**Report 8**

**Part 1: Course exercises from reading material Ch6.1**

1.1: Line2D

1.2: 堆积图

1.3: 10

2.1: F

2.2: T

2.3: F

3.1: D

3.2: C

3.3: A

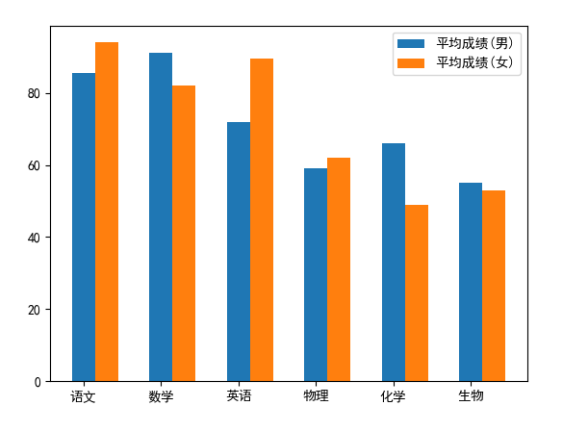
3.4: D

3.5: C

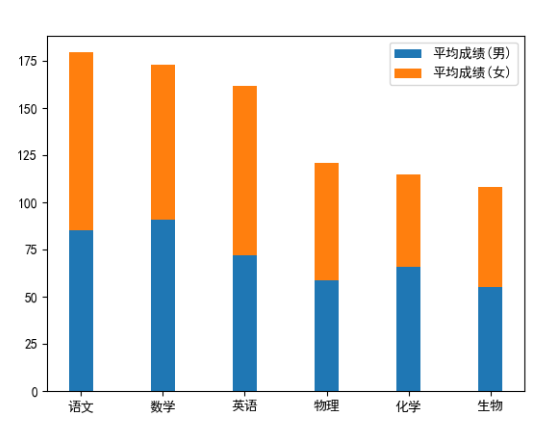
4.1:

import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
df = pd.DataFrame({'学科':['语文','数学','英语','物理','化学','生物'],  
 '平均成绩(男)':[85.5,91,72,59,66,55],  
 '平均成绩(女)':[94,82,89.5,62,49,53]})  
print(df)  
  
x = np.arange(6)  
x1 = df['学科']  
y1 = np.array(df['平均成绩(男)'])  
y2 = np.array(df['平均成绩(女)'])  
  
bar\_width = 0.3  
  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.bar(x,y1,tick\_label=x1,width=bar\_width)  
plt.bar(x+bar\_width,y2,width=bar\_width)

plt.legend()  
plt.show()

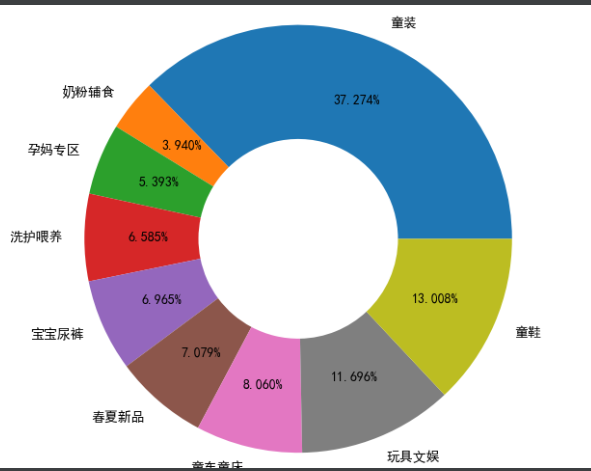


import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
  
  
df = pd.DataFrame({'学科':['语文','数学','英语','物理','化学','生物'],  
 '平均成绩(男)':[85.5,91,72,59,66,55],  
 '平均成绩(女)':[94,82,89.5,62,49,53]})  
print(df)  
  
x = np.arange(6)  
x1 = df['学科']  
y1 = np.array(df['平均成绩(男)'])  
y2 = np.array(df['平均成绩(女)'])  
  
bar\_width = 0.3  
  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.bar(x,y1,tick\_label=x1,width=bar\_width,label='平均成绩(男)')  
# 通过修改第二个柱状图的起始y坐标为上一个数据的 值，最终就能实现柱状图的堆叠  
plt.bar(x,y2,bottom=y1,width=bar\_width,label='平均成绩(女)')  
  
# plt.bar(x,y1,tick\_label=x1,width=bar\_width,label='平均成绩(男)',color='blue')  
# plt.bar(x+bar\_width,y2,width=bar\_width,label='平均成绩(女)',color='red')  
# plt.legend(['平均成绩(男)'])  
plt.legend()  
  
plt.show()



4.2:

import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
  
  
df = pd.DataFrame({'子类目':['童装','奶粉辅食','孕妈专区','洗护喂养','宝宝尿裤','春夏新品','童车童床','玩具文娱','童鞋'],  
 '销售额':[29665,3135.4,4292.4,5240.9,5543.4,5633.8,6414.5,9308.1,10353]})  
  
name = np.array(df['子类目'])  
data = np.array(df['销售额'])  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.pie(data,radius=1.5,wedgeprops={'width' : 0.8},labels=name,  
 autopct='%0.3lf%%',pctdistance=0.7)  
plt.show()

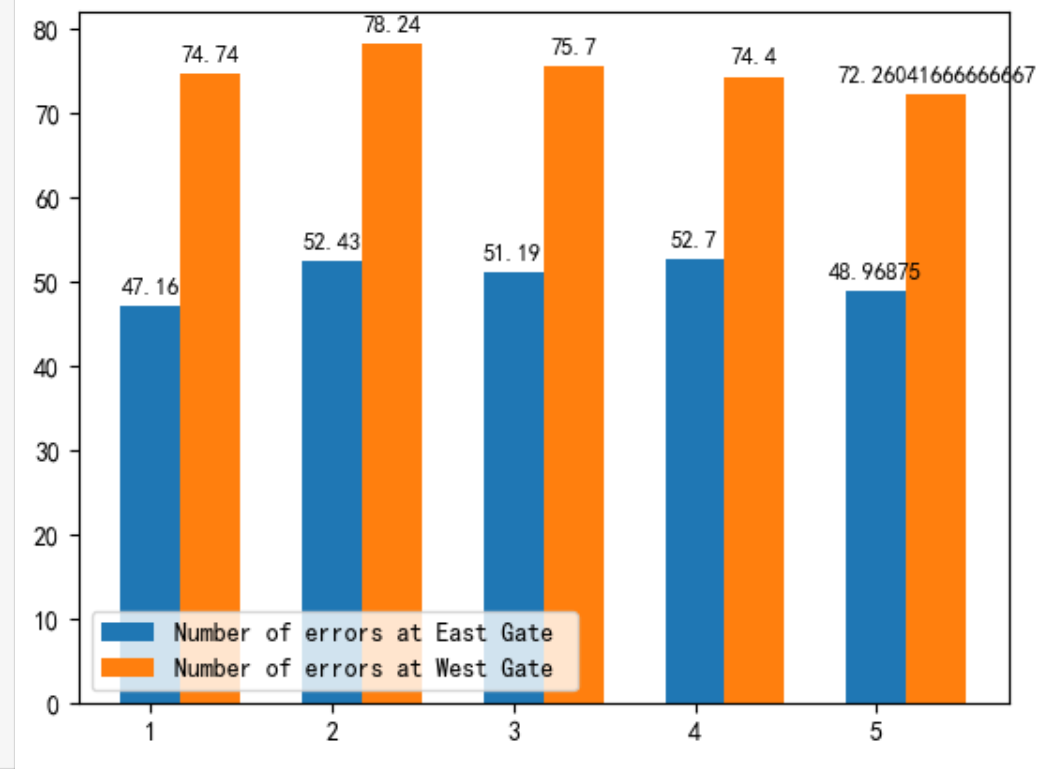


**Part 2:**

**Use matplotlib to implement data visualization tasks in the Assignment7.xlsx on Moodle. You need to choose 1 method to plot the data, explain why you choose this method and analyze what you see from the charts in the report.**

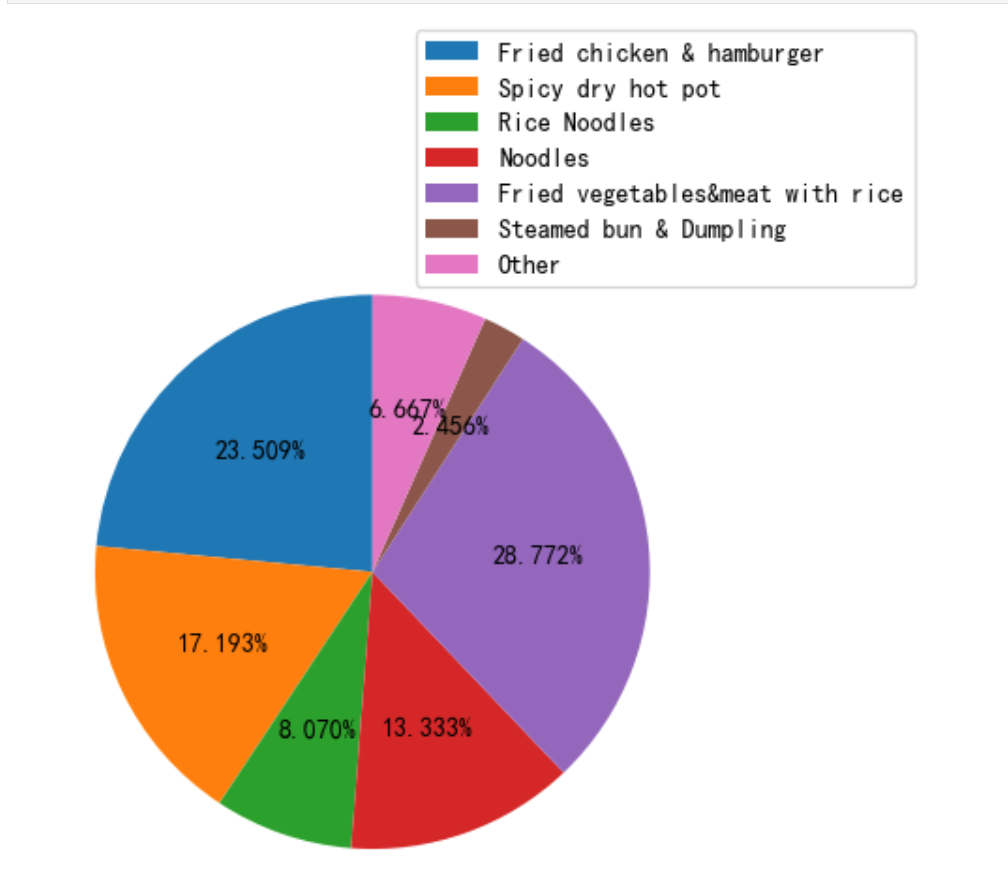
**TASK1:**

import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import openpyxl as oxl  
import datetime  
  
def delet\_nan(df) :  
 df = df.dropna(axis=0,how ='all')  
 return df  
  
task\_df1 = pd.read\_excel('../Assignment8.xlsx',sheet\_name='Task1',header=1)  
  
task\_df1.info()  
  
task\_df1 = delet\_nan(task\_df1)  
task\_df1.info()  
# reindex  
# df1 = task\_df1.reindex(index=np.arange(0,32063))  
# df1  
# 删除缺失值  
df1=task\_df1[['Number of errors at East Gate ','Number of errors at West Gate ']]  
df1 = delet\_nan(df1)  
# 去除重复值  
df1.drop\_duplicates(inplace=True,ignore\_index=False)  
df1.info()  
df1.head()  
df1.tail()  
# df1[['Number of errors at East Gate']]  
# list DataFrame 可以直接获取列名称  
list\_col = []  
list\_col = list(df1)  
# list\_col  
  
east = np.array([])  
west = east  
vcnt = east  
lim = 100  
cnt,esum,wsum = 0,0,0  
  
for index,value in df1.iterrows() :  
 # print(value)  
 s = str(value.iloc[0])  
 # print(s)  
 if not s.isdigit() :  
 continue  
 esum += (int)(value.iloc[0])  
 wsum +=(int)(value.iloc[1])  
 cnt += 1  
 if cnt >= lim :  
 vcnt = np.append(vcnt,[cnt])  
 cnt = 0  
 east = np.append(east,[esum])  
 west = np.append(west,[wsum])  
 esum = 0  
 wsum = 0  
  
  
east = np.append(east,[esum])  
west = np.append(west,[wsum])  
vcnt = np.append(vcnt,[cnt])  
print(east)  
print(west)  
emean = []  
wmean = []  
  
emean = east/vcnt  
wmean = west/vcnt  
  
print(emean)  
print(wmean)  
def autolabel(x) :  
 *"""  
 将柱状图上添加标签  
 """* for rec in x :  
 re\_h = rec.get\_height()  
 re\_x = rec.get\_x()  
 re\_width = rec.get\_width()  
 plt.text(re\_x+re\_width/2,re\_h+1,s='{}'.format(re\_h),ha='center',va='bottom',fontsize=9)  
  
  
bar\_width = 0.33  
x = np.arange(1,6)  
recta = plt.bar(x,emean,width=bar\_width)  
rectb = plt.bar(x+bar\_width,wmean,width=bar\_width)  
autolabel(recta)  
autolabel(rectb)  
  
  
  
plt.legend(list\_col,loc='lower left')  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.show()



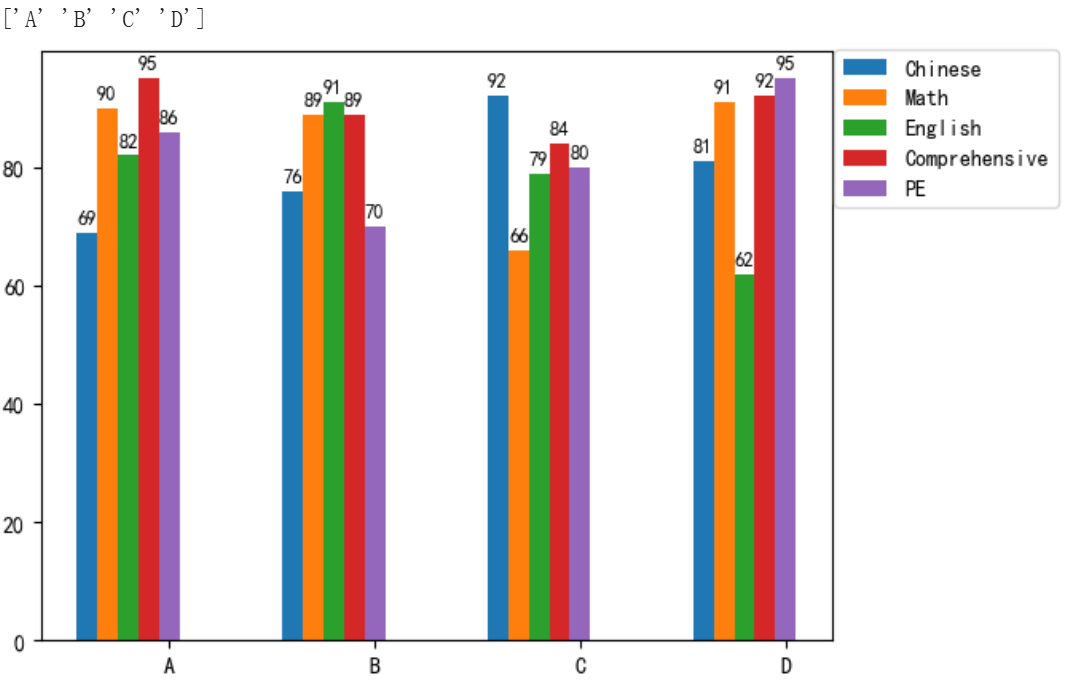
**TASK2:**

import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import openpyxl as oxl  
import datetime  
  
def delet\_nan(df) :  
 df = df.dropna(axis=0,how ='all')  
 return df  
  
task\_df2 = pd.read\_excel('../Assignment8.xlsx',sheet\_name='Task2',header=1)  
  
  
df2 = delet\_nan(task\_df2)  
# df2  
list2 = []  
list2 = list(df2)  
list2 = list2[1:3]  
df2 = df2[list2]  
  
label\_y = np.array(df2[list2[0]])  
x= np.array(df2[list2[1]])  
# print(label\_y)  
# print(x)  
# pie chart  
  
plt.pie(x,autopct='%.03lf%%',startangle=90)  
  
  
  
plt.legend(label\_y,loc='upper right',bbox\_to\_anchor=[1.3,1.3])  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.show()



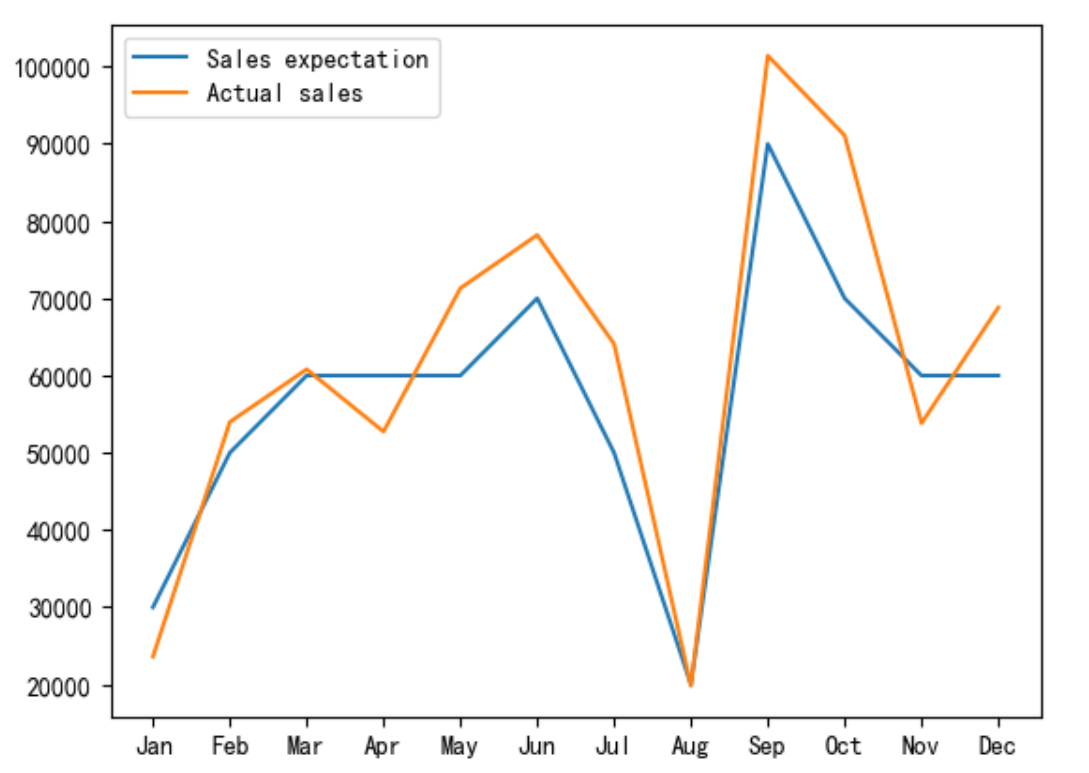
**TASK3:**

import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import openpyxl as oxl  
import datetime  
  
def delet\_nan(df) :  
 df = df.dropna(axis=0,how ='all')  
 return df  
  
task\_df3 = pd.read\_excel('../Assignment8.xlsx',sheet\_name='Task3',header=1)  
  
def autolabel(x) :  
 *"""  
 将柱状图上添加标签  
 """* for rec in x :  
 re\_h = rec.get\_height()  
 re\_x = rec.get\_x()  
 re\_width = rec.get\_width()  
 plt.text(re\_x+re\_width/2,re\_h+1,s='{}'.format(re\_h),ha='center',va='bottom',fontsize=9)  
  
  
df3= delet\_nan(task\_df3)  
# df3  
list3 = []  
list3 = list(df3)  
list3 = list3[1:7]  
df3 = df3[list3]  
label3 = df3[list3[0]]  
df3 = df3[list3[1:]]  
arr3 = np.array(df3)  
label\_y = np.array(list3[1:])  
label\_x = np.array(label3)  
print(label\_x)  
# print(arr3)  
# print(label\_y)  
  
arr3 = arr3.T  
cnt = 0  
width3 = 0.1  
pos\_x = np.arange(1,5)  
for rows in arr3:  
 recta = plt.bar(pos\_x+cnt\*width3,rows,width=width3,tick\_label=label\_x)  
 autolabel(recta)  
 cnt+= 1  
plt.legend(label\_y,loc='upper right',bbox\_to\_anchor=[1.3,1.02])  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.show()

****

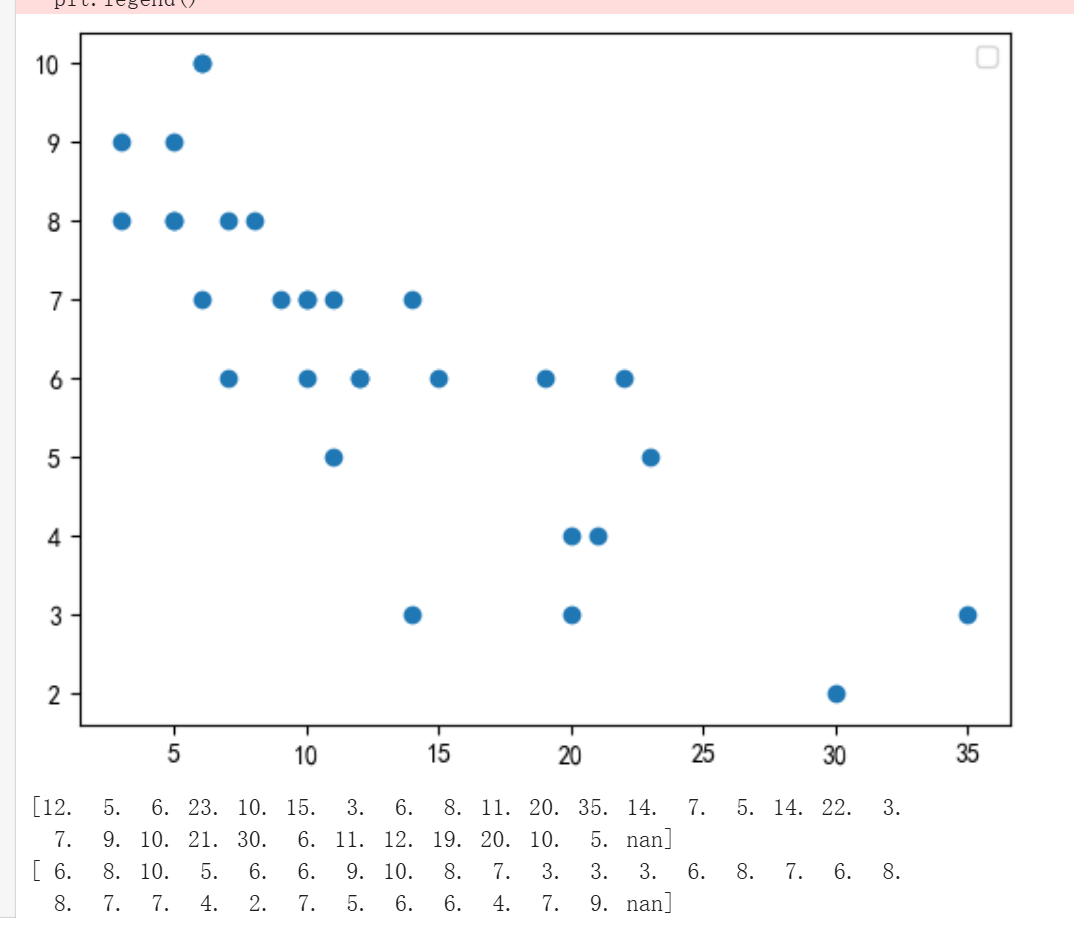
**TASK4:**

import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import openpyxl as oxl  
import datetime  
  
def delet\_nan(df) :  
 df = df.dropna(axis=0,how ='all')  
 return df  
  
task\_df4 = pd.read\_excel('../Assignment8.xlsx',sheet\_name='Task4',header=1)  
  
  
df4 = delet\_nan(task\_df4)  
  
  
list4 = []  
list4 = list(df4)  
list4 = list4[1:4]  
df4 = df4[list4]  
  
label\_y = list4[1:]  
label4 = df4[list4[0]]  
df4 = df4[list4[1:]]  
df4 = np.array(df4)  
label4 = np.array(label4)  
  
plt.plot(label4,df4)  
plt.legend(label\_y)  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.show()  
# label\_y



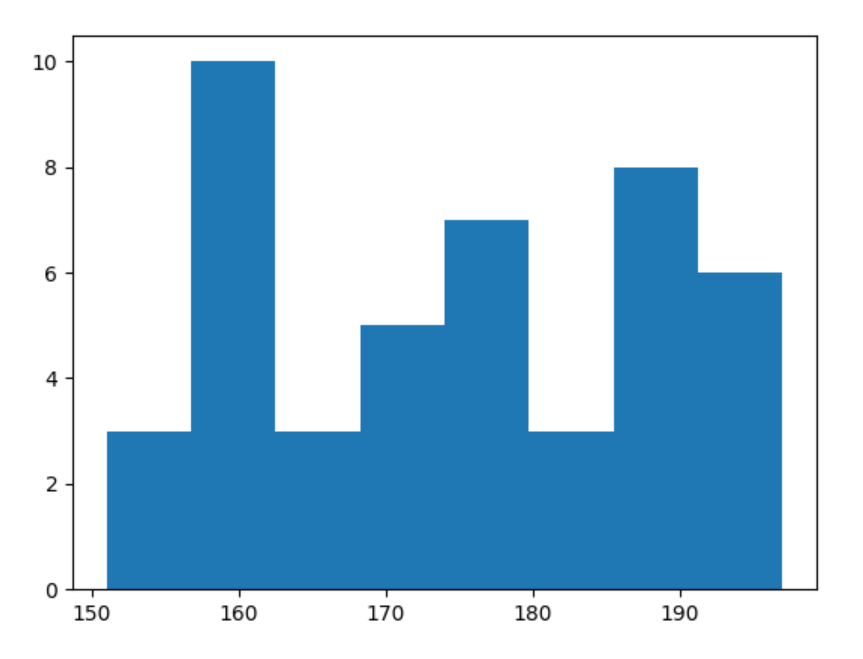
**TASK5:**

import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import openpyxl as oxl  
import datetime  
  
def delet\_nan(df) :  
 df = df.dropna(axis=0,how ='all')  
 return df  
  
task\_df5 = pd.read\_excel('../Assignment8.xlsx',sheet\_name='Task5',header=1)  
  
df5 = delet\_nan(task\_df5)  
  
  
list5 = []  
list5 = list(df5)  
list5 = list5[2:4]  
df5 = df5[list5]  
# df5 = delet\_nan(df5)  
  
label\_y = list5  
# print(df5.iloc[:,0])  
px = np.array(df5.iloc[:,0])  
py = np.array(df5.iloc[:,1])  
  
plt.scatter(px,py)  
  
plt.show()  
print(px)  
print(py)  
  
# label5 = df5[list5[0]]  
# px



**TASK6:**

import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import openpyxl as oxl  
import datetime  
  
def delet\_nan(df) :  
 df = df.dropna(axis=0,how ='all')  
 return df  
  
task\_df6 = pd.read\_excel('../Assignment8.xlsx',sheet\_name='Task6',header=1)  
  
df6 = delet\_nan(task\_df6)  
df6 =df6.iloc[0:10,1:6]  
print(df6)  
# df6 = df6.sort\_values()  
arr = np.array(df6)  
# 两种将二维数组更改为一维数组的方式  
# arr = arr.flatten()  
# arr = arr.reshape(arr.shape[0]\*arr.shape[1])  
arr = arr.reshape(-1,)  
# numpy 统计个数 一个数组内同一元素的个数  
# dfarr = pd.Series(arr).value\_counts()  
# x = np.array(dfarr.index)  
# y = dfarr.values  
  
plt.hist(arr,bins=8)  
plt.show()

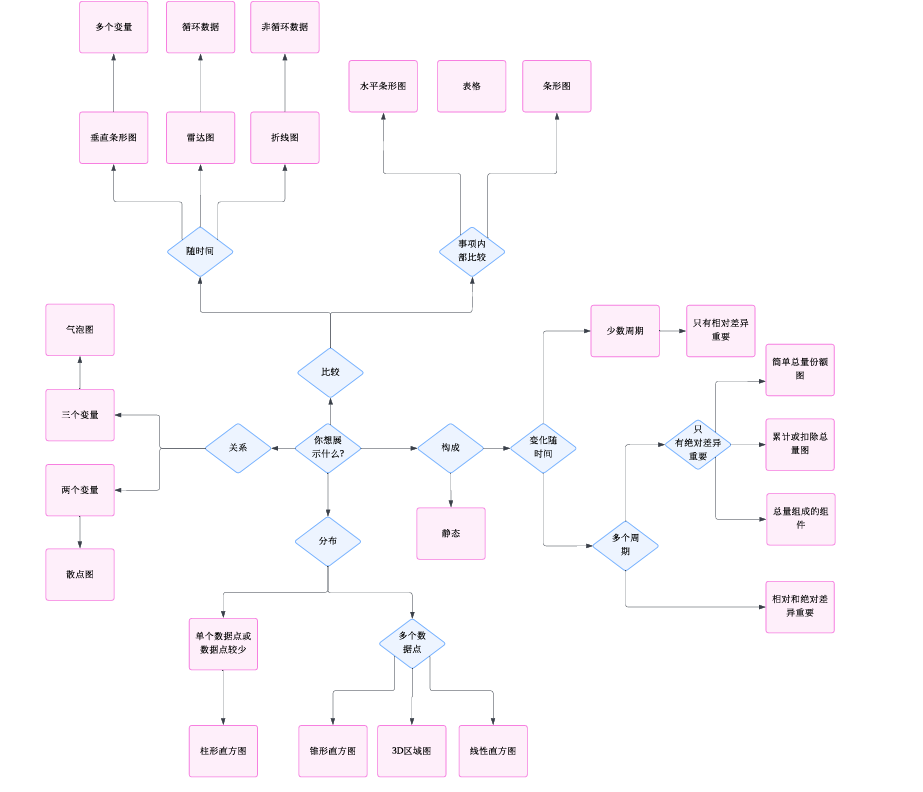


Explain：

We choose the line chart. Because we want to show the relationships between two variables , timeseries and quantity. The line chart can show the change in quantity over time.

**Part 3:**

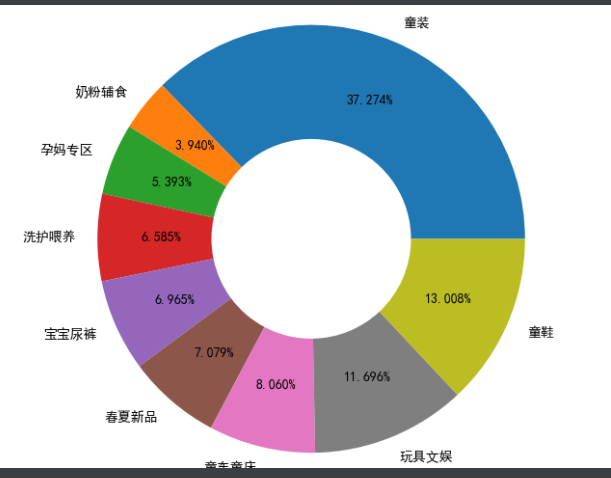
**Watch the video“How to choose the right chart for your data visualization”on Moodle and in this video, how to choose the right chart for your data visualization is introduced. Please first summarize what you learn from this video and then re-draw the graph that created by Andrew Avila by using a propreate software and make sure all contents are in Chinese. Note: export your graph into an image format (e.g. jpeg/png/bmp etc.) and attach it in the end of the Report 7.**



The point is what would you like to show with your data distribution relationships. We can divide four categories according to this standard , including relationship、composition、distribution、comparision. Then we can choose different kinds chart according to the numbers of variables.

**Part 4: research task**

**Find a data visualization chart from any source (internet, your Alipay report, etc.) and try to use a different method to represent the same data, summarize your results in the report.**



import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
  
  
df = pd.DataFrame({'子类目':['童装','奶粉辅食','孕妈专区','洗护喂养','宝宝尿裤','春夏新品','童车童床','玩具文娱','童鞋'],  
 '销售额':[29665,3135.4,4292.4,5240.9,5543.4,5633.8,6414.5,9308.1,10353]})  
  
name = np.array(df['子类目'])  
data = np.array(df['销售额'])  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.pie(data,radius=1.5,labels=name,  
 autopct='%0.3lf%%',pctdistance=0.5)  
plt.show()

