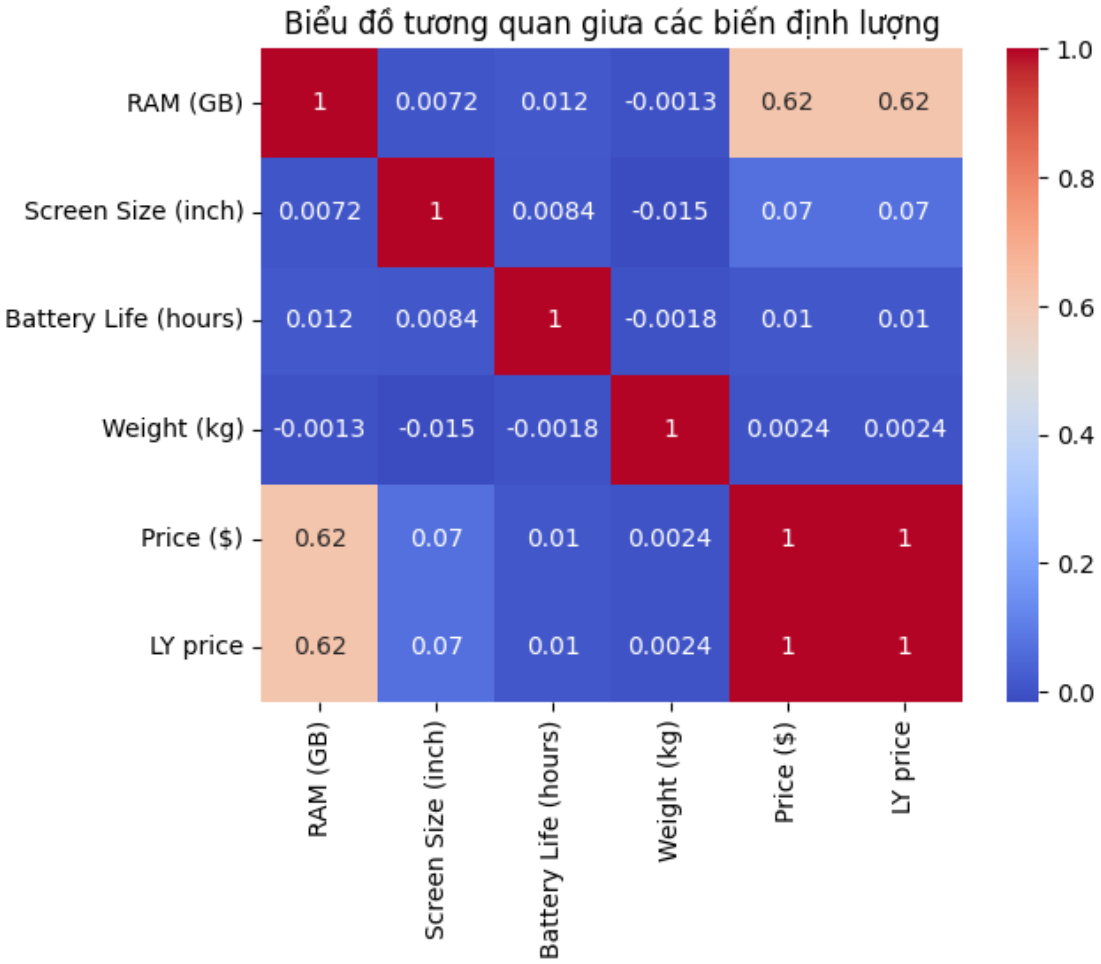
Phân tích biểu đồ phân phối của các biến numeric - *Các biến numeric bao gôm 'RAM (GB)','Screen Size (inch)','Battery Life (hours)','Weight (kg)','Price ($)','LY price' - Phân tích:*

```
+ ta thấy đôi với hai biến đầu tiên là RAM và screen size thì biểu đồ phân phối là biểu đồ
đối với ram và 13,14,16,17 đối với screen size , nhưng trên thực tế ta thấy rằng đối với ra
số mốc nhất định như ở trên vì vậy khi chúng ta chỉ quan sát đến những điểm đó ta thấy dữ l
không chỗ nào thấp quá
====>> : ta thấy rằng nhu cầu sử dụng của khách hàng với screen size và ram đều khá đều , l
đa phân họ sẽ chọn theo nhu cầu sử dụng của bản thân
+ đối với weight và battery thì thay thấy biểu đồ phân phối của chúng là biểu đồ phân phối
là thông tin không có giá trị vì trên thực tế thì battery còn dựa vào thời gian sử dụng của
```

=====>> : trên thực tế người ta thường không quan tâm đến những thông tin này khi mua máy
+ LY price và price : thấy thấy hai biểu đồ phân phối của hai feature này đều là hai biểu
=====>> : nhu cầu sử dụng máy tính và giá của năm 2024 so với năm 2023 không thay đổi , kha
giá tầm vừa và rẻ , khôn nhiều khách hàng chịu bỏ số tiền lớn để mua lap top , Ta thấy được

[17]:
```
# biểu đồ heapmap tương quan của các biến định lượng
sns.heatmap(data.select_dtypes(include=['int','float']).
  ↪corr(),annot=True,cmap='coolwarm')
plt.title("Biểu đồ tương quan giưa các biến định lượng")
```

[17]: Text(0.5, 1.0, 'Biểu đồ tương quan giưa các biến định lượng')



Phân tích biểu đồ tương quan giữa các biến định lượng (Phục cho cho Machine learning và AI)

- Ta thấy biểu đồ heapmap có sự tương quan rất là thấp đa phần chỉ là 0.06 .. , mức tương quan cao nhất cũng chỉ là 0.62

- từ đó ta thấy dữ liệu sẽ phù hợp hơn với những model phi tuyến như SVM , randomforest , ...

[18]:
```
# sns.pairplot(data,hue='Brand')
```

[19]:
```
# lấy ra các thống kê cơ bản
thong_ke = data.describe()
thong_ke
```

[19]:

	RAM (GB)	Screen Size (inch)	Battery Life (hours)	Weight (kg)	\
count	11771.000000	11771.000000	11771.000000	11771.000000	
mean	24.842069	15.212837	8.027569	2.341144	
std	21.748492	1.437200	2.304795	0.667838	
min	4.000000	13.300000	4.000000	1.200000	
25%	8.000000	14.000000	6.000000	1.760000	
50%	16.000000	15.600000	8.000000	2.340000	
75%	32.000000	16.000000	10.000000	2.910000	
max	64.000000	17.300000	12.000000	3.500000	

	Price ($)	LY price
count	11771.000000	11771.000000
mean	2183.417665	2083.417665
std	1316.753596	1316.753596
min	279.570000	179.570000
25%	1272.185000	1172.185000
50%	1839.970000	1739.970000
75%	2698.015000	2598.015000
max	10807.880000	10707.880000

Phân tích thống kê mô tả (Descriptive Statistics)

Dưới đây là bảng thống kê mô tả cho tập dữ liệu laptop với 6 thuộc tính quan trọng:

Thuộc tính	Trung bình	Độ lệch chuẩn	Min	25%	50% (Median)	75%	Max
RAM (GB)	24.84	21.75	4.0	8.0	16.0	32.0	64.0
Screen Size (inch)	15.21	1.44	13.3	14.0	15.6	16.0	17.3
Battery Life (hrs)	8.03	2.30	4.0	6.0	8.0	10.0	12.0
Weight (kg)	2.34	0.67	1.2	1.76	2.34	2.91	3.5
Price () ∗	2083.42	1316.75	179.6	1172.2	1739.97	2598.0	10707.9

∗|2183.42|1316.75|279.6|1272.2|1839.97|2698.0|10807.9||∗
∗$LY Price()$

Phân tích dạng phân phối dữ liệu

- **RAM, Price, LY Price** có độ lệch lớn giữa mean và median, và max rất cao → phân phối **lệch phải** (right-skewed). Dữ liệu có thể chứa nhiều giá trị ngoại lai (outliers).

- **Screen Size** có mean median và std nhỏ → có xu hướng gần **phân phối chuẩn**.
- **Battery Life** và **Weight** cũng có phân phối tương đối cân đối, với độ lệch chuẩn thấp → có thể gần **phân phối chuẩn**.

Gợi ý: nên trực quan hóa với biểu đồ Histogram hoặc Boxplot để kiểm tra trực tiếp dạng phân phối.

Mức độ phân tán (Coefficient of Variation)

Hệ số biến thiên (CV = std / mean) giúp đánh giá mức độ phân tán tương đối:

Thuộc tính	CV	Đánh giá
RAM	0.87	Phân tán mạnh
Screen Size	0.095	Rất đồng đều
Battery Life	0.29	Phân tán trung bình
Weight	0.29	Phân tán trung bình
Price	0.60	Phân tán mạnh
LY Price	0.63	Phân tán mạnh

So sánh giữa các cột

- **RAM** và **Price**: Có khả năng tương quan dương. Máy RAM cao thường đi kèm giá cao.
- **Battery Life** và **Weight**: Có thể có quan hệ nghịch chiều – thiết bị nhẹ có thể ít pin hơn.
- **Price** và **LY Price**: Hai cột giá gần nhau và rất có thể có tương quan tuyến tính mạnh.
- **Screen Size** và **Weight**: Có thể màn hình to → máy nặng hơn. Nên kiểm tra qua biểu đồ scatter.

Gợi ý: - Dùng biểu đồ scatter để kiểm tra tương quan giữa các cặp biến liên tục. - Có thể tạo ma trận tương quan bằng `.corr()` để xem tổng quan các mối quan hệ.

Kết luận

- Các thuộc tính như **RAM**, **Price**, **LY Price** có độ phân tán lớn, chứa nhiều outlier → cần xử lý trước khi modeling.
- **Screen Size**, **Battery Life**, **Weight** có phân phối khá ổn định, phù hợp để đưa vào mô hình mà không cần biến đổi phức tạp.
- Nên tiếp tục phân tích tương quan và trực quan hóa để phát hiện mối quan hệ tiềm năng giữa các biến.

- Khám phá dữ liệu định tính

[20]:
```
# lấy ra các cột có kiểu dữ liệu oject
cot_dinh_tinh = data.select_dtypes(include=['O']).columns.to_list()
cot_dinh_tinh
```

[20]:
```
['Brand',
 'Processor',
 'Storage',
 'GPU',
 'Resolution',
 'Operating System',
 'months']
```

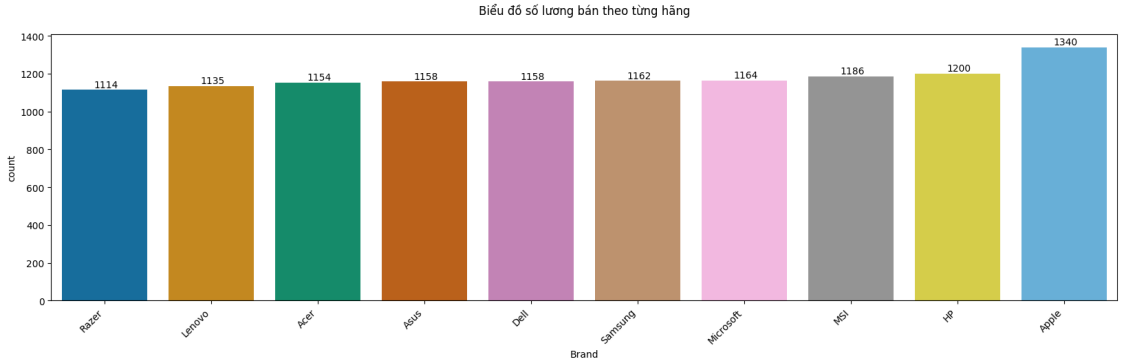
[21]:
```
data.head()
```

[21]:

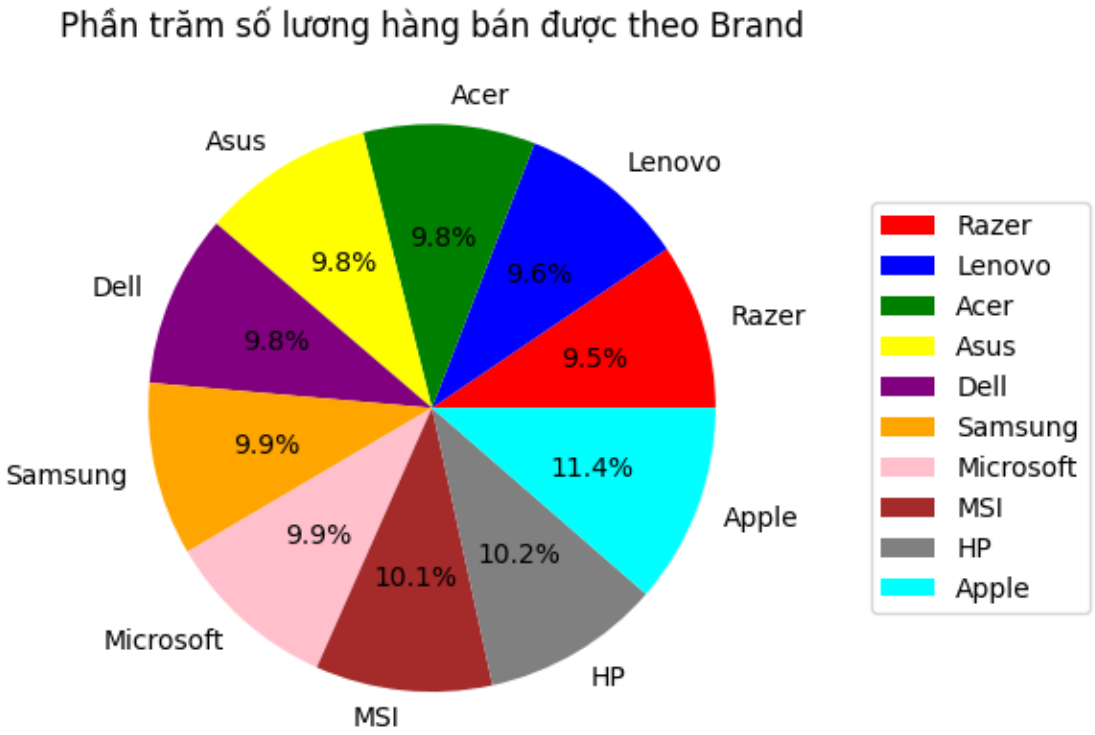
	Brand	Processor	RAM (GB)	Storage	GPU	Screen Size (inch)	Resolution	Battery Life (hours)	Weight (kg)	Operating System	Price ($)	LY price	months
0	Apple	AMD Ryzen 3	64	512GB SSD	Nvidia GTX 1650	17.3	2560x1440	8.9	1.42	FreeDOS	3997.07	3897.07	October
1	Razer	AMD Ryzen 7	4	1TB SSD	Nvidia RTX 3080	14.0	1366x768	9.4	2.57	Linux	1355.78	1255.78	December
2	Asus	Intel i5	32	2TB SSD	Nvidia RTX 3060	13.3	3840x2160	8.5	1.74	FreeDOS	2673.07	2573.07	October
3	Lenovo	Intel i5	4	256GB SSD	Nvidia RTX 3080	13.3	1366x768	10.5	3.10	Windows	751.17	651.17	August
4	Razer	Intel i3	4	256GB SSD	AMD Radeon RX 6600	16.0	3840x2160	5.7	3.38	Linux	2059.83	1959.83	October

[22]:
```
# vẽ biểu đồ sản lượng bán hàng theo Brand
ve_bieu_do_cot_tong_so_luon_ban("Brand","Biểu đồ số lương bán theo từng hãng")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:39:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y='count',data=df_cot,palette=palettes[randint(0,len(pal
ettes)-1)])
```



[23]:
```
# vẽ biểu đồ tròn thể hiện phần trăm số lượng hàng bán được theo Brand
ve_bieu_do_tron("Brand",'Phần trăm số lương hàng bán được theo Brand')
```

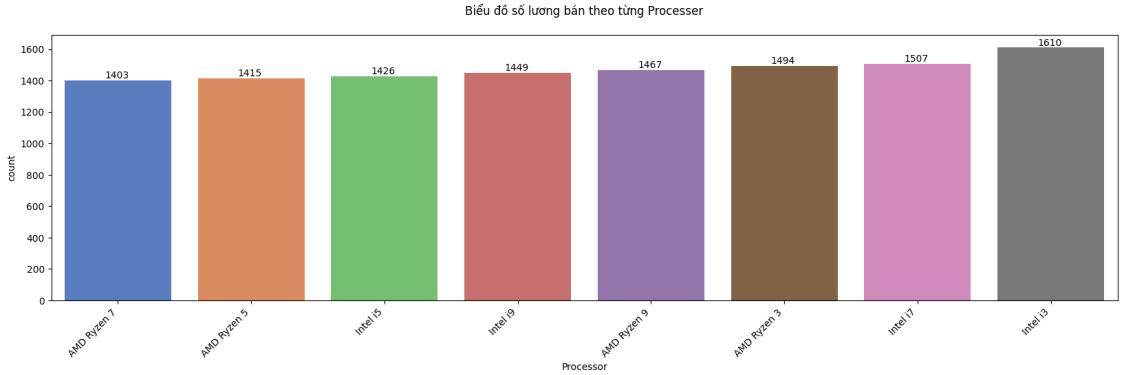


Sản lượng bán máy tính theo Brand - *Ta thấy sản lượng bán máy tính của hãng Apple cao hơn so với các hãng khác 1340 sản phẩm và chiếm 11.4% tổng số máy tính chúng ta bán , mặc dù trên thực tế giá thành của hãng Apple cáo hơn các hãng khác khá nhiều từ đó ta thấy được sự uy tính của Brand cũng như sự tin tưởng của khác hàng đối với Brand này*

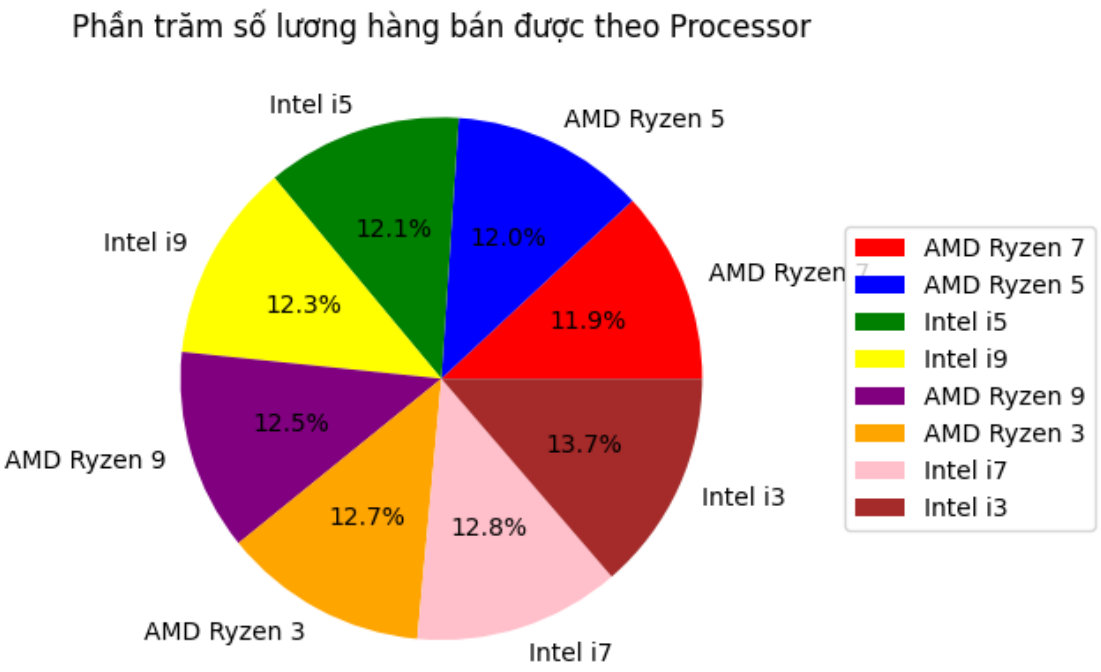
===» : xu hướng khác hàng thích sử dụng apple hơn so với những hãng máy tính khác

[24]:
```
# biểu đồ số lương máy tính bán được theo Processer
ve_bieu_do_cot_tong_so_luon_ban("Processor","Biểu đồ số lương bán theo từng␣
↪Processer")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:39:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y='count',data=df_cot,palette=palettes[randint(0,len(pal
ettes)-1)])
```



[25]:
```
# vẽ biểu đồ tròn thể hiện phần trăm số lượng hàng bán được theo Processor
ve_bieu_do_tron("Processor","Phần trăm số lương hàng bán được theo Processor")
```



Sản lượng bán máy tính theo Processer - *Ta thấy sản lượng bán máy tính của theo intel3 cao hơn so với các Precesser khác 1610 sản phẩm và chiếm 13.4% tổng số máy tính chúng ta bán, điều này xảy ra là vì giá trị trung bình của một chiếc máy với bộ xử lý intel 3 rẻ hơn các bộ xử lý khác rất nhiều phù hợp với túi tiền của hầu hết người sử dụng , học sinh , sinh viên , và với các tác vụ làm văn phòng bình thường như kế toán kiểm toán ,.. thì intel 3 có thể làm tốt vì vậy intel 3 vẫn là lựa chọn ưa chuộng của đa số khách hàng bình thường*

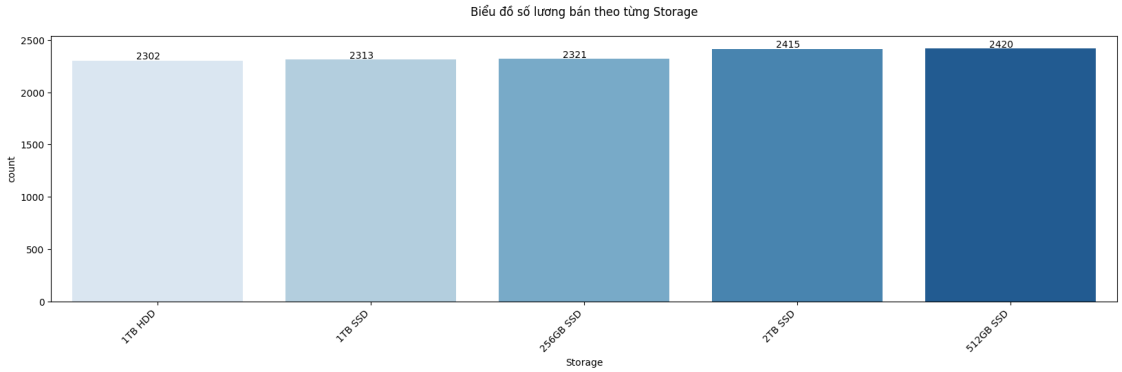
- Ngược lại chúng ta thấy rằng những processer như AMR7 và AMDR5 là hai sản phẩm đứng thấp nhất sản lượng bán lần lượt là 1403 và 1415 chiếm 11.9% và 12% tổng sản lượng bán vì sao lại có điều này , về cơ bản thì chip AMD chỉ phù hợp với tiểu số khách hàng đó là những người tập trung vào tác vụ xử lý đồ họa như game deverloper , streamer , thiết kế đồ họa kiến trúc , ... và cộng tác AMD cũng là dòng sản phẩm đắt hơn so với intel vì vậy không được đa số khách hàng tin dùng - Có một điều khá đặc biệt là intel 5 lại là sản phẩm bán được ít thứ 3 chỉ sau AMD 7 và 5 mặc dù trên thức tế intel 5 là sản phẩm được rất nhiều khách hàng quan tâm vì sự đa dụng là xử lý tất cả các tác vụ ở mức rất ổn định của nó(Cần kiểm chứng)

===» : xu hướng khách hàng thích sử dụng vẫn ưa chuộng những máy có giá thành rẻ và làm tốt các tác vụ văn phòng

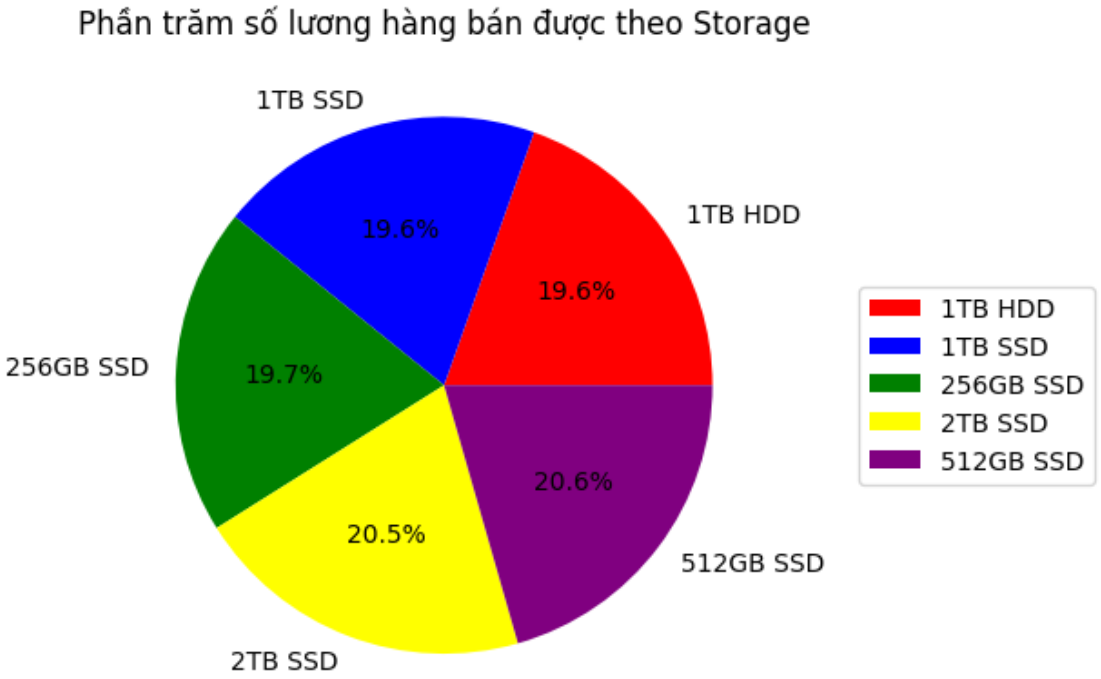
[26]:
```
# Biểu đồ số lượng bán máy tính theo dung lương bộ nhớ
ve_bieu_do_cot_tong_so_luon_ban("Storage","Biểu đồ số lương bán theo từng␣
↪Storage")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:39:
FutureWarning:
```

```
Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y='count',data=df_cot,palette=palettes[randint(0,len(pal
ettes)-1)])
```



[27]:
```
# vẽ biểu đồ tròn thể hiện phần trăm số lượng hàng bán được theo Storage
ve_bieu_do_tron("Storage","Phần trăm số lương hàng bán được theo Storage")
```

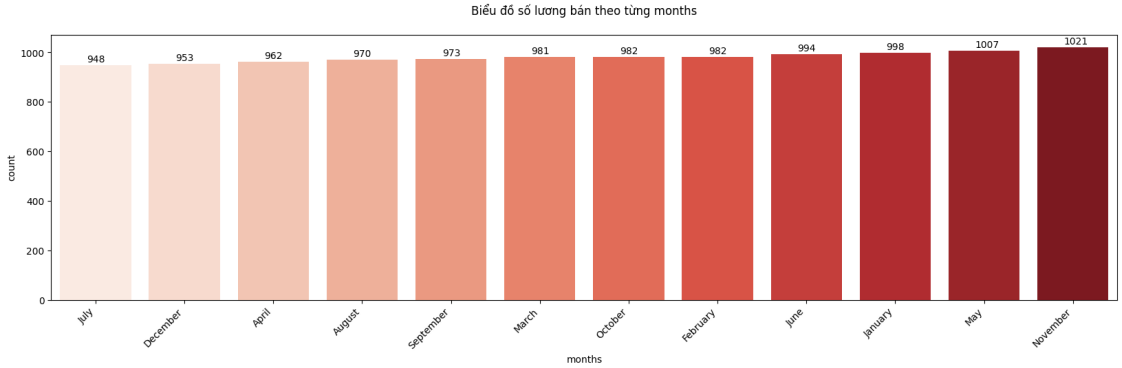


Sản lượng bán máy tính theo Storge - *Nhìn chung thì ta có thể thấy rằng dung lượng ổ cứng không có một sản phẩm nào quá được ưa chuộng đa phần khác hàng sẽ xử dụng theo nhu cầu bản thân nếu cần thì sẽ tăng len sau , nhưng chúng ta có thể thấy rõ rằng sản phẩm ổ cứng về HHD đang bị khác hàng bỏ qua khá nhiều bằng chứng là qua ba năm 2022,2023,2024 doanh số của ổ cứng HHD đều giảm và đứng cuối cùng , riêng năm 2024 HHD bán 2302 sản phẩm chiếm 19.6% tổng sản lượng bán , thứ 2 các hãng brand cũng đã giảm sản lượng sản xuất HHD mà thay vào đó sản xuất tăng cường SSD và đa dạng hóa chúng*

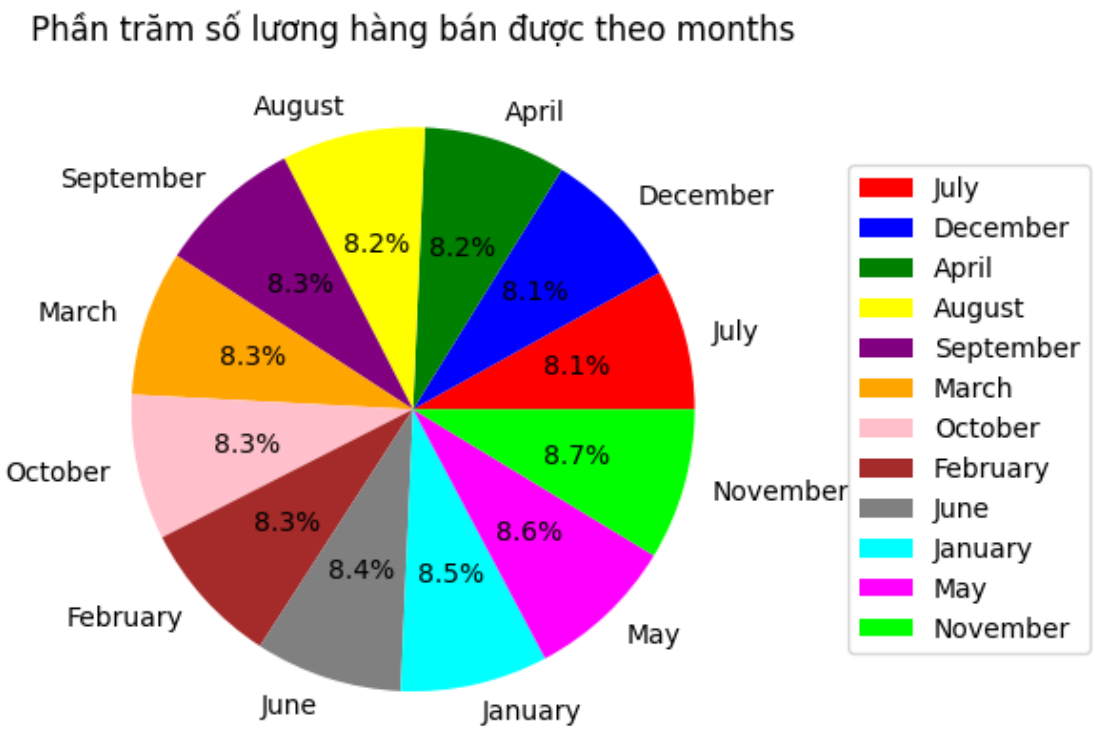
===» : xu hướng khác hàng thích sử và đang đân ưa chuộng SSD hơn vì sự hiện đại của chúng

[28]:
```
# sản lượng máy tính bán được theo từng tháng trong năm
ve_bieu_do_cot_tong_so_luon_ban("months",'Biểu đồ số lương bán theo từng␣
↪months')
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:39:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y='count',data=df_cot,palette=palettes[randint(0,len(pal
ettes)-1)])
```

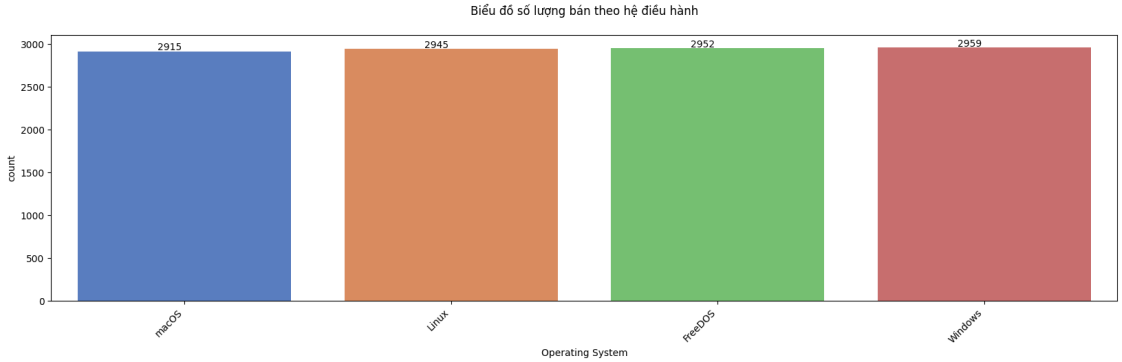


[29]:
```
# vẽ biểu đồ tròn thể hiện phần trăm số lượng hàng bán được theo months
ve_bieu_do_tron("months","Phần trăm số lương hàng bán được theo months")
```

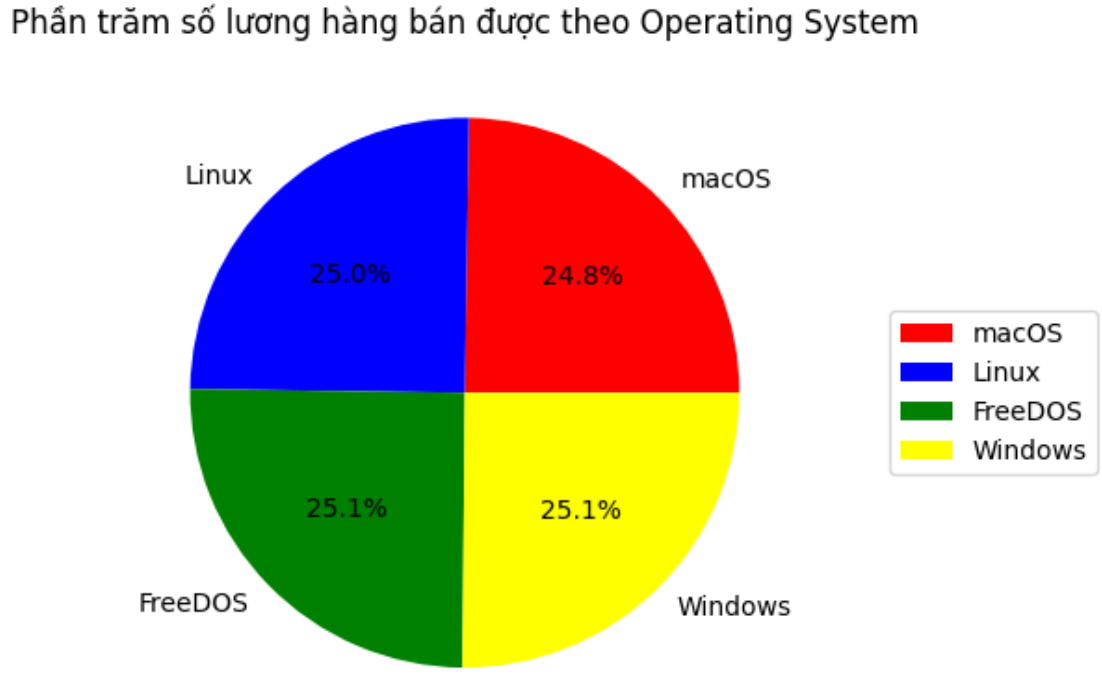


[30]:
```
# số lương máy tính bán được theo hệ điều hành
ve_bieu_do_cot_tong_so_luon_ban("Operating System","Biểu đồ số lượng bán theo␣
 ↪hệ điều hành")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:39:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y='count',data=df_cot,palette=palettes[randint(0,len(pal
ettes)-1)])
```



[31]:
```
# vẽ biểu đồ tròn thể hiện phần trăm số lượng hàng bán được theo Operating System
ve_bieu_do_tron("Operating System","Phần trăm số lương hàng bán được theo Operating System")
```



Sản lượng bán máy tính theo months và Operating System - *về months thì không có một quy luật nào cả đa phần khác hàng sẽ mua máy tính vào bất kì một ngày nào trong năm vd có thể ngày sinh nhật , ngày lãnh lưỡng - một điểm khá bất ngờ là về sự cân bằng của các Operating System , tưởng trừng như window sẽ là sản phẩm được ưa chuộng nhất , nhưng không phải sản lượng bán của window chỉ ngang bằng một sản phẩm khá ít người biết đến là freedos Nhưng trên thực thì không*

phải như vậy bản chất Freedos chỉ là một mã nguồn mở được viết bởi một kỹ sư IT khi microsoft không hỗ trợ MSDOS cho cong ty anh ta nữa vì vậy việc FreeDos cao chỉ vì chiêu trò thương mại của các hãng máy tính , họ sẽ cài FreeDos mã nguồn mở để sau đó người sử dụng từ cài windows lậu hoặc người dùng tự có bản quyền windows vì vậy trên thực tế thì tất cả khách hàng đều quan và có xu hướng sử dụng windows

===» : xu hướng khác hàng thích sử và trung thành với Operating System window

- Khám phá mối quan hệ giữa biến định tính và biến định lượng

[32]:
```
data.head()
```

[32]:

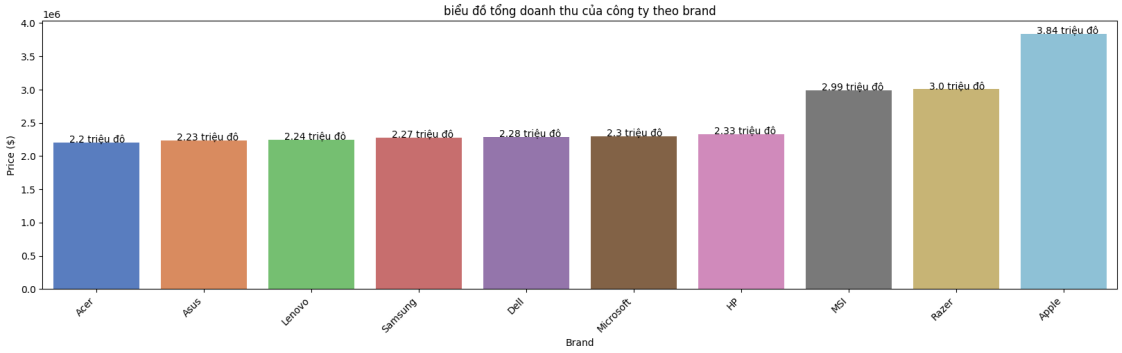
	Brand	Processor	RAM (GB)	Storage	GPU	\
0	Apple	AMD Ryzen 3	64	512GB SSD	Nvidia GTX 1650	
1	Razer	AMD Ryzen 7	4	1TB SSD	Nvidia RTX 3080	
2	Asus	Intel i5	32	2TB SSD	Nvidia RTX 3060	
3	Lenovo	Intel i5	4	256GB SSD	Nvidia RTX 3080	
4	Razer	Intel i3	4	256GB SSD	AMD Radeon RX 6600	

	Screen Size (inch)	Resolution	Battery Life (hours)	Weight (kg)	\
0	17.3	2560x1440	8.9	1.42	
1	14.0	1366x768	9.4	2.57	
2	13.3	3840x2160	8.5	1.74	
3	13.3	1366x768	10.5	3.10	
4	16.0	3840x2160	5.7	3.38	

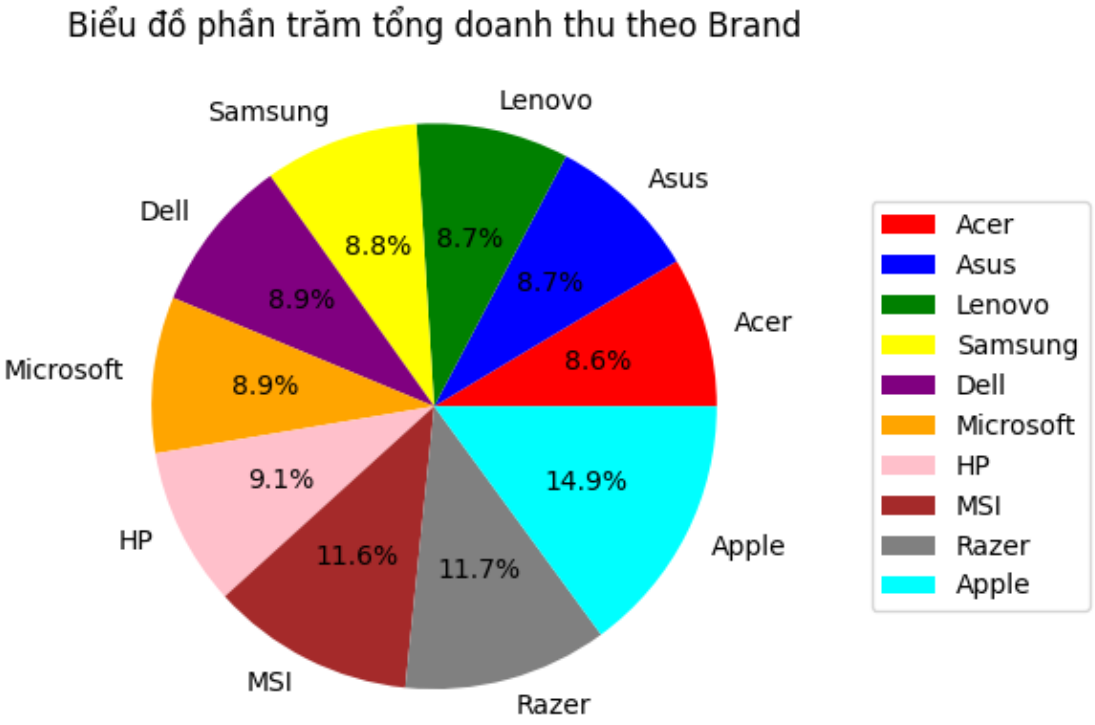
	Operating System	Price ($)	LY price	months
0	FreeDOS	3997.07	3897.07	October
1	Linux	1355.78	1255.78	December
2	FreeDOS	2673.07	2573.07	October
3	Windows	751.17	651.17	August
4	Linux	2059.83	1959.83	October

[33]:
```
# vẽ biểu đồ cột tổng doanh thu theo Brand
ve_bieu_do_cot_tong_doanh_thu("Brand","biểu đồ tổng doanh thu của công ty theo␣
↪brand")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:66:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y="Price
($)",data=df_cot,palette=palettes[randint(0,len(palettes)-1)])
```



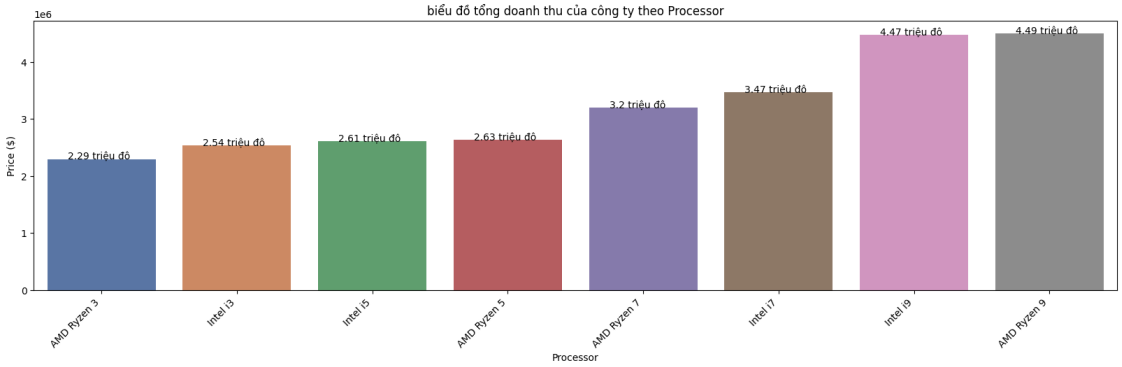
[34]:
```
# vẽ biểu đồ trong thể hiện phần trăm tổng doanh số theo Brand
ve_bieu_do_phan_tram_tong_doanh_so("Brand","Biểu đồ phần trăm tổng doanh thu␣
↪theo Brand")
```



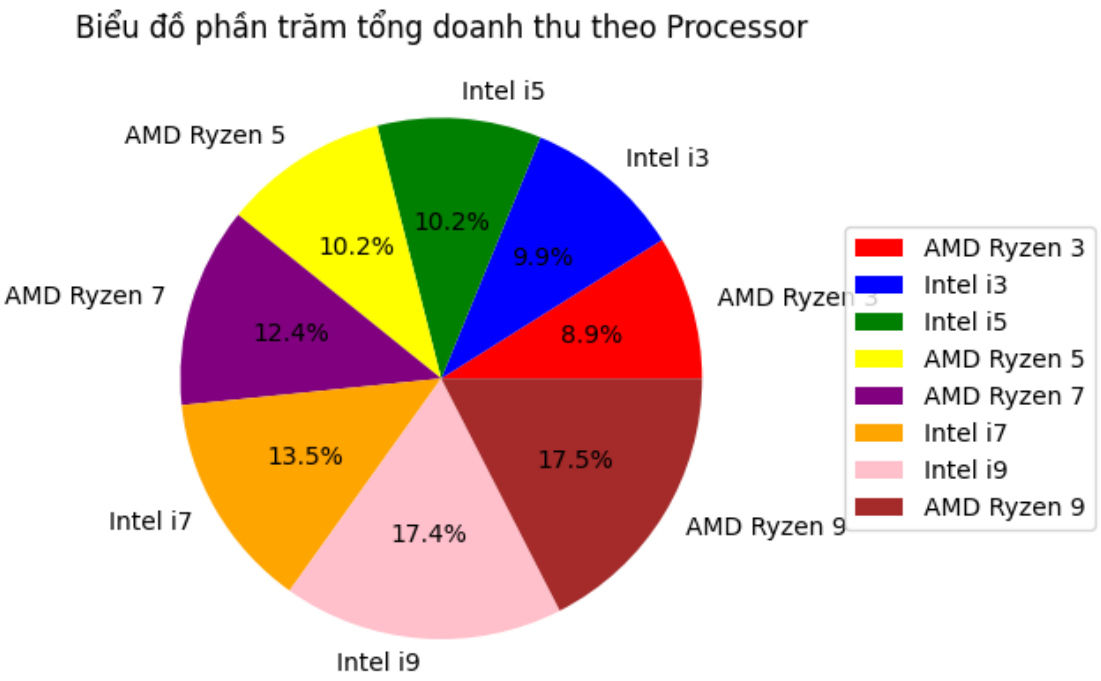
Tổng doanh số bán máy tính theo Brand - *Ta thấy rằng top 3 hãng mang lại doanh thu cao nhất lần lượt là Apple , razer , MSI , 3.84tr$, 3tr$ và 2.99tr$ chiếm ~15% , 11.7% và 11.6% , mặc dù về sản lượng bán MSI chỉ đứng thứ 3 và razer thì đứng cuối cùng nhưng doanh số của razer mang lại cao thứ 2 chỉ sau Apple từ đấy ta có thể khẳng định được giả thuyết đã đưa ra ở trên về AMD đó là giá thành của AMD cao từ đó ta thấy được rằng lươn nhuận AMD mang lại cũng sẽ là cao*

[35]:
```
# vẽ biểu đồ cột tổng doanh thu theo Processor
ve_bieu_do_cot_tong_doanh_thu("Processor","biểu đồ tổng doanh thu của công ty␣
↪theo Processor")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:66:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y="Price
($)",data=df_cot,palette=palettes[randint(0,len(palettes)-1)])
```



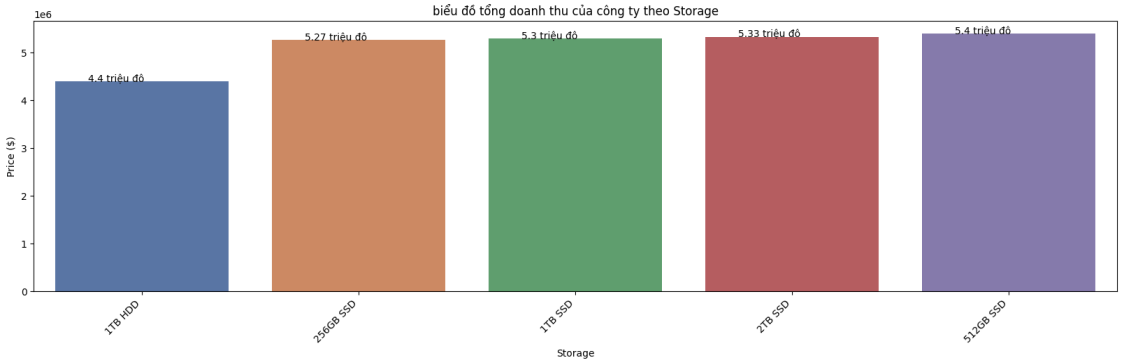
[36]:
```
# vẽ biểu đồ trong thể hiện phần trăm tổng doanh số theo Processor
ve_bieu_do_phan_tram_tong_doanh_so("Processor","Biểu đồ phần trăm tổng doanh␣
↪thu theo Processor")
```



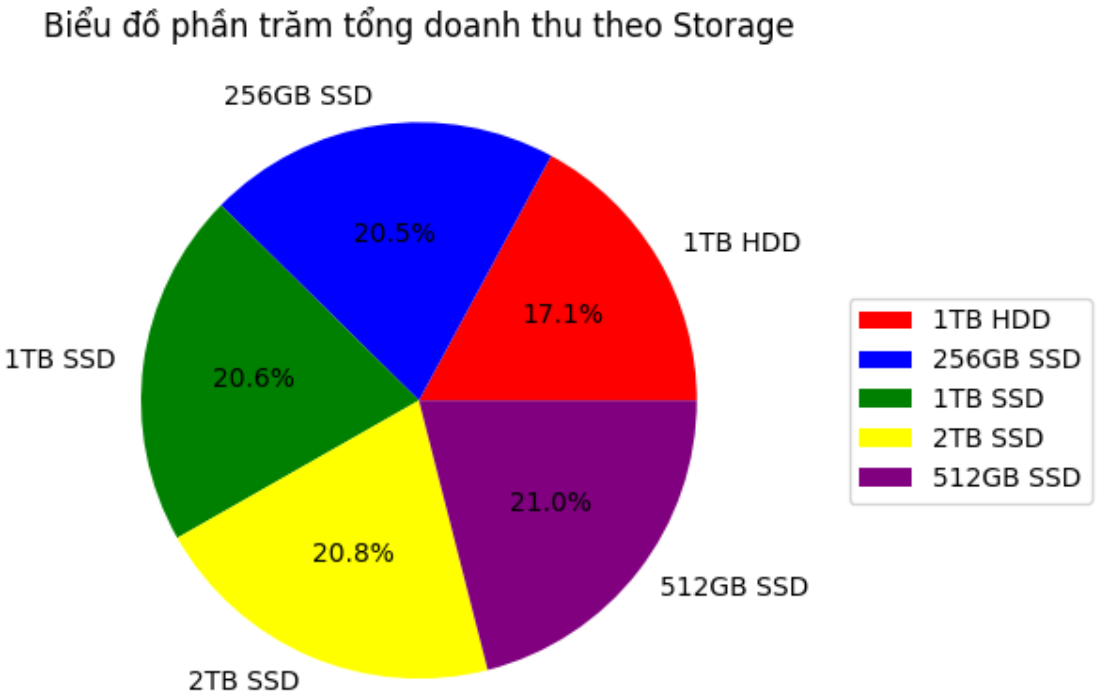
Tổng doanh số bán máy tính theo Processer - *Tương tự như phần Brand Ta thấy rằng top 3 hãng mang lại doanh thu cao nhất lần lượt là AMD9 , intel9 , intel7 , 4.49tr$, 4.47tr$ và 3.47tr$ chiếm ~17.5% , 17.4% và 13.5% , mặc dù về sản lượng bán AMD9 đứng cuối cùng còn intel9 thì đứng thứ 5*

[37]:
```
# vẽ biểu đồ cột tổng doanh thu theo Storage
ve_bieu_do_cot_tong_doanh_thu("Storage","biểu đồ tổng doanh thu của công ty␣
↪theo Storage")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:66:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y="Price
($)",data=df_cot,palette=palettes[randint(0,len(palettes)-1)])
```



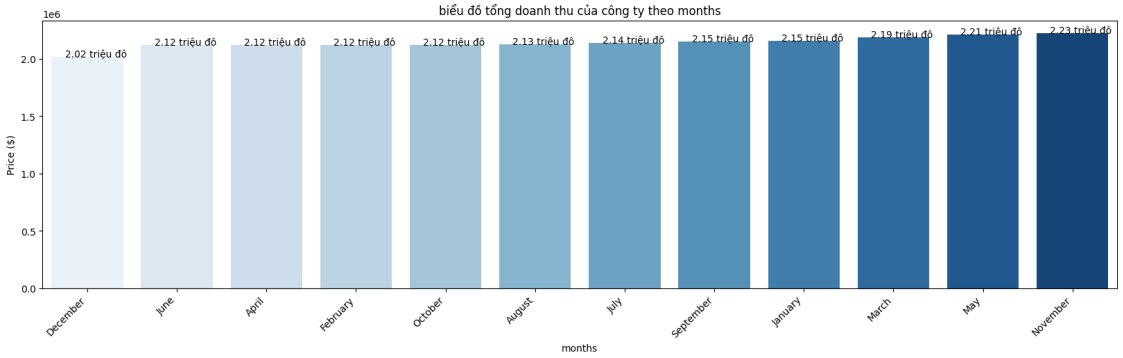
[38]:
```
# vẽ biểu đồ trong thể hiện phần trăm tổng doanh số theo Storage
ve_bieu_do_phan_tram_tong_doanh_so("Storage","Biểu đồ phần trăm tổng doanh thu␣
↪theo Storage")
```



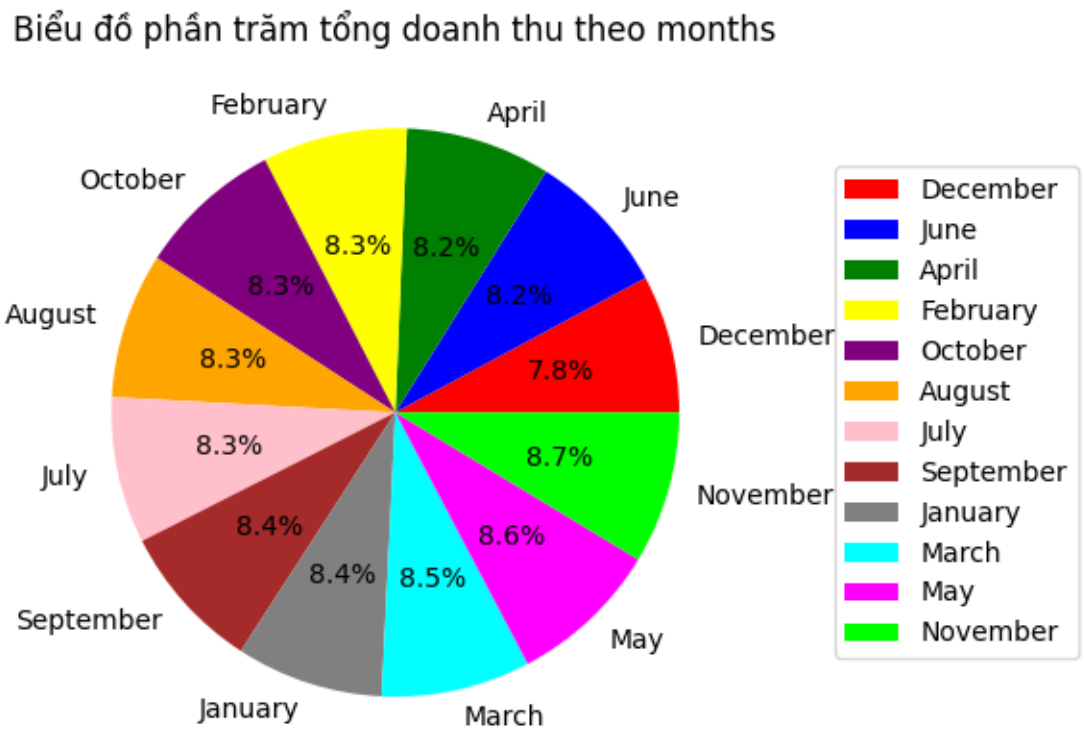
Tổng doanh số bán máy tính theo storage - *ta có thể thấy rằng doanh số mà các ổ cứng SSD mang lại khá khá cân bằng nhau tuy nhiên chúng ta có thể thấy rằng HHD đang không được ưa chuộng cả về sản lượng lẫn giá thành , doanh số mà HHD mang lại rất thấp chỉ chiếm có ~17% trong khi các SSD thì đều trên 20%*

[39]:
```
# vẽ biểu đồ cột tổng doanh thu theo months
ve_bieu_do_cot_tong_doanh_thu("months","biểu đồ tổng doanh thu của công ty theo␣
 ↪months")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:66:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=ten_cot,y="Price
($)",data=df_cot,palette=palettes[randint(0,len(palettes)-1)])
```



[40]:
```
# vẽ biểu đồ tròn thể hiện phần trăm tổng doanh số theo months
ve_bieu_do_phan_tram_tong_doanh_so("months","Biểu đồ phần trăm tổng doanh thu␣
 ↪theo months")
```



Kết Luận - *Ta thấy rằng về xu hướng thì khách hàng vẫn có xu hướng thích những máy tính có cấu hình thấp và giá thành rẻ nhưng trên thực tế doanh số là lợi nhuận của công ty đến chủ yếu từ những sản phẩm chất lượng có giá thành từ tầm trung đến cao - ==> nên tập trung quảng cáo những sản phẩm chất lượng cao , giá thành cao vì đây là những sản phẩm mang lại doanh thu và lợi nhuận cao cho tông ty nhưng lại không được khách hàng tin dùng và mua nhiều vd như hãng MSI razer,.., không cần tập trung quảng quá quá nhiều về những sản phẩm phổ thông như hãng Apple, processer intel 3 tại vì nó đã quá phổ biến và nổi tiếng*

[41]:
```
data.head()
```

[41]:

	Brand	Processor	RAM (GB)	Storage	GPU	\
0	Apple	AMD Ryzen 3	64	512GB SSD	Nvidia GTX 1650	
1	Razer	AMD Ryzen 7	4	1TB SSD	Nvidia RTX 3080	
2	Asus	Intel i5	32	2TB SSD	Nvidia RTX 3060	
3	Lenovo	Intel i5	4	256GB SSD	Nvidia RTX 3080	
4	Razer	Intel i3	4	256GB SSD	AMD Radeon RX 6600	

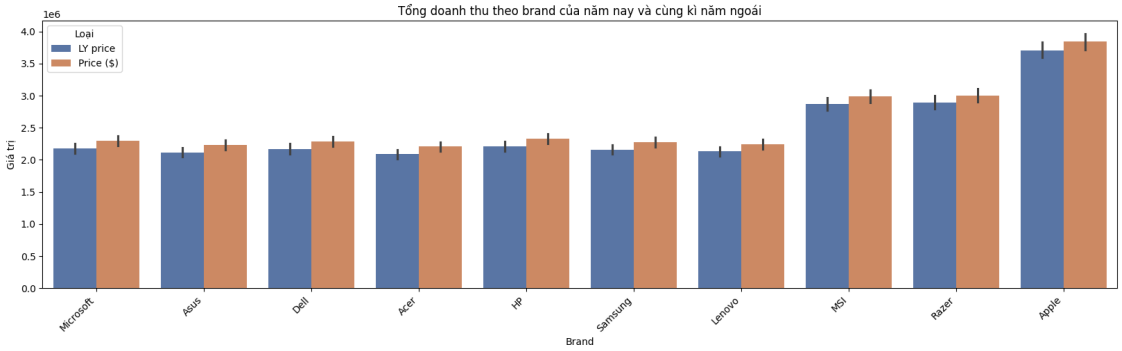
	Screen Size (inch)	Resolution	Battery Life (hours)	Weight (kg)	\
0	17.3	2560x1440	8.9	1.42	
1	14.0	1366x768	9.4	2.57	
2	13.3	3840x2160	8.5	1.74	
3	13.3	1366x768	10.5	3.10	

```
4                16.0  3840x2160                    5.7         3.38
```

```
  Operating System  Price ($)  LY price    months
0          FreeDOS    3997.07   3897.07   October
1            Linux    1355.78   1255.78  December
2          FreeDOS    2673.07   2573.07   October
3          Windows     751.17    651.17    August
4            Linux    2059.83   1959.83   October
```

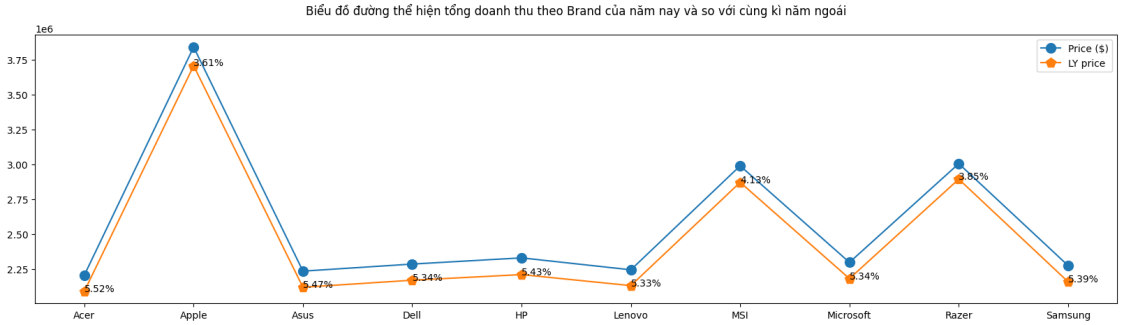
[42]:
```
# vẽ biểu đồ cột với giá trị là tổng doanh thu của năm nay so sánh với cùng kì␣
 ↪năm ngoái theo Brand
danh_sach_ten_cot = ['Brand','Price ($)','LY price']
ds_gia_tri_ve_theo_cot = ['Price ($)','LY price']
ds_df_tr_ve = ve_nhieu_bieu_do_cot(danh_sach_ten_cot,"Tổng doanh thu theo brand␣
 ↪của năm nay và cùng kì năm ngoái",'Brand',ds_gia_tri_ve_theo_cot)
print(ds_df_tr_ve[0])
```

```
      Brand     Giá trị
0      Acer  2089433.76
2      Asus  2118064.01
5    Lenovo  2129834.59
9   Samsung  2157571.36
3      Dell  2168755.68
7  Microsoft  2180134.26
4        HP  2208962.22
6       MSI  2870710.10
8     Razer  2893492.35
1     Apple  3706951.00
```



[43]:
```
# vẽ biểu đồ đường thể hiện độ tăng trưởng của tổng doanh thu theo Brand
a = ve_bieu_do_duong(danh_sach_ten_cot,"Brand","Biểu đồ đường thể hiện tổng␣
 ↪doanh thu theo Brand của năm nay và so với cùng kì năm ngoái\n","%")
```

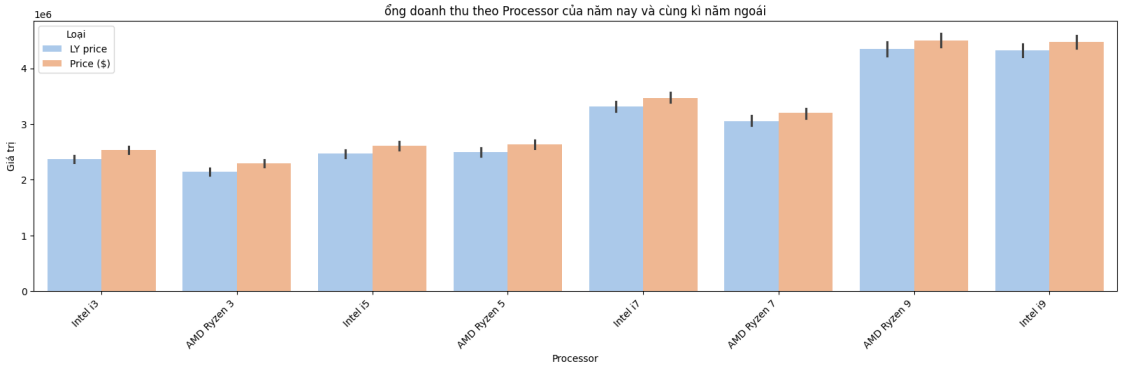
```
['Price ($)', 'LY price']
```



Tổng doanh số bán máy tính theo storage - *Ta thấy doanh sô tăng trưởng cao nhất là Acer 5.52% điều này xảy ra là vì những năm gần đây các sản phẩm đến từ đài loan nơi có trụ sở của công ty chip hàng đầu NVIDIA đặt tại đó sản xuất chip vô cùng phổ biến vì giá thành hợp lý , chip hoạt động tốt phù hợp với lứa sinh viên , học sinh - Tăng trưởng thấp nhất thuộc về Apple 3.5% vì cơ bản sản phẩm này đã uy tín và bán được nhiều hàng từ rất lâu về trước vậy nên việc tăng trưởng thấp là dễ hiểu , apple tập trung vào việc dữ ổn định tập khách hàng hơn là giới thiệu khác hàng mới*

[44]:
```
# vẽ biểu đồ cột với giá trị là tổng doanh thu của năm nay so sánh với cùng kì␣
↪năm ngoái theo Processor
danh_sach_ten_cot = ['Processor','Price ($)','LY price']
ds_gia_tri_ve_theo_cot = ['Price ($)','LY price']
ds_df_tr_ve = ve_nhieu_bieu_do_cot(danh_sach_ten_cot,"ổng doanh thu theo␣
↪Processor của năm nay và cùng kì năm␣
↪ngoái",'Processor',ds_gia_tri_ve_theo_cot)
print(ds_df_tr_ve[0])
```

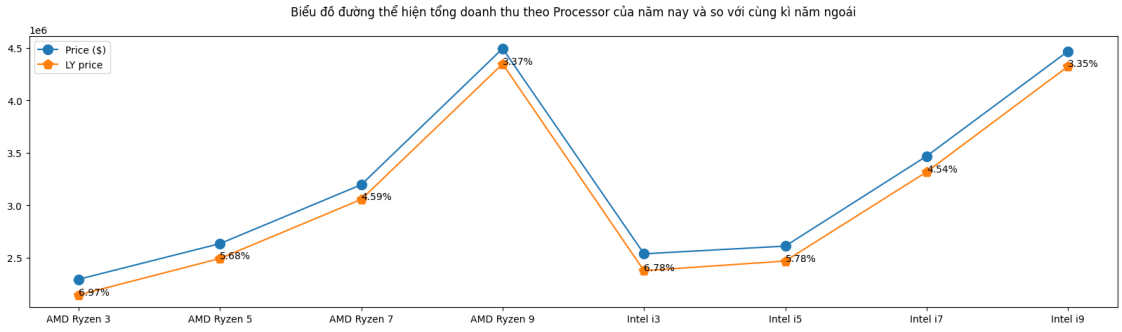
```
     Processor     Giá trị
0  AMD Ryzen 3  2144172.35
4     Intel i3  2375554.64
5     Intel i5  2467443.05
1  AMD Ryzen 5  2492416.26
2  AMD Ryzen 7  3055564.12
6     Intel i7  3317416.71
7     Intel i9  4323535.42
3  AMD Ryzen 9  4347806.78
```



[45]:
```
# vẽ biểu đồ đường thể hiện độ tăng trưởng của tổng doanh thu theo Processor
ve_bieu_do_duong(danh_sach_ten_cot,'Processor',"Biểu đồ đường thể hiện tổng␣
↪doanh thu theo Processor của năm nay và so với cùng kì năm ngoái\n",'%')
```

```
['Price ($)', 'LY price']
```

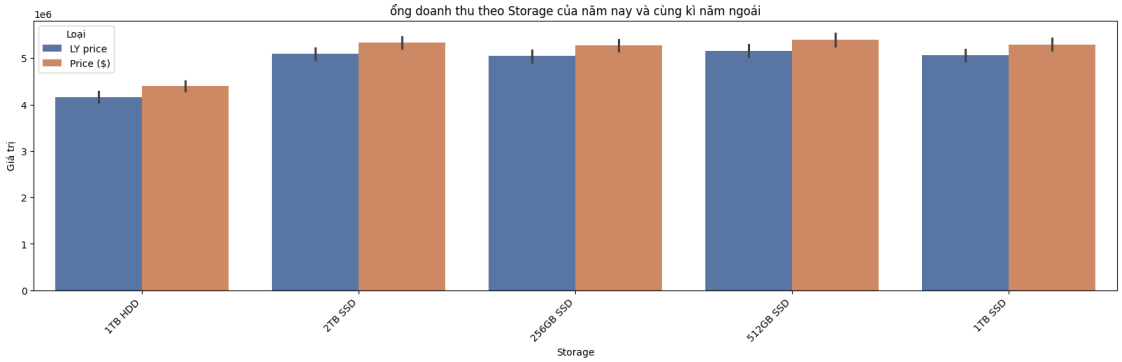
[45]:
```
(['AMD Ryzen 3',
  'AMD Ryzen 5',
  'AMD Ryzen 7',
  'AMD Ryzen 9',
  'Intel i3',
  'Intel i5',
  'Intel i7',
  'Intel i9'],
 array([6.96772347, 5.67722183, 4.59162349, 3.37411498, 6.77736463,
        5.77926206, 4.54269129, 3.35142391]))
```



[46]:
```
# vẽ biểu đồ cột với giá trị là tổng doanh thu của năm nay so sánh với cùng kì␣
↪năm ngoái theo Storage
danh_sach_ten_cot = ['Storage','Price ($)','LY price']
ds_gia_tri_ve_theo_cot = ['Price ($)','LY price']
```

```
ds_df_tr_ve = ve_nhieu_bieu_do_cot(danh_sach_ten_cot,"ổng doanh thu theo␣
  ↪Storage của năm nay và cùng kì năm ngoái",'Storage',ds_gia_tri_ve_theo_cot)
print(ds_df_tr_ve[0])
```

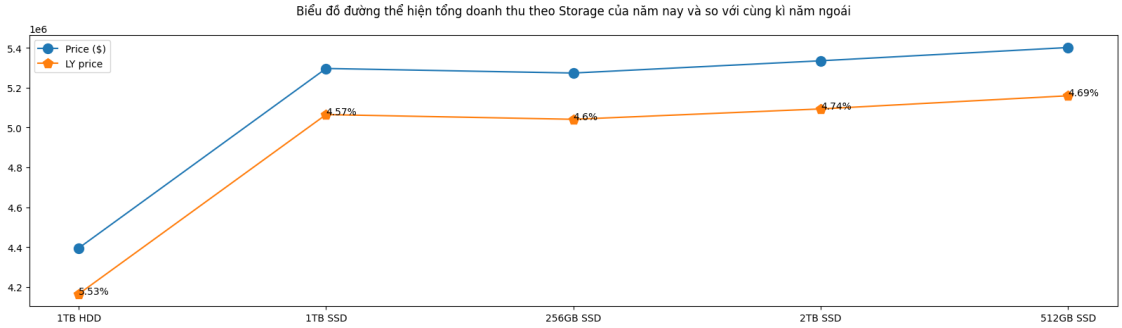
```
     Storage     Giá trị
0    1TB HDD  4165172.53
2  256GB SSD  5041225.23
1    1TB SSD  5064919.14
3    2TB SSD  5093336.78
4  512GB SSD  5159255.65
```



[47]:
```
# vẽ biểu đồ đường thể hiện độ tăng trưởng của tổng doanh thu theo Storage
ve_bieu_do_duong(danh_sach_ten_cot,'Storage',"Biểu đồ đường thể hiện tổng doanh␣
  ↪thu theo Storage của năm nay và so với cùng kì năm ngoái\n",'%')
```

```
['Price ($)', 'LY price']
```

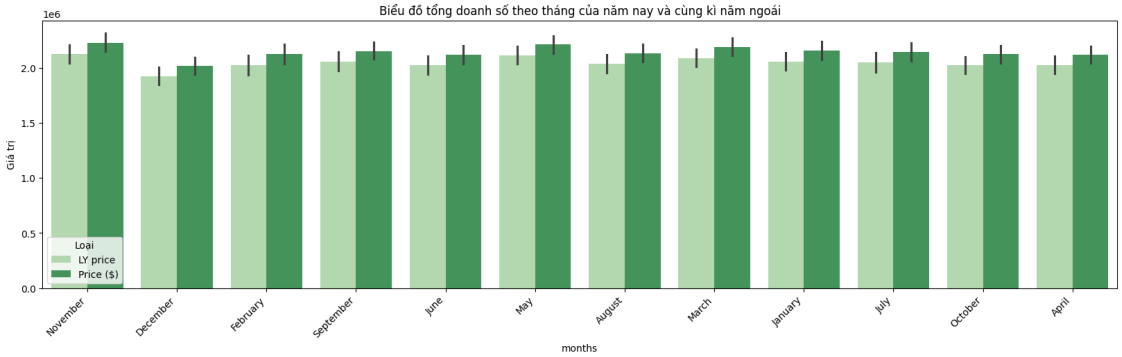
[47]:
```
(['1TB HDD', '1TB SSD', '256GB SSD', '2TB SSD', '512GB SSD'],
 array([5.52678186, 4.56670667, 4.60403948, 4.74148894, 4.69059912]))
```



[48]:
```
# vẽ biểu đồ cột với giá trị là tổng doanh thu của năm nay so sánh với cùng kì␣
  ↪năm ngoái theo months
```

```
danh_sach_ten_cot = ['months','Price ($)','LY price']
ds_gia_tri_ve_theo_cot = ['Price ($)','LY price']
ve_nhieu_bieu_do_cot(danh_sach_ten_cot,"Biểu đồ tổng doanh số theo tháng của␣
 ↪năm nay và cùng kì năm ngoái","months",ds_gia_tri_ve_theo_cot)
```

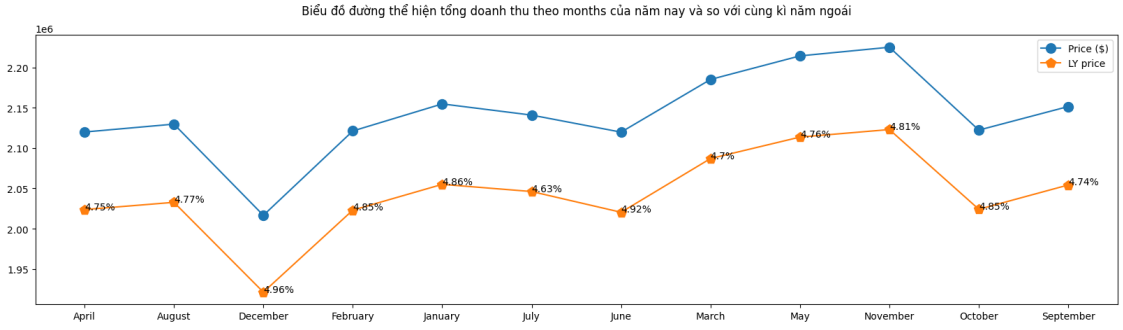
[48]:
```
[       months     Giá trị
 2    December  1921265.13
 6        June  2020352.90
 3    February  2022994.63
 0       April  2023694.85
 10    October  2024086.67
 1      August  2032684.81
 5        July  2046076.82
 11  September  2054207.95
 4      January  2054972.05
 7        March  2086993.41
 8          May  2113608.06
 9    November  2122972.05,
        months     Giá trị
 2    December  2016565.13
 6        June  2119752.90
 0       April  2119894.85
 3    February  2121194.63
 10    October  2122286.67
 1      August  2129684.81
 5        July  2140876.82
 11  September  2151507.95
 4      January  2154772.05
 7        March  2185093.41
 8          May  2214308.06
 9    November  2225072.05]
```



[49]:
```
# vẽ biểu đồ đường thể hiện độ tăng trưởng của tổng doanh thu theo months
ve_bieu_do_duong(danh_sach_ten_cot,"months","Biểu đồ đường thể hiện tổng doanh␣
 ↪thu theo months của năm nay và so với cùng kì năm ngoái\n",'%')
```

```
['Price ($)', 'LY price']
```

[49]:
```
(['April',
  'August',
  'December',
  'February',
  'January',
  'July',
  'June',
  'March',
  'May',
  'November',
  'October',
  'September'],
 array([4.75368112, 4.77201382, 4.96027323, 4.85418985, 4.85651374,
        4.63325712, 4.91993255, 4.70054191, 4.76436487, 4.80929553,
        4.85157091, 4.7366188 ]))
```



[50]:
```
data.head()
```

[50]:

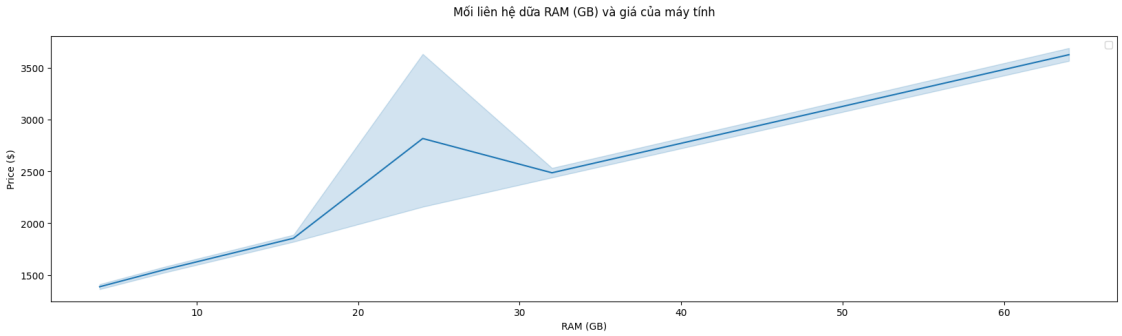
	Brand	Processor	RAM (GB)	Storage	GPU	\
0	Apple	AMD Ryzen 3	64	512GB SSD	Nvidia GTX 1650	
1	Razer	AMD Ryzen 7	4	1TB SSD	Nvidia RTX 3080	
2	Asus	Intel i5	32	2TB SSD	Nvidia RTX 3060	
3	Lenovo	Intel i5	4	256GB SSD	Nvidia RTX 3080	
4	Razer	Intel i3	4	256GB SSD	AMD Radeon RX 6600	

	Screen Size (inch)	Resolution	Battery Life (hours)	Weight (kg)	\
0	17.3	2560x1440	8.9	1.42	
1	14.0	1366x768	9.4	2.57	
2	13.3	3840x2160	8.5	1.74	

```
3                13.3   1366x768                  10.5         3.10
4                16.0  3840x2160                   5.7         3.38

  Operating System  Price ($)  LY price    months
0          FreeDOS    3997.07   3897.07   October
1            Linux    1355.78   1255.78  December
2          FreeDOS    2673.07   2573.07   October
3          Windows     751.17    651.17    August
4            Linux    2059.83   1959.83   October
```

[51]:
```
ten_cot = ["RAM (GB)",'Price ($)']
ve_bieu_do_duong(ten_cot,'RAM (GB)','Mối liên hệ dữa RAM (GB) và giá của máy␣
↪tính\n','1')
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:136:
UserWarning: No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is called with no
argument.
  plt.legend()
```



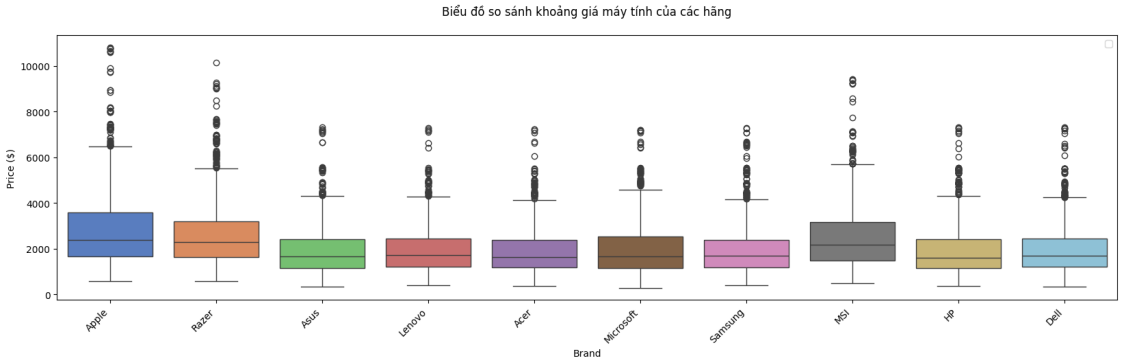
[52]:
```
data.head()
```

[52]:
```
    Brand    Processor  RAM (GB)    Storage                GPU  \
0   Apple  AMD Ryzen 3        64  512GB SSD     Nvidia GTX 1650
1   Razer  AMD Ryzen 7         4    1TB SSD     Nvidia RTX 3080
2    Asus     Intel i5        32    2TB SSD     Nvidia RTX 3060
3  Lenovo     Intel i5         4  256GB SSD     Nvidia RTX 3080
4   Razer     Intel i3         4  256GB SSD  AMD Radeon RX 6600

   Screen Size (inch) Resolution  Battery Life (hours)  Weight (kg)  \
0                17.3  2560x1440                   8.9         1.42
1                14.0   1366x768                   9.4         2.57
2                13.3  3840x2160                   8.5         1.74
3                13.3   1366x768                  10.5         3.10
4                16.0  3840x2160                   5.7         3.38
```

	Operating System	Price ($)	LY price	months
0	FreeDOS	3997.07	3897.07	October
1	Linux	1355.78	1255.78	December
2	FreeDOS	2673.07	2573.07	October
3	Windows	751.17	651.17	August
4	Linux	2059.83	1959.83	October

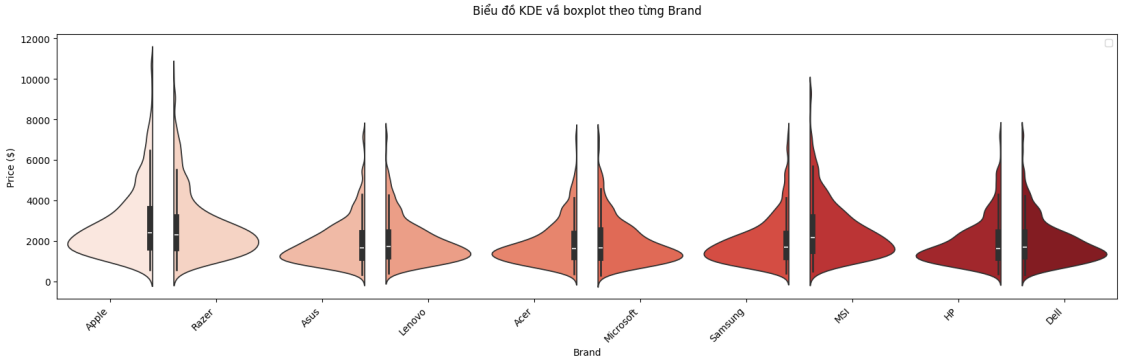
[53]:
```
# vẽ biểu đồ boxplot để so sánh khoảng giá máy tính theo brand
ten_cot = ['Brand','Price ($)']
ve_bieu_do_box_plot(ten_cot,'Biểu đồ so sánh khoảng giá máy tính của các␣
 ↪hãng\n','Brand')
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:153:
UserWarning: No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is called with no
argument.
  plt.legend()
```



[54]:
```
# vẽ biểu đồ violin thể hiện sự phân bố của giá và khoảng giá theo từng Brand
ten_cot = ['Brand','Price ($)']
ve_bieu_do_violin_plot(ten_cot,'Biểu đồ KDE và boxplot theo từng␣
 ↪Brand\n','Brand')
```

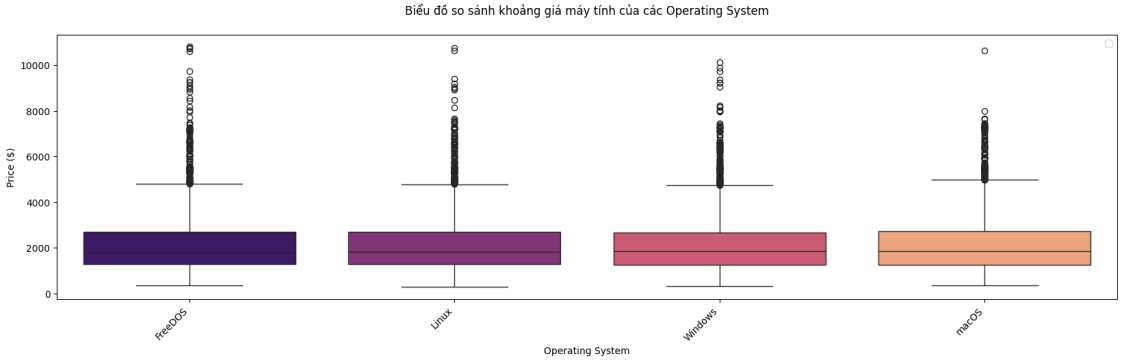
```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:162:
UserWarning: No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is called with no
argument.
  plt.legend()
```



Phân tích biểu đồ phân phối giá của máy tính theo Brand - *Từ biểu đồ Boxplot và biểu đồ violin ta thấy rằng đa phần dữ liệu về giá của các brand đều bị lệch xuống dưới cho thấy dữ liệu có độ phân tán mạnh tập trung chủ yếu ở những khoảng giá thấp*

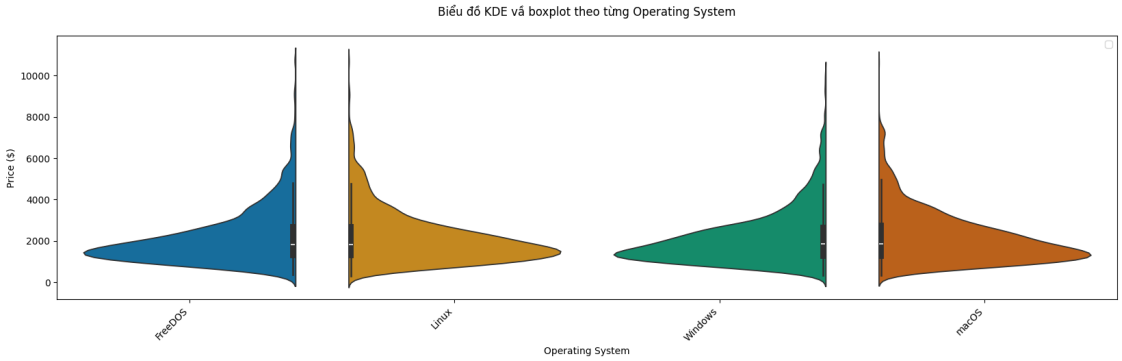
[55]:
```
# vẽ biểu đồ boxplot để so sánh khoảng giá máy tính theo Operating␣
 ↪System
ten_cot = ['Operating System','Price ($)']
ve_bieu_do_box_plot(ten_cot,'Biểu đồ so sánh khoảng giá máy tính của các␣
 ↪Operating System\n','Operating System')
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:153:
UserWarning: No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is called with no
argument.
  plt.legend()
```



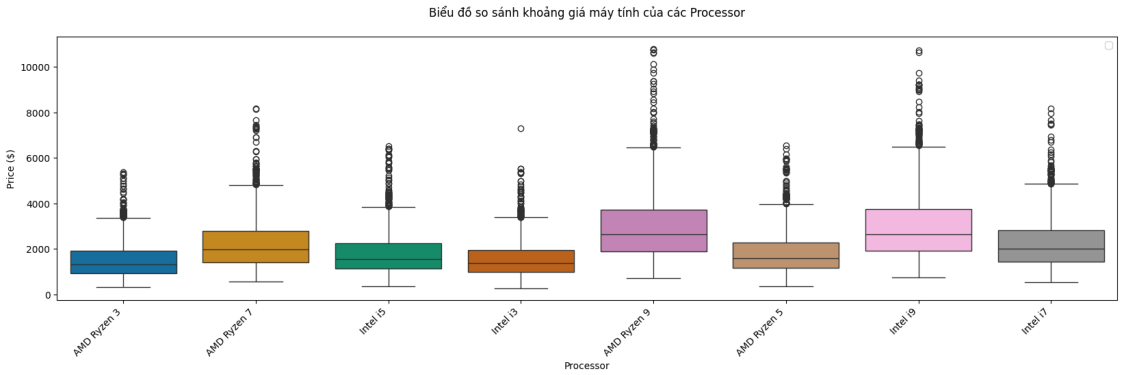
[56]:
```
# vẽ biểu đồ violin thể hiện sự phân bố của giá và khoảng giá theo từng␣
 ↪Operating System
ten_cot = ['Operating System','Price ($)']
ve_bieu_do_violin_plot(ten_cot,'Biểu đồ KDE và boxplot theo từng Operating␣
 ↪System\n','Operating System')
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:162:
UserWarning: No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is called with no
argument.
  plt.legend()
```



[57]:
```
# vẽ biểu đồ boxplot để so sánh khoảng giá máy tính theo Processor
ten_cot = ['Processor','Price ($)']
ve_bieu_do_box_plot(ten_cot,'Biểu đồ so sánh khoảng giá máy tính của các␣
 ↪Processor\n','Processor')
```

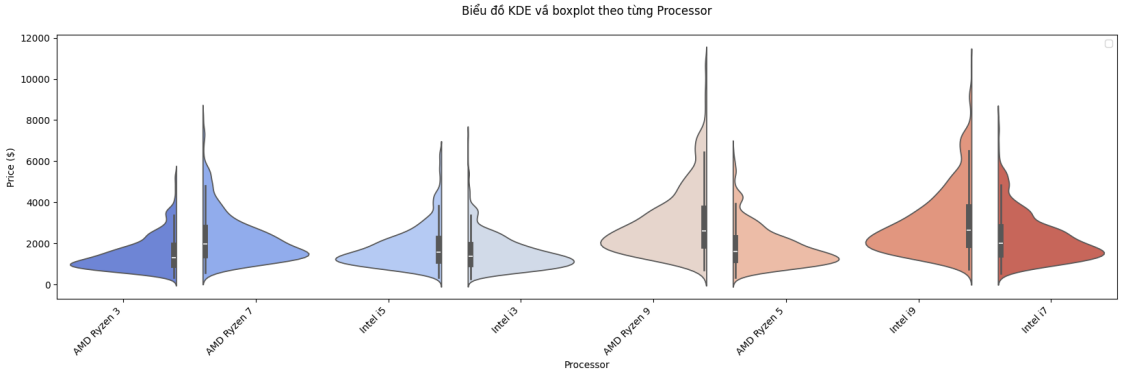
```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:153:
UserWarning: No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is called with no
argument.
  plt.legend()
```



[58]:
```
# vẽ biểu đồ violin thể hiện sự phân bố của giá và khoảng giá theo từng␣
 ↪Processor
ten_cot = ['Processor','Price ($)']
```

```
ve_bieu_do_violin_plot(ten_cot,'Biểu đồ KDE và boxplot theo từng␣
↪Processor\n','Processor')
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\3140114121.py:162:
UserWarning: No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is called with no
argument.
  plt.legend()
```



[59]:
```
data.head()
```

[59]:

	Brand	Processor	RAM (GB)	Storage	GPU	\
0	Apple	AMD Ryzen 3	64	512GB SSD	Nvidia GTX 1650	
1	Razer	AMD Ryzen 7	4	1TB SSD	Nvidia RTX 3080	
2	Asus	Intel i5	32	2TB SSD	Nvidia RTX 3060	
3	Lenovo	Intel i5	4	256GB SSD	Nvidia RTX 3080	
4	Razer	Intel i3	4	256GB SSD	AMD Radeon RX 6600	

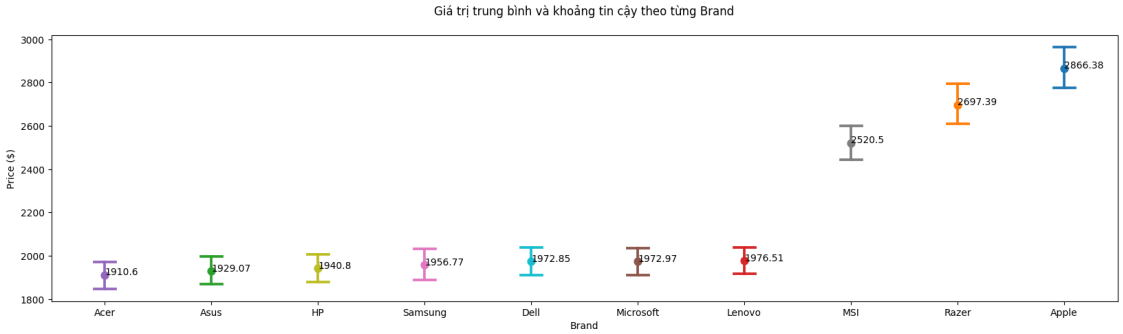
	Screen Size (inch)	Resolution	Battery Life (hours)	Weight (kg)	\
0	17.3	2560x1440	8.9	1.42	
1	14.0	1366x768	9.4	2.57	
2	13.3	3840x2160	8.5	1.74	
3	13.3	1366x768	10.5	3.10	
4	16.0	3840x2160	5.7	3.38	

	Operating System	Price ($)	LY price	months
0	FreeDOS	3997.07	3897.07	October
1	Linux	1355.78	1255.78	December
2	FreeDOS	2673.07	2573.07	October
3	Windows	751.17	651.17	August
4	Linux	2059.83	1959.83	October

[60]:
```
# giá trị trung bình của máy tính theo Brand
ten_cot = ['Brand','Price ($)']
```

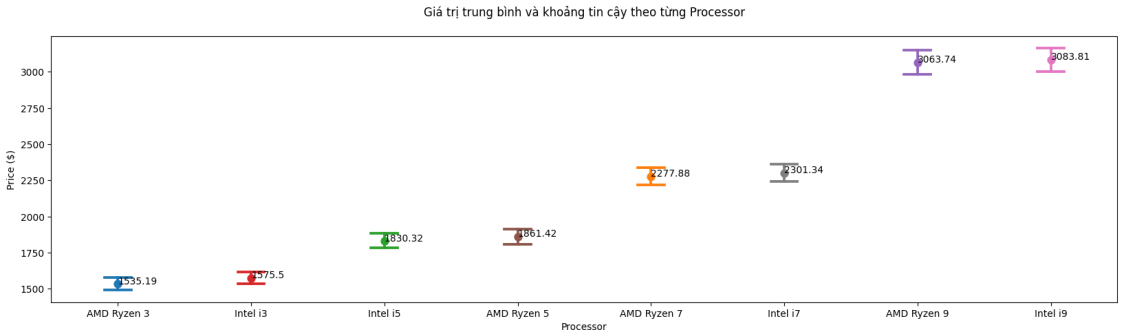
```
ve_bieu_do_poinplot(ten_cot,'Giá trị trung bình và khoảng tin cậy theo từng␣
↪Brand\n',hue='Brand')
```

```
Brand
Acer          1910.601179
Asus          1929.070820
HP            1940.801850
Samsung       1956.773976
Dell          1972.846010
Microsoft     1972.967577
Lenovo        1976.506247
MSI           2520.497555
Razer         2697.389901
Apple         2866.381343
Name: Price ($), dtype: float64
```



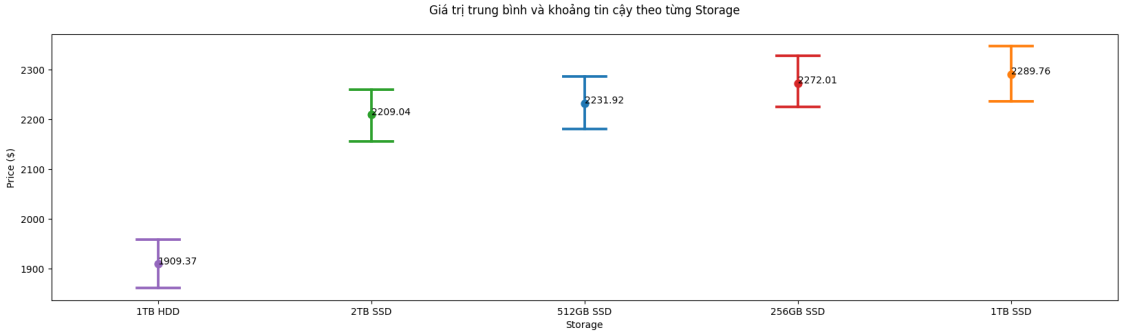
[61]:
```
# giá trị trung bình của máy tính theo Processor
ten_cot = ['Processor','Price ($)']
ve_bieu_do_poinplot(ten_cot,'Giá trị trung bình và khoảng tin cậy theo từng␣
↪Processor\n',hue='Processor')
```

```
Processor
AMD Ryzen 3    1535.188989
Intel i3       1575.499776
Intel i5       1830.324719
AMD Ryzen 5    1861.424919
AMD Ryzen 7    2277.878917
Intel i7       2301.338228
AMD Ryzen 9    3063.740136
Intel i9       3083.806363
Name: Price ($), dtype: float64
```



[62]:
```
# giá trị trung bình của máy tính theo Storage
ten_cot = ['Storage','Price ($)']
ve_bieu_do_poinplot(ten_cot,'Giá trị trung bình và khoảng tin cậy theo từng␣
↪Storage\n',hue='Storage')
```

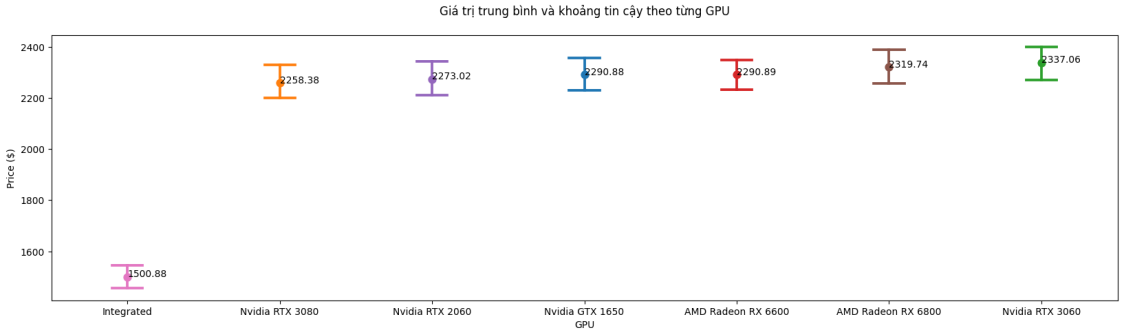
```
Storage
1TB HDD      1909.371212
2TB SSD      2209.042145
512GB SSD    2231.923822
256GB SSD    2272.005700
1TB SSD      2289.761842
Name: Price ($), dtype: float64
```



[63]:
```
# giá trị trung bình của máy tính theo GPU
ten_cot = ['GPU','Price ($)']
ve_bieu_do_poinplot(ten_cot,'Giá trị trung bình và khoảng tin cậy theo từng␣
↪GPU\n',hue='GPU')
```

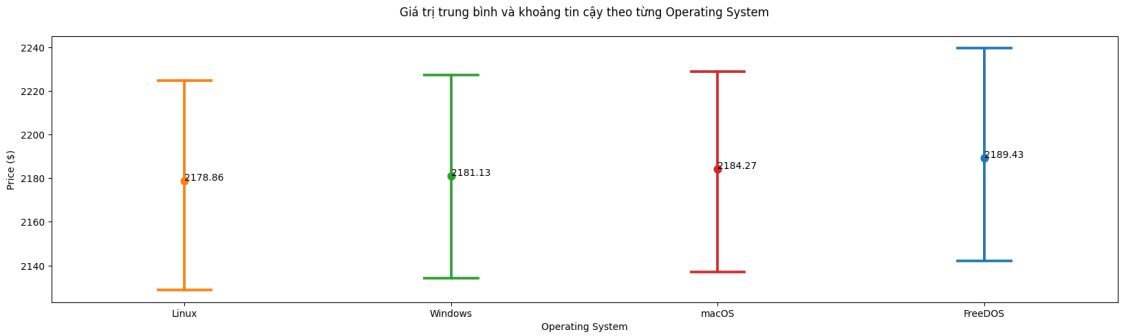
```
GPU
Integrated           1500.884743
Nvidia RTX 3080      2258.381861
Nvidia RTX 2060      2273.021833
Nvidia GTX 1650      2290.877444
```

```
AMD Radeon RX 6600    2290.889069
AMD Radeon RX 6800    2319.744506
Nvidia RTX 3060       2337.062001
Name: Price ($), dtype: float64
```



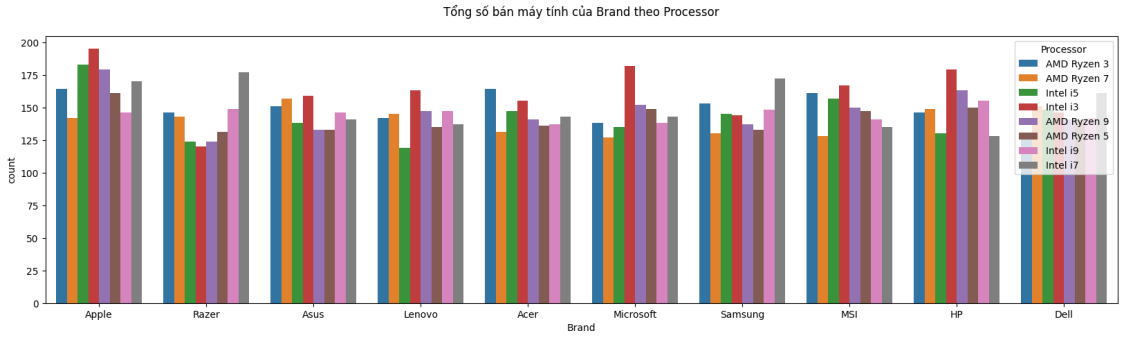
[64]:
```
# giá trị trung bình của máy tính theo Operating System
ten_cot = ['Operating System','Price ($)']
ve_bieu_do_poinplot(ten_cot,'Giá trị trung bình và khoảng tin cậy theo từng␣
↪Operating System\n',hue='Operating System')
```

```
Operating System
Linux      2178.855307
Windows    2181.125181
macOS      2184.268827
FreeDOS    2189.426629
Name: Price ($), dtype: float64
```



[65]:
```
# Tổng số bán máy tính của brand theo processer
data.head()
plt.figure(figsize=(20,5))
sns.countplot(x='Brand',data=data,hue='Processor')
plt.title("Tổng số bán máy tính của Brand theo Processor\n")
```

[65]: Text(0.5, 1.0, 'Tổng số bán máy tính của Brand theo Processor\n')



[66]:
```
# Lấy ra xem mỗi brand thì có bao nhiêu processer
ten_hang = data['Brand'].value_counts().index.to_list()
df_brand_processoe = data.groupby("Brand")['Processor'].value_counts().
 ↪reset_index()
total_processer_brand = {}
for i in range(len(ten_hang)):
    total_processer_brand[ten_hang[i]] =␣
 ↪len(df_brand_processoe[df_brand_processoe['Brand']==ten_hang[i]])
total_processer_brand
```

[66]:
```
{'Apple': 8,
 'HP': 8,
 'MSI': 8,
 'Microsoft': 8,
 'Samsung': 8,
 'Asus': 8,
 'Dell': 8,
 'Acer': 8,
 'Lenovo': 8,
 'Razer': 8}
```

[67]:
```
# sử dụng pivot để lấy ra từng hãng có bao nhiêu processer mỗi loại
df_brand_processoe_wide_forrmat = df_brand_processoe.
 ↪pivot(index="Processor",columns="Brand",values="count")
df_brand_processoe_wide_forrmat = df_brand_processoe_wide_forrmat.reset_index()
df_brand_processoe_wide_forrmat
```

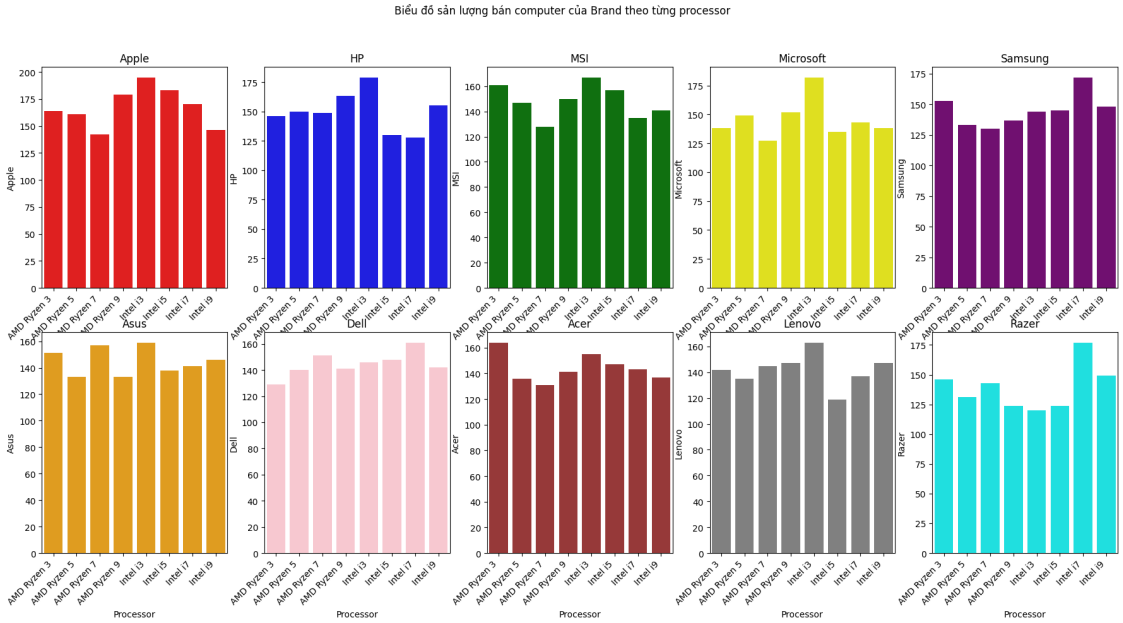
[67]:

Brand	Processor	Acer	Apple	Asus	Dell	HP	Lenovo	MSI	Microsoft	\
0	AMD Ryzen 3	164	164	151	129	146	142	161	138	
1	AMD Ryzen 5	136	161	133	140	150	135	147	149	
2	AMD Ryzen 7	131	142	157	151	149	145	128	127	
3	AMD Ryzen 9	141	179	133	141	163	147	150	152	

```
4          Intel i3   155    195   159   146  179     163  167        182
5          Intel i5   147    183   138   148  130     119  157        135
6          Intel i7   143    170   141   161  128     137  135        143
7          Intel i9   137    146   146   142  155     147  141        138

Brand  Razer  Samsung
0        146      153
1        131      133
2        143      130
3        124      137
4        120      144
5        124      145
6        177      172
7        149      148
```

[68]:
```
# vẽ biểu đồ thể hiện tổng số lượng bán của brand theo từng loại processer
fig,ax = plt.subplots(ncols=5,nrows=2,figsize=(22,10))
index=0
for i in range(2):
    for j in range(5):
        sns.
 ↪barplot(x='Processor',y=ten_hang[index],data=df_brand_processoe_wide_forrmat,color=color_li
        ax[i][j].
 ↪set_xticks(range(len(df_brand_processoe_wide_forrmat['Processor'])))
        ax[i][j].set_xticklabels(df_brand_processoe_wide_forrmat['Processor'],␣
 ↪rotation=45,ha='right')
        ax[i][j].set_title(ten_hang[index])
        index+=1
fig.suptitle("Biểu đồ sản lượng bán computer của Brand theo từng processor\n")
```

[68]:
```
Text(0.5, 0.98, 'Biểu đồ sản lượng bán computer của Brand theo từng
processor\n')
```



[69]:
```
data.head()
```

[69]:

	Brand	Processor	RAM (GB)	Storage	GPU	\
0	Apple	AMD Ryzen 3	64	512GB SSD	Nvidia GTX 1650	
1	Razer	AMD Ryzen 7	4	1TB SSD	Nvidia RTX 3080	
2	Asus	Intel i5	32	2TB SSD	Nvidia RTX 3060	
3	Lenovo	Intel i5	4	256GB SSD	Nvidia RTX 3080	
4	Razer	Intel i3	4	256GB SSD	AMD Radeon RX 6600	

	Screen Size (inch)	Resolution	Battery Life (hours)	Weight (kg)	\
0	17.3	2560x1440	8.9	1.42	
1	14.0	1366x768	9.4	2.57	
2	13.3	3840x2160	8.5	1.74	
3	13.3	1366x768	10.5	3.10	
4	16.0	3840x2160	5.7	3.38	

	Operating System	Price ($)	LY price	months
0	FreeDOS	3997.07	3897.07	October
1	Linux	1355.78	1255.78	December
2	FreeDOS	2673.07	2573.07	October
3	Windows	751.17	651.17	August
4	Linux	2059.83	1959.83	October

[70]:
```
# Tạo một list danh sách gồm các tháng trong năm
months = [
    "January", "February", "March", "April", "May", "June",
    "July", "August", "September", "October", "November", "December"
```

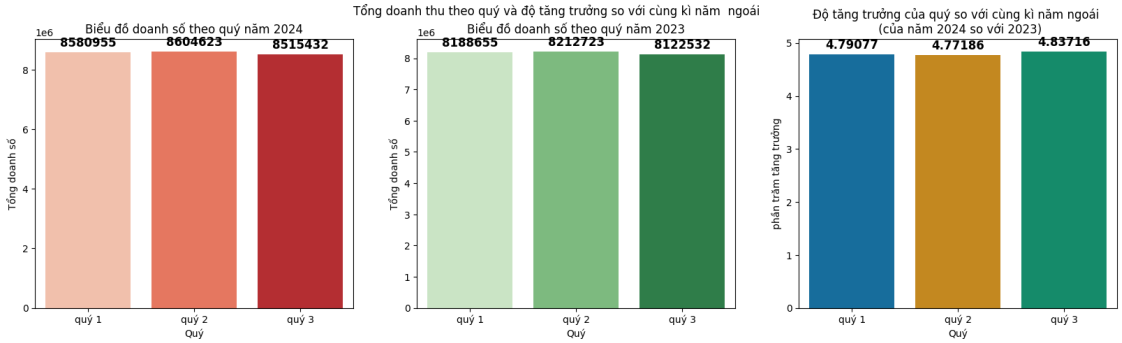
```
]

# tạo các list rỗng để chứa  dữ liệu doanh thu từng quý của năm 2023 và 2024
tong_doanh_thu_theo_quy_2024 = []
tong_doanh_thu_theo_quy_2023 = []

# Tính toán để đưa ra doanh số theo từng quý của năm 2024 và cùng kì năm ngoái␣
 ↪(2023)
for i in range(0,len(months),4):
    batch = months[i:i+4]
    quy = data[data['months'].isin(batch)]
    quy_2024 = quy['Price ($)'].sum()
    quy_2023 = quy['LY price'].sum()
    tong_doanh_thu_theo_quy_2024.append(quy_2024)
    tong_doanh_thu_theo_quy_2023.append(quy_2023)

# vẽ biểu đồ doanh số theo từng quý năm 2024
fig,ax = plt.subplots(ncols=3,nrows=1,figsize=(20,5))
bars = sns.barplot(x=['quý 1','quý 2','quý␣
 ↪3'],y=tong_doanh_thu_theo_quy_2024,ax=ax[0],palette=palettes[randint(0,len(palettes)-1)])
ax[0].set_title("Biểu đồ doanh số theo quý năm 2024")
ax[0].set_xlabel('Quý')
ax[0].set_ylabel('Tổng doanh số')

# ghi số liệu cho biểu đồ năm 2024
for i in range(3):
    ax[0].bar_label(bars.containers[i], fmt="%.0f", fontsize=12,␣
 ↪fontweight="bold", padding=3)

# vẽ biểu đồ doanh số theo từng quý năm 2023
bar_1 = sns.barplot(x=['quý 1','quý 2','quý␣
 ↪3'],y=tong_doanh_thu_theo_quy_2023,ax=ax[1],palette=palettes[randint(0,len(palettes)-1)])
ax[1].set_title("Biểu đồ doanh số theo quý năm 2023")
ax[1].set_xlabel('Quý')
ax[1].set_ylabel('Tổng doanh số')

# ghi số liệu cho năm 2023
for i in range(3):
    ax[1].bar_label(bar_1.containers[i], fmt="%.0f", fontsize=12,␣
 ↪fontweight="bold", padding=3)

# Tính toán phần trăm tăng trưởng của quý trong năm 2024 so với cùng kì năm␣
 ↪ngoái (2023)
phan_trams= []
for i in range(3):
```

```
    phan_tram =␣
 ↪((tong_doanh_thu_theo_quy_2024[i]-tong_doanh_thu_theo_quy_2023[i])/
 ↪tong_doanh_thu_theo_quy_2023[i])*100
    phan_trams.append(phan_tram)
bar_2 = sns.barplot(x=['quý 1','quý 2','quý␣
 ↪3'],y=phan_trams,palette=palettes[randint(0,len(palettes)-1)])
ax[2].set_xlabel("Quý")
ax[2].set_ylabel("phần trăm tăng trưởng")
ax[2].set_title("Độ tăng trưởng của quý so với cùng kì năm ngoái \n (của năm␣
 ↪2024 so với 2023)")

# ghi số liệu tăng trưởng
for i in range(3):
    ax[2].bar_label(bar_2.containers[i], fontsize=12, fontweight="bold",␣
 ↪padding=3)
phan_trams


fig.suptitle("Tổng doanh thu theo quý và độ tăng trưởng so với cùng kì năm ␣
 ↪ngoái \n\n\n")
```

```
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\712322202.py:22:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  bars = sns.barplot(x=['quý 1','quý 2','quý 3'],y=tong_doanh_thu_theo_quy_2024,
ax=ax[0],palette=palettes[randint(0,len(palettes)-1)])
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\712322202.py:32:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  bar_1 = sns.barplot(x=['quý 1','quý 2','quý 3'],y=tong_doanh_thu_theo_quy_2023
,ax=ax[1],palette=palettes[randint(0,len(palettes)-1)])
C:\Users\HP Victus\AppData\Local\Temp\ipykernel_9732\712322202.py:46:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  bar_2 = sns.barplot(x=['quý 1','quý 2','quý
```

```
3'],y=phan_trams,palette=palettes[randint(0,len(palettes)-1)])
```

[70]:
```
Text(0.5, 0.98, 'Tổng doanh thu theo quý và độ tăng trưởng so với cùng kì năm
ngoái \n\n\n')
```



[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: