Python实验报告7--参考答案

```
班级：                    姓名：                    学号：
```

要求：

1. 在文档开头部分填写自己的信息；
2. 在每个题下方的代码块中书写该题的代码，并运行出结果；
3. 在2节课的时间内完成前5个题；打印为pdf文件并提交，文件名改为“Python实验7+班级姓名.pdf”。

一、实验目的：

1. 掌握Pandas中的Series和DataFrame的创建和基本操作方法。
2. 掌握Pandas中常用的文件读写、绘图和探索性分析的方法。

二、实验内容（在各题后书写程序和运行结果）

1. 已知一个开发小组6名成员，姓名依次为['Zhao', 'Qian', 'Sun', 'Li', 'Zhou', 'Wu']，月薪为[1.35, 1.43, 1.35, 1.67, 1.43, 1.35]，年终奖金为[3.5, 4.2, 3.6, 5.2, 4.0, 3.7]。

(1) 建立一个名为'staff'的DataFrame，存放上述数据，列标签分别为'name', 'monthly', 'bonus'.

(2) 在上述DataFrame后插入一列年龄数据[24, 25, 24, 27, 25, 24]，列标签为'age'.

In [1]:

```
import pandas as pd

staff = pd.DataFrame({'name':['Zhao', 'Qian', 'Sun', 'Li', 'Zhou', 'Wu'],
                      'monthly':[1.35, 1.43, 1.35, 1.67, 1.43, 1.35],
                       'bonus':[3.5, 4.2, 3.6, 5.2, 4.0, 3.7]})
staff['age']=[24, 25, 24, 27, 25, 24]
staff
```

Out[1]:

	name	monthly	bonus	age
0	Zhao	1.35	3.5	24
1	Qian	1.43	4.2	25
2	Sun	1.35	3.6	24
3	Li	1.67	5.2	27
4	Zhou	1.43	4.0	25
5	Wu	1.35	3.7	24

2. 在第1题的基础上，添加另4名成员的数据，姓名依次为['张三','李四','王五','赵六']，月薪依次为[1.80, 1.73, 1.87, 1.67]，年终奖金分别为[5.9, 5.2, 6.6, 4.8]，年龄分别为[27, 26, 27, 25]。再增加一列年收入数据，列标签为'yearly'，这里年收入 = 月薪 × 16 + 奖金。然后对新的DataFrame做概要性统计。

In [2]:
```
df1 = pd.DataFrame({'name':['张三','李四','王五','赵六'],
                    'monthly':[1.80, 1.73, 1.87, 1.67],
                    'bonus':[5.9, 5.2, 6.6, 4.8],
                    'age':[27, 26, 27, 25]})
df2 = pd.concat([staff, df1], ignore_index=True)
df2['yearly'] = df2['monthly']*16+df2['bonus']
df2
```

Out[2]:

	name	monthly	bonus	age	yearly
0	Zhao	1.35	3.5	24	25.10
1	Qian	1.43	4.2	25	27.08
2	Sun	1.35	3.6	24	25.20
3	Li	1.67	5.2	27	31.92
4	Zhou	1.43	4.0	25	26.88
5	Wu	1.35	3.7	24	25.30
6	张三	1.80	5.9	27	34.70
7	李四	1.73	5.2	26	32.88
8	王五	1.87	6.6	27	36.52
9	赵六	1.67	4.8	25	31.52

In [3]:
```
df2.describe()
```

Out[3]:

	monthly	bonus	age	yearly
count	10.000000	10.000000	10.000000	10.000000
mean	1.565000	4.670000	25.400000	29.710000
std	0.203484	1.050978	1.264911	4.287597
min	1.350000	3.500000	24.000000	25.100000
25%	1.370000	3.775000	24.250000	25.695000
50%	1.550000	4.500000	25.000000	29.300000
75%	1.715000	5.200000	26.750000	32.640000
max	1.870000	6.600000	27.000000	36.520000

3. 在第2题的基础上，按年龄对成员分组，一组不超过25，一组超过25。然后，对第2组按年收入从大到小输出各成员的数据，格式为"xx，xx岁，月薪x.x万，年终奖金x.x万，年收入x.x万."

In [4]:
```
df3 = df2[df2['yearly']<=25]
df4 = df2[df2['yearly']>25]
df5 = df4.sort_values(by="yearly", ascending=False)
for row_index, row in df5.iterrows():
    print('{0[0]}，{0[3]}岁，月薪{0[1]:.2f}万，'\
          '年终奖金{0[2]:.2f}万，年收入{0[4]:.2f}万'.format(row))
```

```
王五，27岁，月薪1.87万，年终奖金6.60万，年收入36.52万
张三，27岁，月薪1.80万，年终奖金5.90万，年收入34.70万
李四，26岁，月薪1.73万，年终奖金5.20万，年收入32.88万
Li，27岁，月薪1.67万，年终奖金5.20万，年收入31.92万
赵六，25岁，月薪1.67万，年终奖金4.80万，年收入31.52万
Qian，25岁，月薪1.43万，年终奖金4.20万，年收入27.08万
Zhou，25岁，月薪1.43万，年终奖金4.00万，年收入26.88万
Wu，24岁，月薪1.35万，年终奖金3.70万，年收入25.30万
Sun，24岁，月薪1.35万，年终奖金3.60万，年收入25.20万
Zhao，24岁，月薪1.35万，年终奖金3.50万，年收入25.10万
```

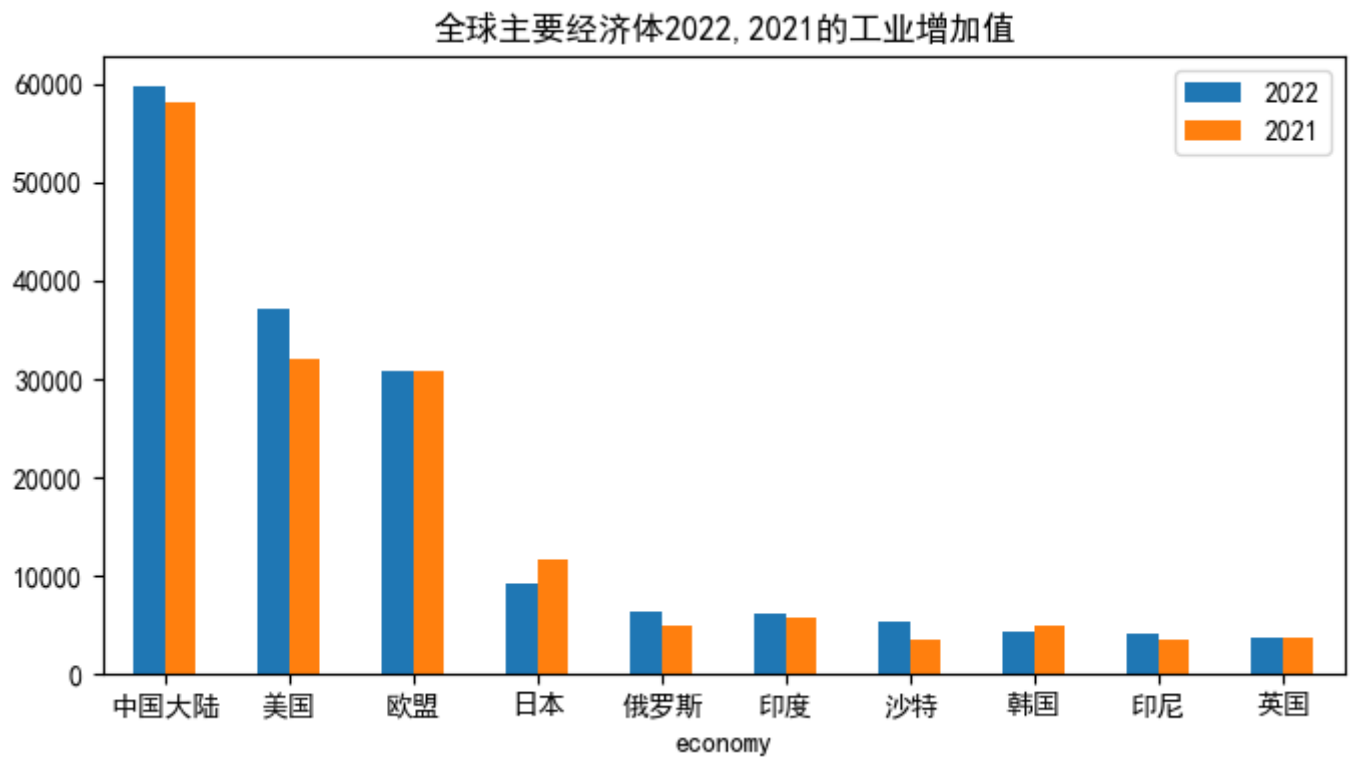
In [5]:
```
# 另一种迭代方式
for i in range(df5.shape[0]):
    print('{0[0]}，{0[3]}岁，月薪{0[1]:.2f}万，'\
          '年终奖金{0[2]:.2f}万，年收入{0[4]:.2f}万'.format(df5.iloc[i,:]))
```

```
王五，27岁，月薪1.87万，年终奖金6.60万，年收入36.52万
张三，27岁，月薪1.80万，年终奖金5.90万，年收入34.70万
李四，26岁，月薪1.73万，年终奖金5.20万，年收入32.88万
Li，27岁，月薪1.67万，年终奖金5.20万，年收入31.92万
赵六，25岁，月薪1.67万，年终奖金4.80万，年收入31.52万
Qian，25岁，月薪1.43万，年终奖金4.20万，年收入27.08万
Zhou，25岁，月薪1.43万，年终奖金4.00万，年收入26.88万
Wu，24岁，月薪1.35万，年终奖金3.70万，年收入25.30万
Sun，24岁，月薪1.35万，年终奖金3.60万，年收入25.20万
Zhao，24岁，月薪1.35万，年终奖金3.50万，年收入25.10万
```

4. 2022年全球主要经济体的工业增加值前10为 ['中国大陆', '美国', '欧盟', '日本', '俄罗斯', '印度', '沙特', '韩国', '印尼', '英国']，数额依次为 [59714.46, 37173.58, 30852.27, 9157.12, 6297.45, 6155.51, 5411.46, 4425.21,4169.40, 3798.07]；2021年这些经济体的工业增加值依次为 [58055.60, 32091.75, 30694.91, 11651.53, 5010.89, 5747.71, 3430.38, 4940.82, 3488.52, 3760.97]。请用Pandas绘制下列图形：(1) 在两组条形图中同时表现2022年和2021年前10大主要经济体的工业增加值，一个按国家分组，另一个按年份分组；(2)以两个饼图分别表现2022年和2021年的工业增加值。

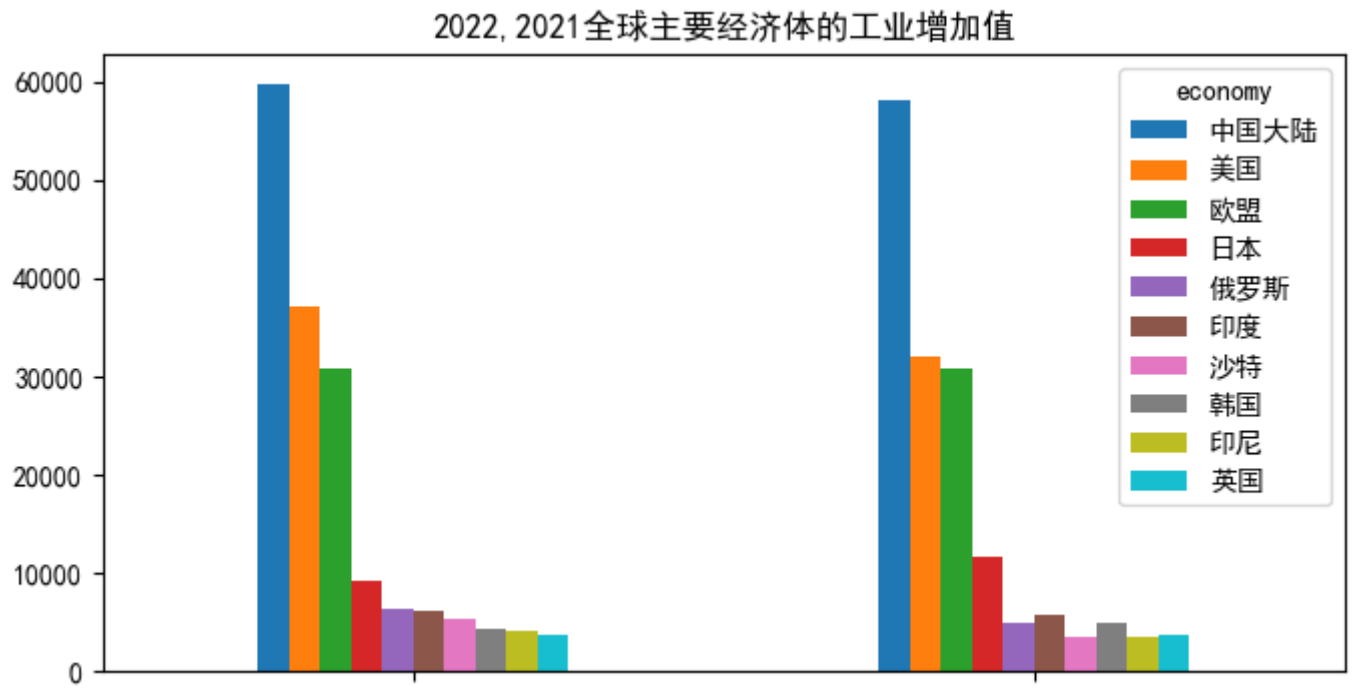
In [6]:
```
import pandas as pd
import matplotlib.pyplot as plt

# 产生数据帧
economy = ['中国大陆', '美国', '欧盟', '日本', '俄罗斯',
           '印度', '沙特', '韩国', '印尼', '英国']
indust_2022 = [59714.46, 37173.58, 30852.27, 9157.12, 6297.45,
               6155.51, 5411.46, 4425.21,4169.40, 3798.07]
indust_2021 = [58055.60, 32091.75, 30694.91, 11651.53, 5010.89,
               5747.71, 3430.38, 4940.82, 3488.52, 3760.97]
df = pd.DataFrame({'economy':economy, '2022':indust_2022, '2021': indust_2021})
```

In [7]:
```
# 设置中文显示
plt.rcParams['font.sans-serif'] = 'simhei'  # 设置中文字体为国标黑体
plt.rcParams['axes.unicode_minus'] = False  # 设置正常显示负号
# 按国家分组的条形图（以国家为x轴刻度）
df.plot.bar(x='economy', figsize=(8,4), rot=0,
            title='全球主要经济体2022,2021的工业增加值');
```



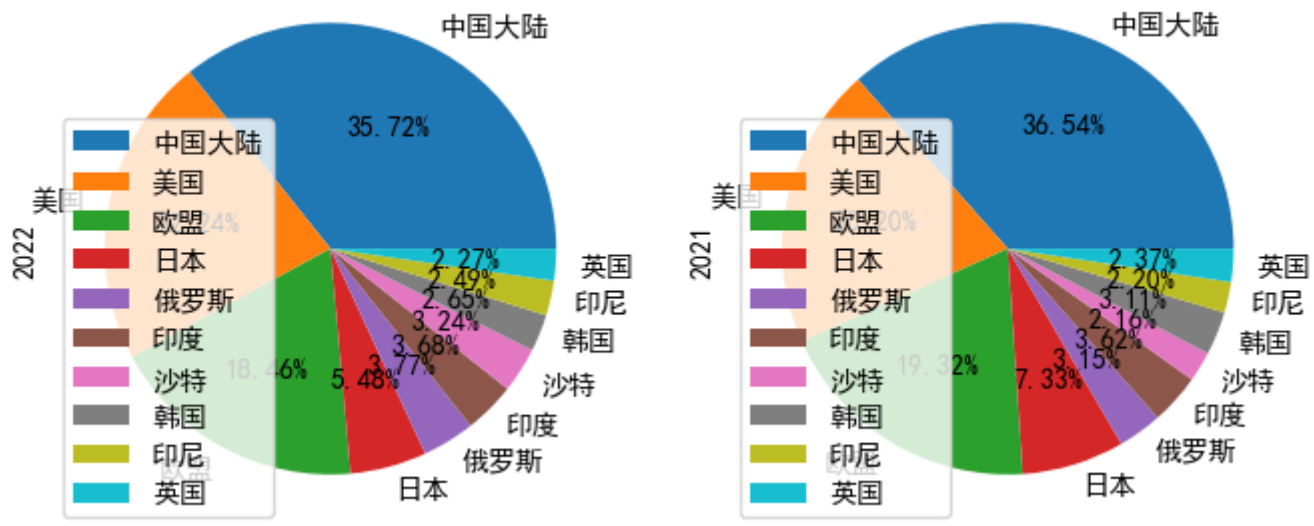
In [8]:

```
# 按年份分组的条形图（以年份为x轴刻度）
df1 = df.T  # 转置
df1.columns = df1.iloc[0] # 第0行作为列名
df1.iloc[1:,:].plot.bar(rot=0,figsize=(8,4),
                        title='2022,2021全球主要经济体的工业增加值');
```



In [9]:

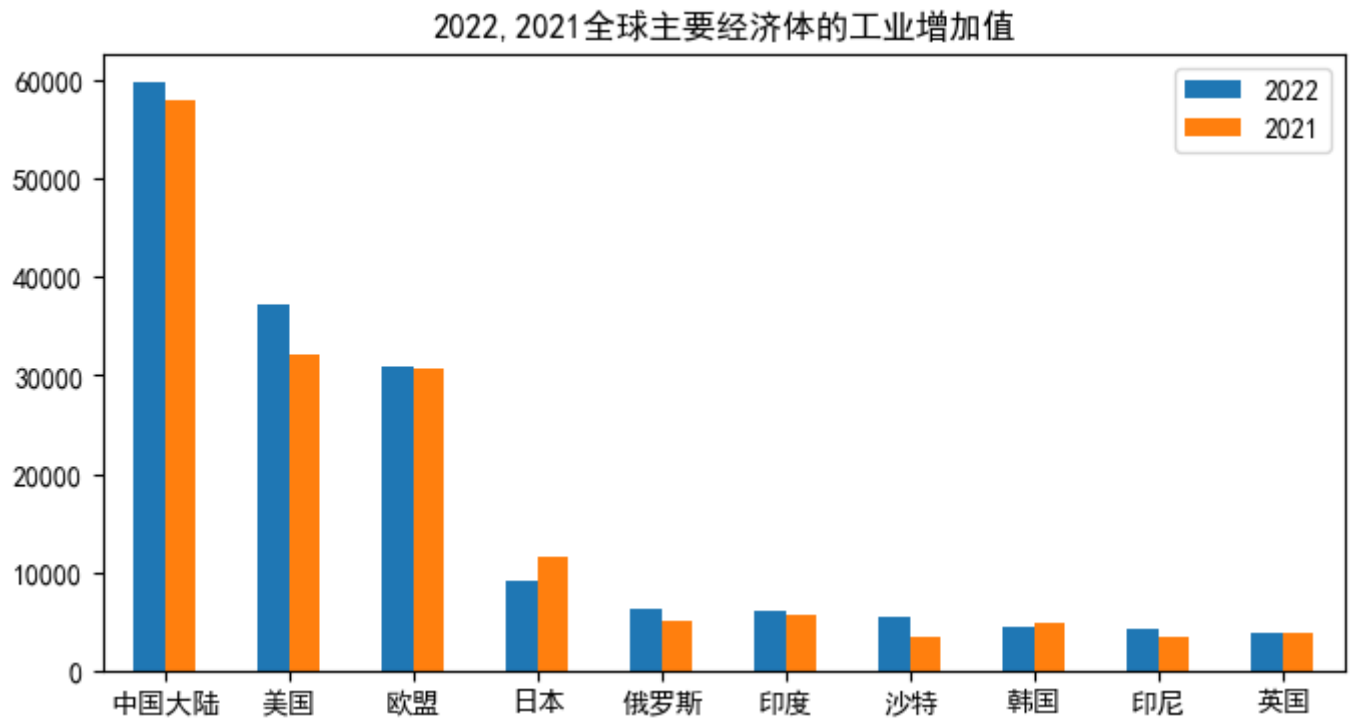
```
# 饼图
df.iloc[:,1:].plot.pie(subplots=True, labels=economy,
                       autopct="%.2f%%", figsize=(8,4));
```



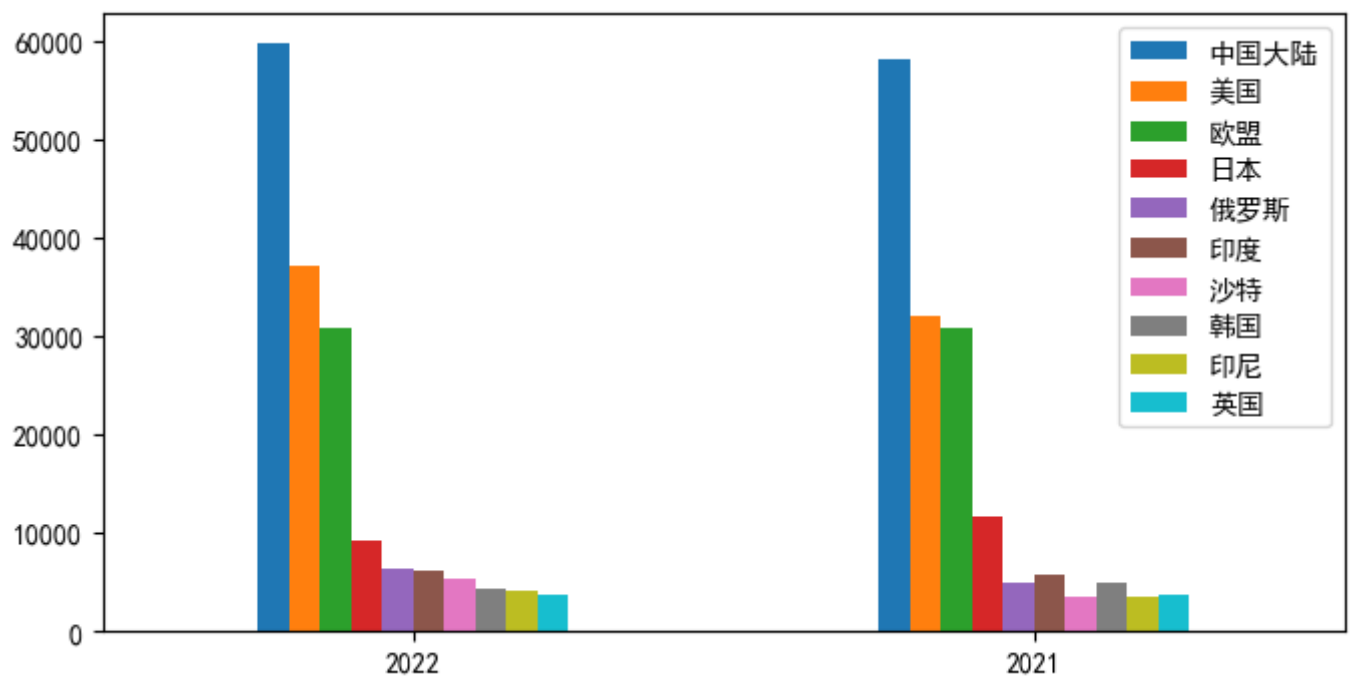
注意：上面创建的数据帧中第1列是经济体'economy'，这是一个字符列，在按经济体分组画条形图时可用x='economy'表明用这一列作为x轴的刻度标签。在按年份分组画条形图时，先将原数据帧转置，则第0行为字符串类型，可作为列名，在画分组条形图时，第0行不参与。而在画饼图时，原数据帧的第0列不参与，才能正确画图，且标注各扇形的标签要用 label=economy。

上述的过程比较麻烦，另一个方便的方法是在创建数据帧时直接将经济体'economy'作为行标签'index'.

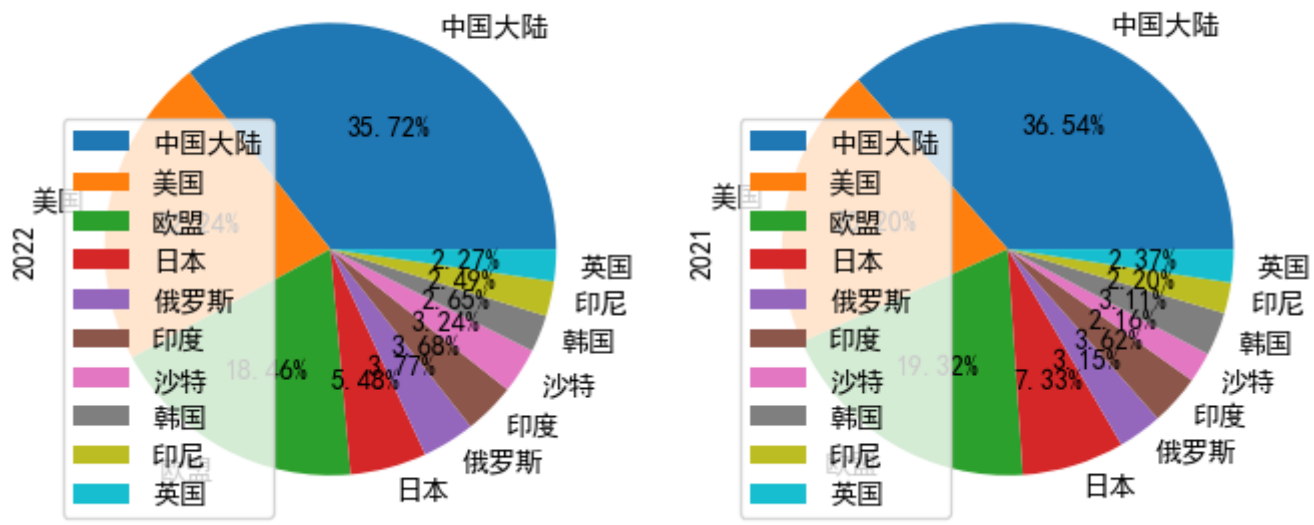
In [10]:
```
# 方法2
df1 = pd.DataFrame({'2022':indust_2022, '2021': indust_2021}, index=economy)
df1.plot.bar(figsize=(8,4), rot=0, title='2022,2021全球主要经济体的工业增加值');
```



In [11]:
```
df1.T.plot.bar(rot=0, figsize=(8,4));
```



In [12]:
```
df1.plot.pie(subplots=True, autopct="%.2f%%", figsize=(8,4));
```



5. 读取数据集'health.csv'，这是来自某个人群的健康数据。试按年龄分3组：小于45,[45,60)，60及以上，对3组人的各项指标绘制直方图。

In [13]:

```
import pandas as pd
import matplotlib.pyplot as plt

health = pd.read_csv('health.csv')
health.head()
```

Out[13]:

	age	totChol	sysBP	diaBP	BMI	heartRate	glucose
0	43	207	117.0	65.0	24.42	60	100
1	56	192	122.0	82.5	28.61	68	58
2	47	231	102.5	66.0	23.40	70	78
3	41	260	101.0	68.0	22.49	80	77
4	59	229	100.5	66.0	25.18	44	81

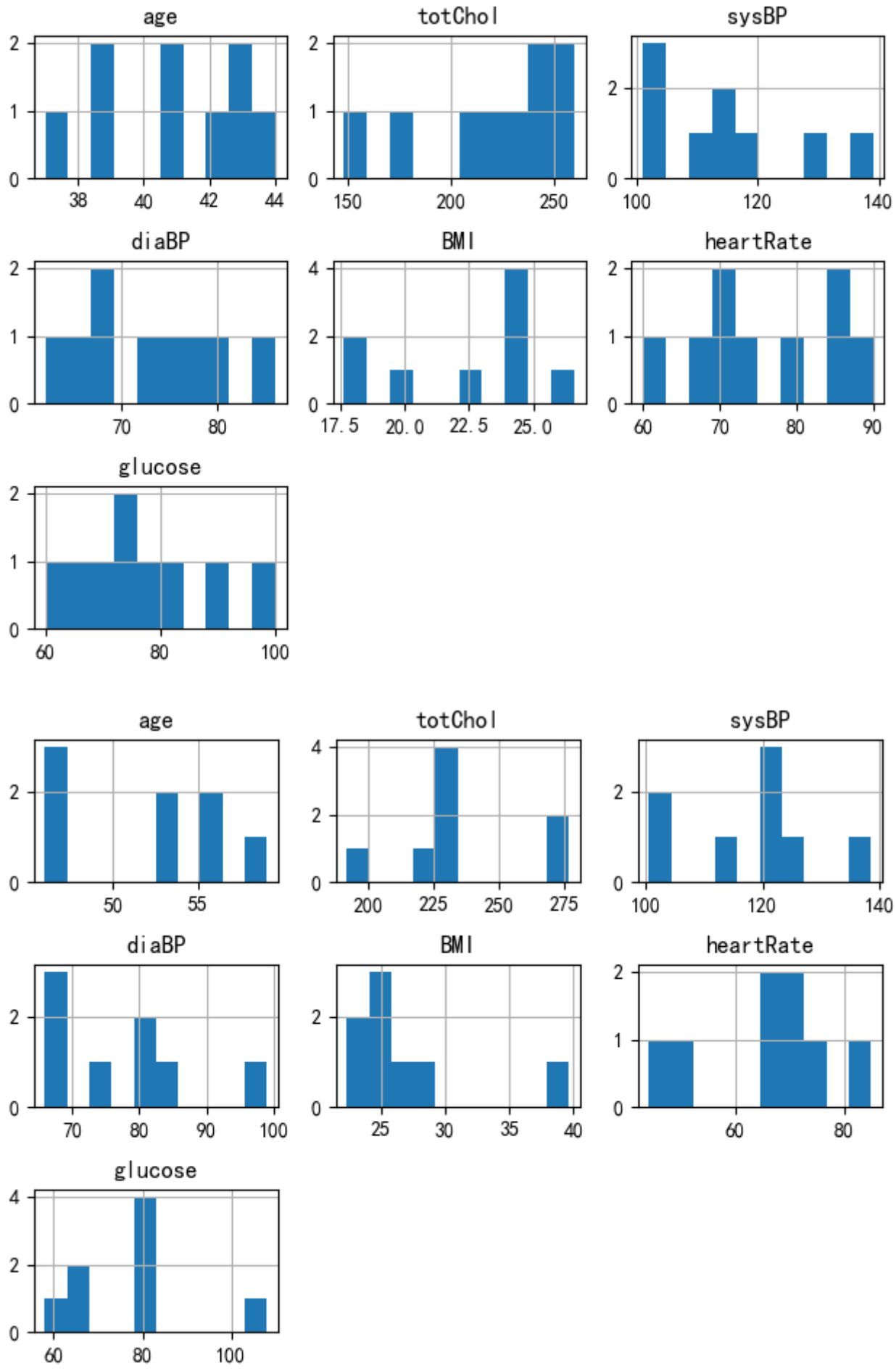
In [14]:

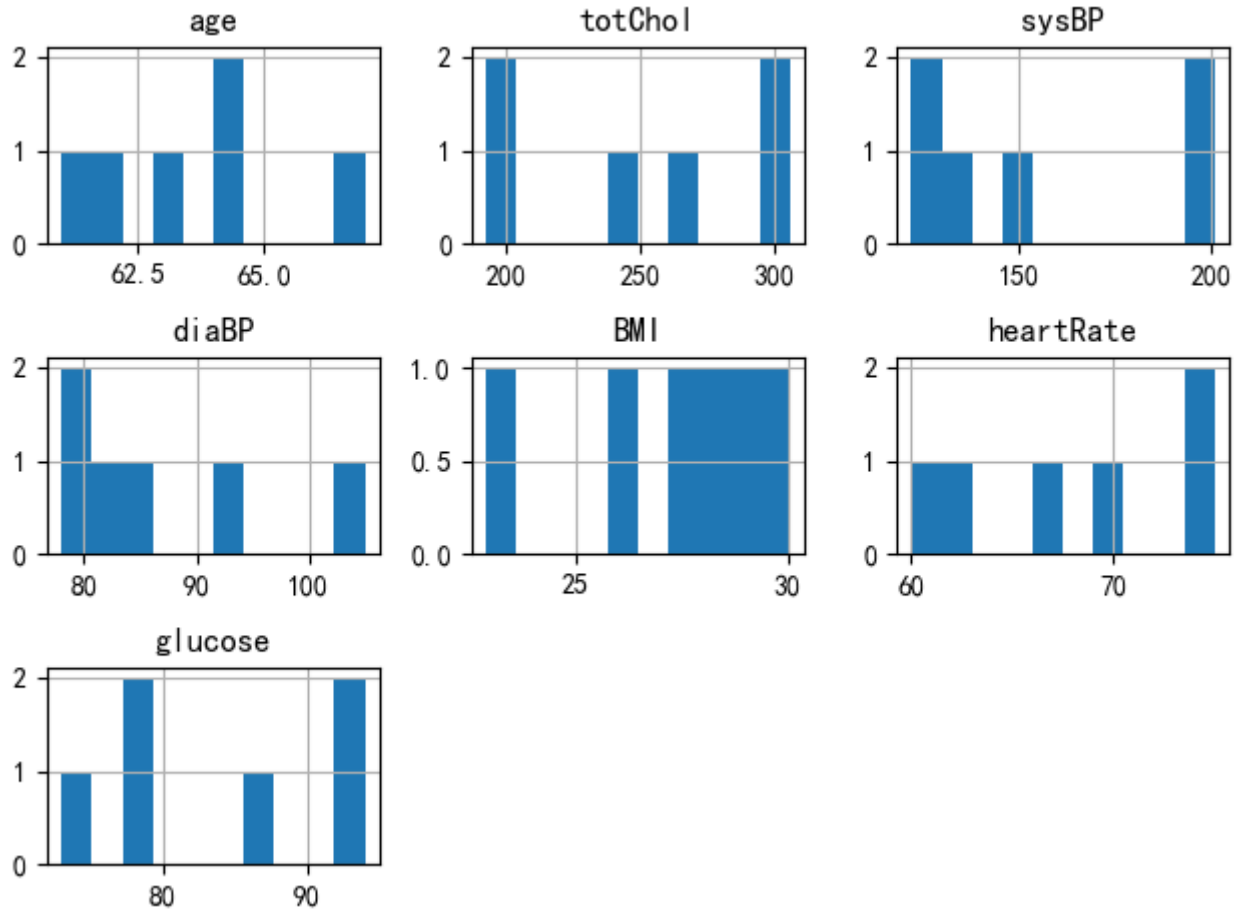
```
# 根据年龄对数据帧分组
# 方法1：按布尔值分组
h1 = health[health['age']<45]
h2 = health[np.logical_and(health['age']<60, health['age']>=45)]
h3 = health[health['age']>=60]
print(h1)
print(h2)
print(h3)
```

```
    age  totChol  sysBP  diaBP    BMI  heartRate  glucose
0    43      207  117.0   65.0  24.42         60      100
3    41      260  101.0   68.0  22.49         80       77
5    41      242  139.0   80.0  19.68         72       60
6    39      148  101.0   62.0  24.47         70       81
8    37      232  129.0   74.0  24.46         86       88
14   44      239  103.0   67.0  26.58         66       73
15   42      218  116.0   86.0  17.81         85       69
20   39      180  113.0   73.0  17.65         70       73
21   43      252  112.0   78.0  24.25         90       65
    age  totChol  sysBP  diaBP    BMI  heartRate  glucose
1    56      192  122.0   82.5  28.61         68       58
2    47      231  102.5   66.0  23.40         70       78
4    59      229  100.5   66.0  25.18         44       81
7    46      229  125.0   80.0  27.27         66       80
12   47      277  138.5   99.0  39.64         85       81
17   53      234  113.0   68.0  24.80         76      108
18   56      269  121.0   75.0  22.36         50       66
19   53      222  123.0   82.0  25.52         72       67
    age  totChol  sysBP  diaBP    BMI  heartRate  glucose
9    67      263  201.0   93.0  30.04         75       78
10   61      239  122.0   83.0  28.85         62       94
11   64      196  150.0   84.0  25.98         60       93
13   62      193  132.5   80.0  27.20         70       78
16   63      306  195.0  105.0  27.96         75       87
22   64      295  127.0   78.0  22.89         67       73
```

In [15]:

```
# 分组绘制直方图
h1.hist(); plt.tight_layout()
h2.hist(); plt.tight_layout()
h3.hist(); plt.tight_layout()
```





In [ ]:

```
# 方法2：用groupby分组，参数by用函数返回值
f = lambda x: sum([x>=n for n in (45, 60)])
grouped = health.groupby(f(health['age']))
for ind, group in grouped:
    print(ind)
    print(group)
# 对3组人的各项指标绘制直方图
for ind, group in grouped:
    group.hist();
    plt.tight_layout()
```

In [ ]:

```
# 方法3：使用 groupby() 结合 cut()分组

health['group'] = pd.cut(health['age'], [0, 45 ,60, 100]) # 增加一列表示分组
grouped = health.groupby('group')
for ind, group in grouped:
    print(ind)
    print(group)
# 对3组人的各项指标绘制直方图
for ind, group in grouped:
    group.hist();
    plt.tight_layout()
```

6. 读取数据集'iris.csv'，这是一个鸢尾花分类的数据集。共有3个品种：Iris Setosa（山鸢尾）、Iris Versicolour（杂色鸢尾），Iris Virginica（维吉尼亚鸢尾），各50个样本，4个特征：Sepal.Length（花萼长度）、Sepal.Width（花萼宽度）、Petal.Length（花瓣长度）、Petal.Width（花瓣宽度）。前4列为特征数据，最后一列为品种（variety）代号。请用Pandas对该数据集完成下列工作：

   （1）对4个特征数据做概要性统计。

   （2）对4个特征数据分品种做概要性统计。

In [18]:
```
import pandas as pd

names = ['Sepal.length', 'Sepal.width', 'Petal.length', 'Petal.width', 'variety']
iris = pd.read_csv('iris.csv', skiprows=1, names=names)
iris.iloc[:,:-1].describe()
```

Out[18]:

	Sepal.length	Sepal.width	Petal.length	Petal.width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [19]:
```
# 分品种做概要性统计
grouped = iris.groupby("variety")
for name, group in grouped:
    print('品种', name, '概要性统计：')
    print(group.iloc[:,:-1].describe())
```

```
品种 0 概要性统计：
       Sepal.length  Sepal.width  Petal.length  Petal.width
count      50.00000    50.000000     50.000000     50.00000
mean        5.00600     3.418000      1.464000      0.24400
std         0.35249     0.381024      0.173511      0.10721
min         4.30000     2.300000      1.000000      0.10000
25%         4.80000     3.125000      1.400000      0.20000
50%         5.00000     3.400000      1.500000      0.20000
75%         5.20000     3.675000      1.575000      0.30000
max         5.80000     4.400000      1.900000      0.60000
品种 1 概要性统计：
       Sepal.length  Sepal.width  Petal.length  Petal.width
count     50.000000    50.000000     50.000000    50.000000
mean       5.936000     2.770000      4.260000     1.326000
std        0.516171     0.313798      0.469911     0.197753
min        4.900000     2.000000      3.000000     1.000000
25%        5.600000     2.525000      4.000000     1.200000
50%        5.900000     2.800000      4.350000     1.300000
75%        6.300000     3.000000      4.600000     1.500000
max        7.000000     3.400000      5.100000     1.800000
品种 2 概要性统计：
       Sepal.length  Sepal.width  Petal.length  Petal.width
count      50.00000    50.000000     50.000000     50.00000
mean        6.58800     2.974000      5.552000      2.02600
std         0.63588     0.322497      0.551895      0.27465
min         4.90000     2.200000      4.500000      1.40000
25%         6.22500     2.800000      5.100000      1.80000
50%         6.50000     3.000000      5.550000      2.00000
75%         6.90000     3.175000      5.875000      2.30000
max         7.90000     3.800000      6.900000      2.50000
```

In [20]:
```
# 另一种更简单的方法
iris.groupby("variety").describe().T
```

Out[20]:

	variety	0	1	2
Sepal.length	**count**	50.000000	50.000000	50.000000
	mean	5.006000	5.936000	6.588000
	std	0.352490	0.516171	0.635880
	min	4.300000	4.900000	4.900000
	25%	4.800000	5.600000	6.225000
	50%	5.000000	5.900000	6.500000
	75%	5.200000	6.300000	6.900000
	max	5.800000	7.000000	7.900000
Sepal.width	**count**	50.000000	50.000000	50.000000
	mean	3.418000	2.770000	2.974000
	std	0.381024	0.313798	0.322497
	min	2.300000	2.000000	2.200000
	25%	3.125000	2.525000	2.800000
	50%	3.400000	2.800000	3.000000
	75%	3.675000	3.000000	3.175000
	max	4.400000	3.400000	3.800000
Petal.length	**count**	50.000000	50.000000	50.000000
	mean	1.464000	4.260000	5.552000
	std	0.173511	0.469911	0.551895
	min	1.000000	3.000000	4.500000
	25%	1.400000	4.000000	5.100000
	50%	1.500000	4.350000	5.550000
	75%	1.575000	4.600000	5.875000
	max	1.900000	5.100000	6.900000
Petal.width	**count**	50.000000	50.000000	50.000000
	mean	0.244000	1.326000	2.026000
	std	0.107210	0.197753	0.274650
	min	0.100000	1.000000	1.400000
	25%	0.200000	1.200000	1.800000
	50%	0.200000	1.300000	2.000000
	75%	0.300000	1.500000	2.300000
	max	0.600000	1.800000	2.500000

7. （选做题）在第6题的基础上，用Pandas对该数据集完成下列工作：

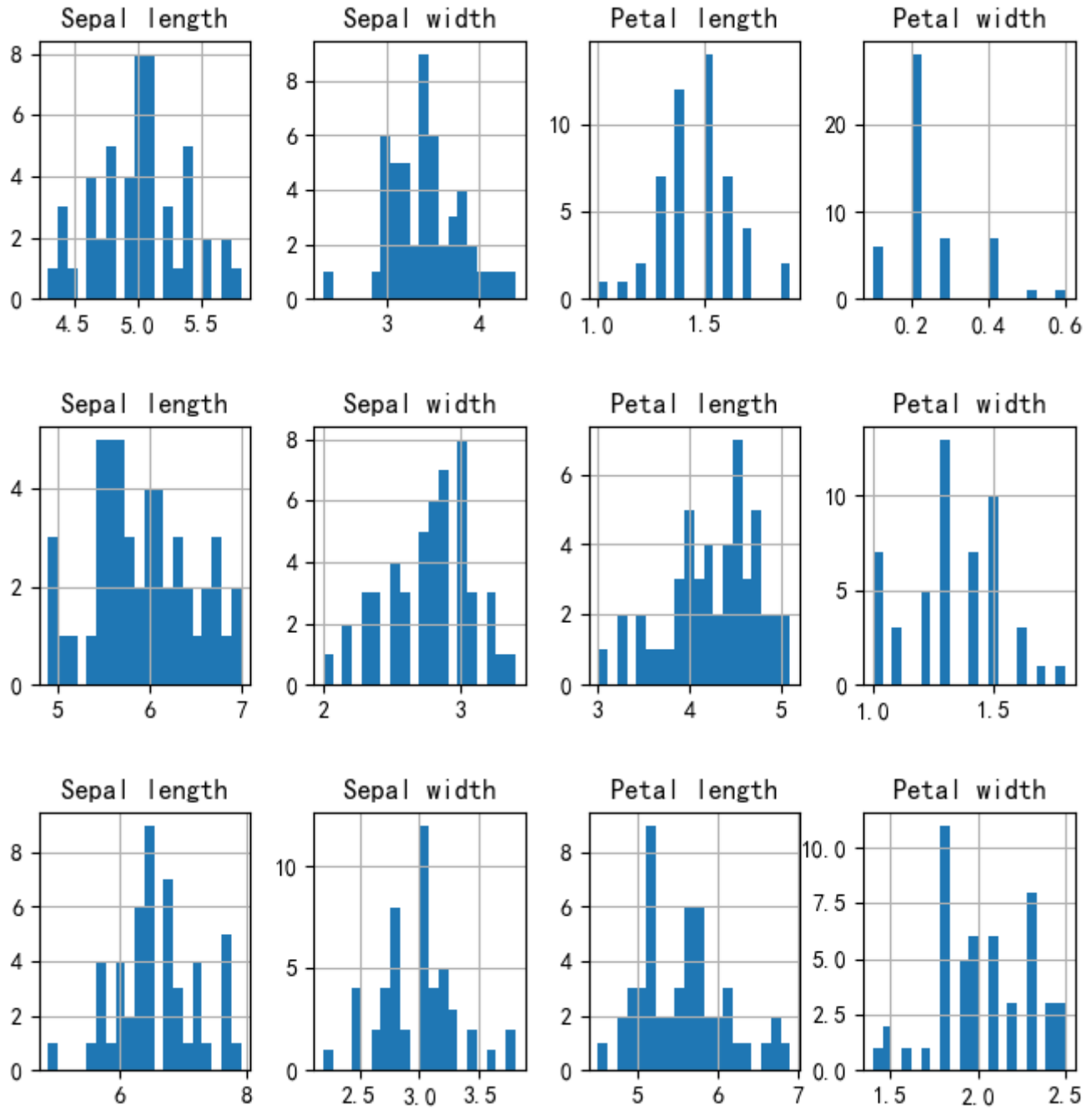
（1）对4个特征数据分品种绘制直方图，同一品种的不同特征的直方图放在同一图像窗口中。

（2）对4个特征数据分品种绘制直方图，不同品种的同一特征的直方图放在同一图形窗口中。

（3）对这些特征两两组合（如花萼长度做x轴，花萼宽度做y轴）绘制散点图，对不同品种在同一张图上用不同颜色的点表示。

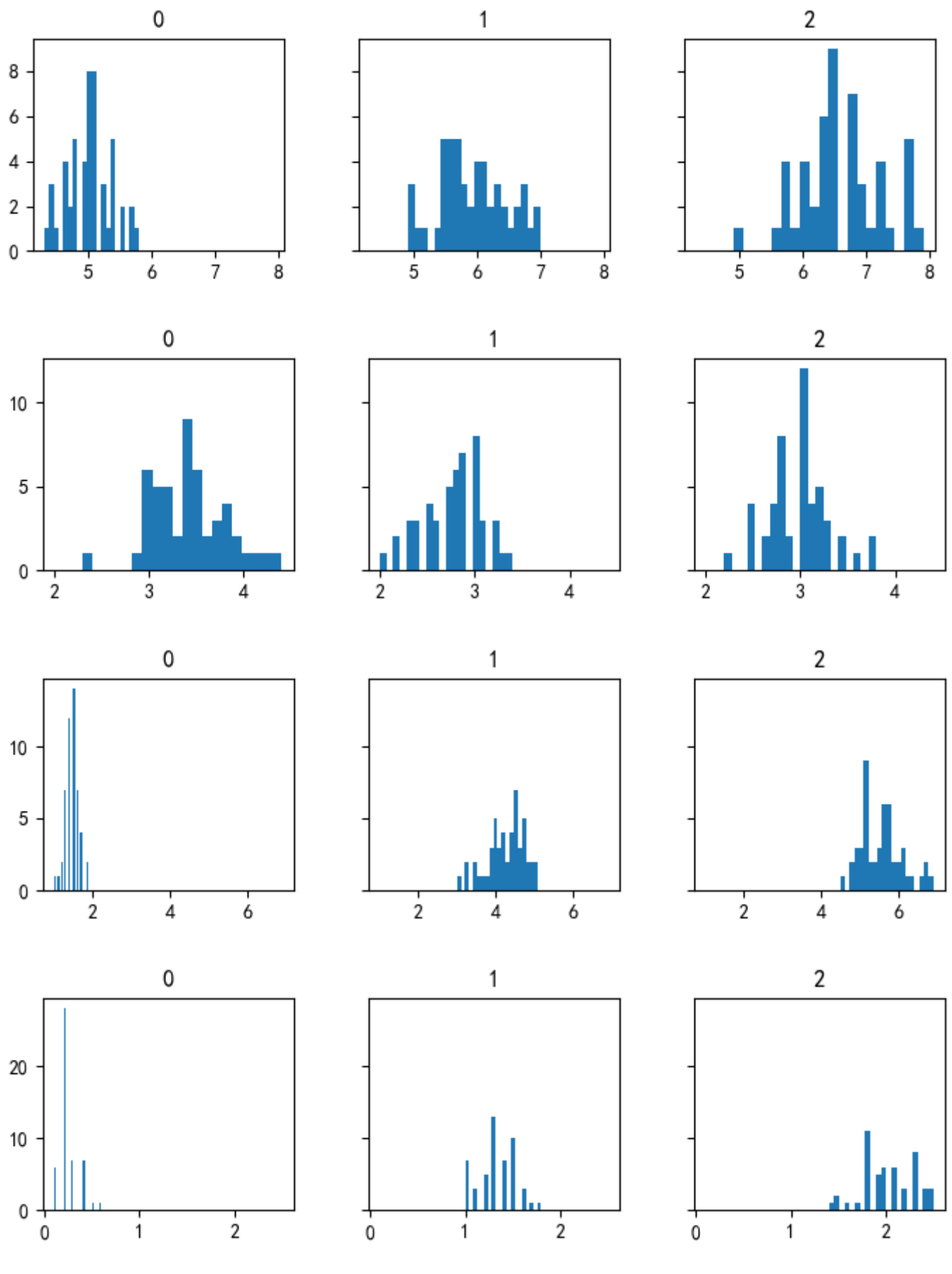
In [21]:

```
import pandas as pd
import matplotlib.pyplot as plt

names = ['Sepal length', 'Sepal width', 'Petal length', 'Petal width', 'variety']
iris = pd.read_csv('iris.csv', skiprows=1, names=names)
# 问题（1）
grouped = iris.groupby("variety")
for name, group in grouped:
    group.iloc[:,:-1].hist(figsize=(8,2), layout=(1,4), bins=20)
plt.show()
```



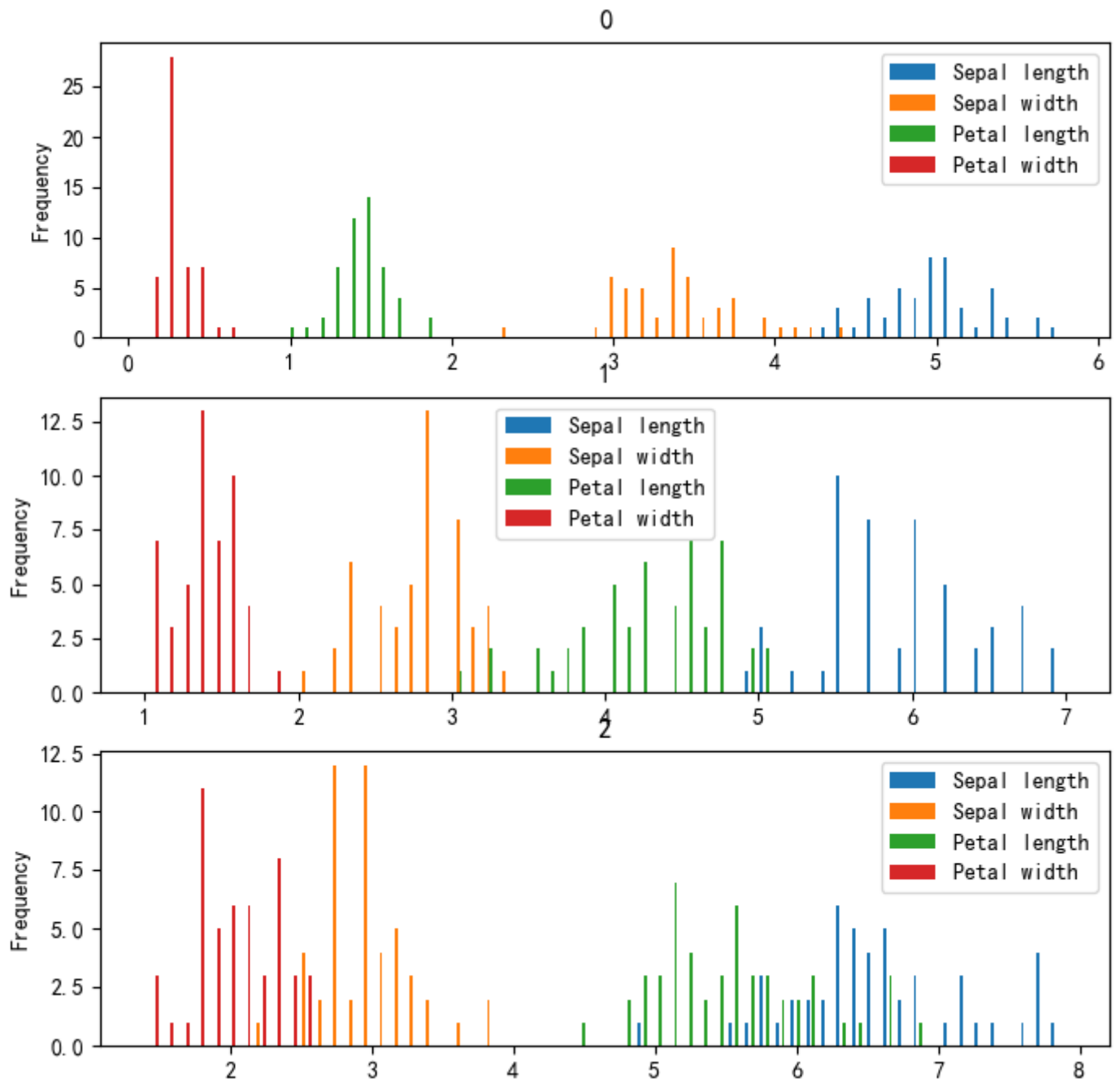
In [22]:

```
# 问题（2）
for k in range(4):
    fig, ax = plt.subplots(1,3,figsize=(8,2), sharex=True, sharey=True) # 指定坐标系
    iris.iloc[:,k].hist(by=iris['variety'], bins=20, ax=ax, rot=0)  # 通过参数'by'分组
plt.show()
```



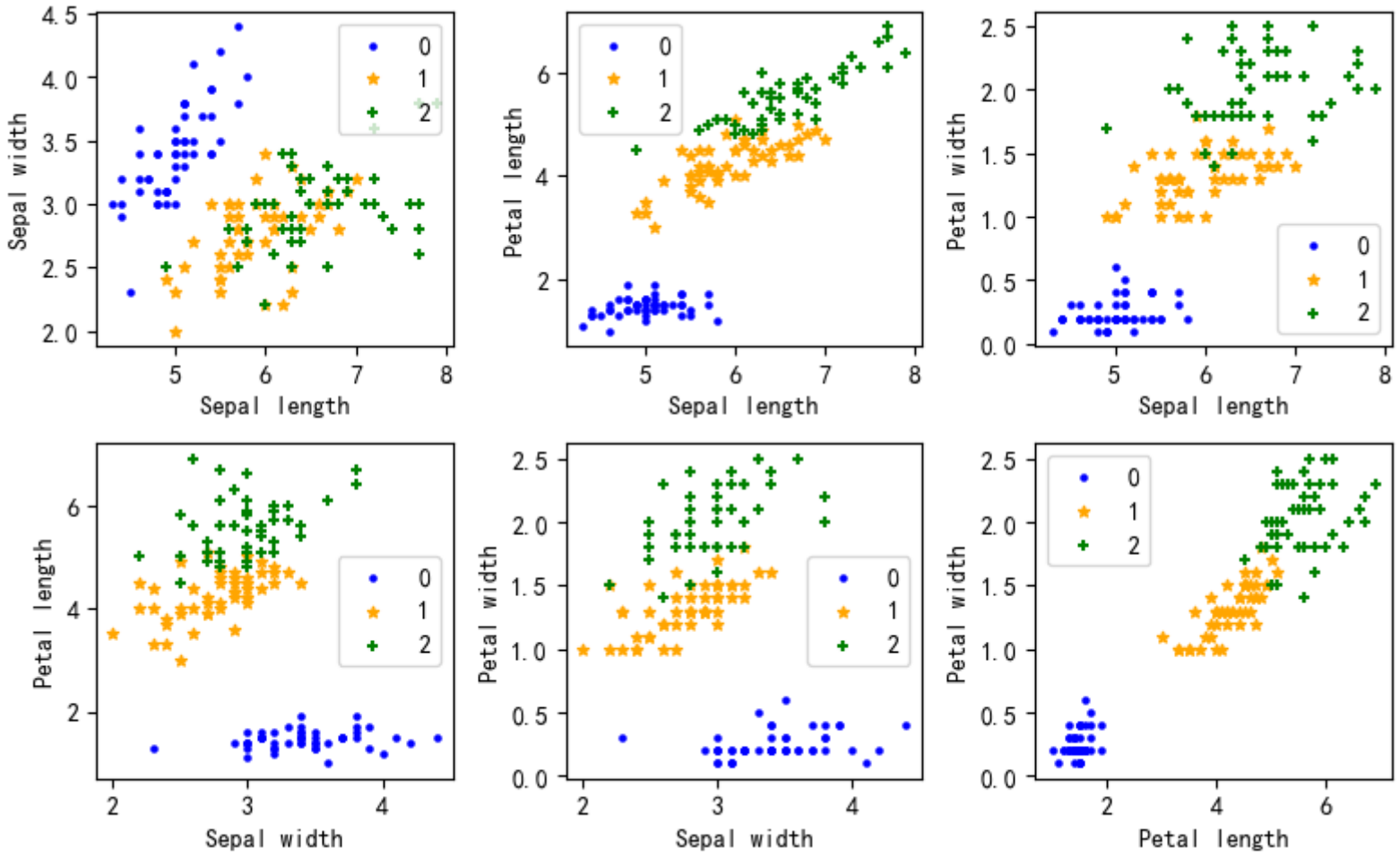
In [23]:

```
# 问题（2）的另一种显示
iris.plot.hist(by='variety', figsize=(8,8), bins=60, rot=0)
plt.show()
```



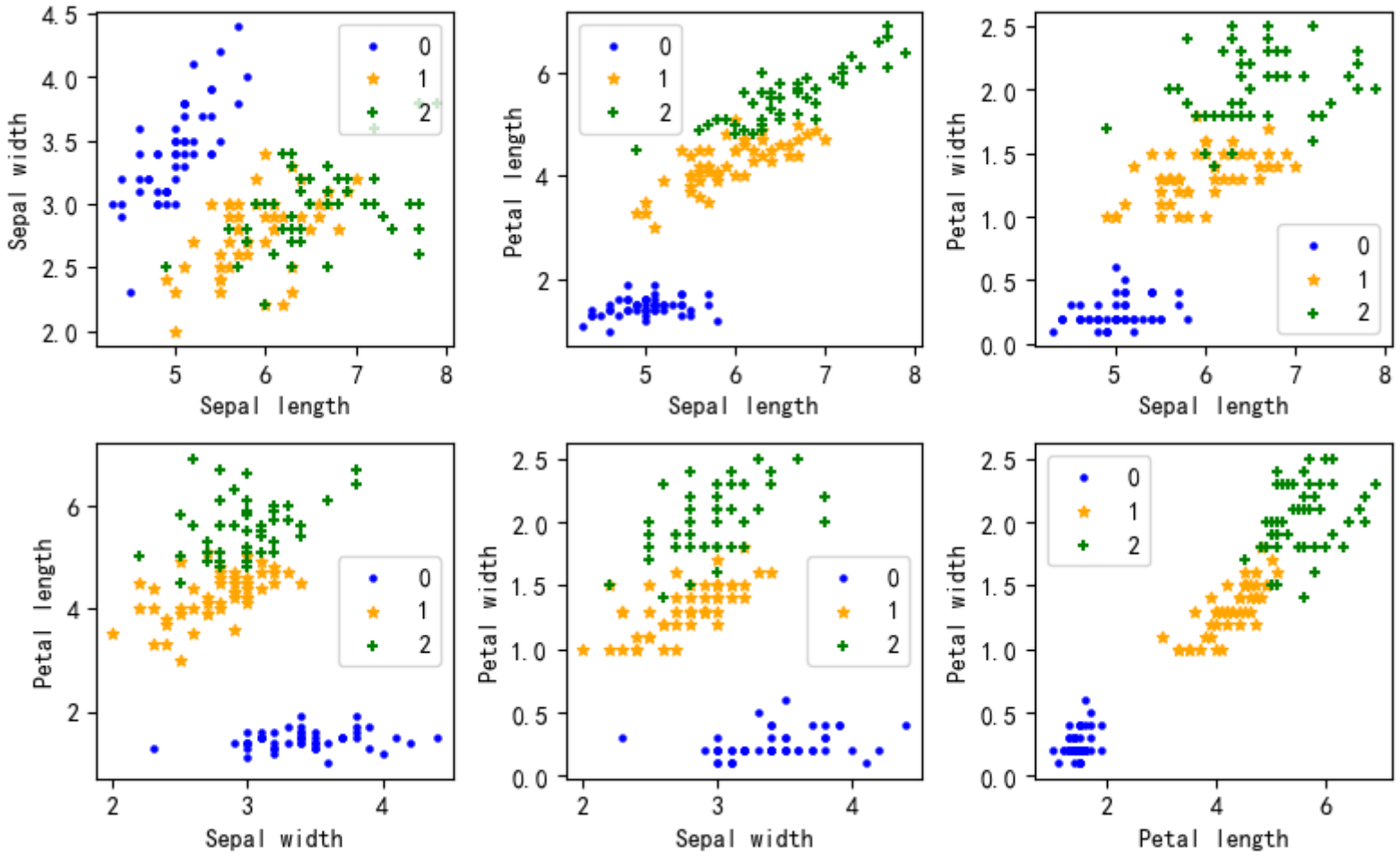
In [24]:

```
# 问题（3）
colors = ['blue', 'orange', 'green']
markers = ['.', '*', '+']
fig = plt.figure(figsize=(8, 5))  # 创建图形窗口
grouped = iris.groupby("variety")  # 按品种分组
i = 0
for k in range(3):    # 第1个特征
    for n in range(k+1, 4):  # 第2个特征
        i = i+1
        ax = plt.subplot(2,3,i)  # 指定坐标系
        for name, group in grouped:
            group.plot.scatter(x=names[k], y=names[n], ax=ax,
                               color=colors[name], marker=markers[name], label=name)
plt.tight_layout()
plt.show()
```



In [25]:

```
# 问题（3）

colors = ['blue', 'orange', 'green']
markers = ['.', '*', '+']
fig = plt.figure(figsize=(8,5))  # 创建图形窗口
i = 0
for k in range(3):    # 第1个特征
    for n in range(k+1, 4):  # 第2个特征
        i = i+1
        ax = plt.subplot(2,3,i)  # 指定坐标系
        for m in range(3):  # 分3次绘制散点图
            iris[iris['variety']==m].plot.scatter(x=names[k], y=names[n],
                        ax=ax, color=colors[m], marker=markers[m], label=m)
plt.tight_layout()
plt.show()
```



8. （选做题）已知某班在大一大二两学年的成绩表"1班成绩表.xlsx"，对其数据重新排列并计算。要求：

(1) 对表格增加一列"学分"，学分的计算方法是每16学时算1学分。然后将数据重新排列，使同一学生的全部信息放在同一行；

(2) 对每一个学生，计算其已获得的学分和不及格学分，将这些数据添加到表格中；

(3) 对每一个学生的成绩，按课程类别汇总，求各类课程的平均分、学分加权平均分，添加到表格中；

(4) 又已知绩点的计算方法是：绩点=分数/10-5,低于60分则绩点为0，计算学生每门课程的绩点，再以该课程的学分计算平均学分绩点。然后对成绩表按平均学分绩点从大到小排序，再将成绩表保存为"scores.xlsx".

```
    原有的(1)中的“将数据重新排列，使同一学生的全部信息放在同一行”，任务取消，因不同的课程名太多，达到145个。
    从(2)开始，另建一个数据帧，容纳原有的姓名、性别，及新产生的已获学分、不及格学分、各类课程平均分、学分加权平均分、平均学分绩点。最后按平均学分绩点排序，再保存为excel文件。
    问题(3)中的学分加权平均分只计算总的学分加权平均分，不算各类课程的学分加权平均分。
```

In [26]:

```
import pandas as pd
import numpy as np

# 读入文件
xlfile = r'1班成绩表.xlsx'
# ================以下用pandas读取excel文件数据======================
df = pd.read_excel(xlfile,index_col=0)
df.head()
```

Out[26]:

	学年	学期	姓名	性别	课程代码	课程名称	开课学院	考核方式	课程性质	平时成绩	期中成绩	期末成绩	实验成绩	成绩性质	总评成绩	折算成绩	补考成绩	课程类别	总学时
0	2022-2023	2	赵一	男	13222301x0	测量学	滨海农业学院	考试	必修	95.50	NaN	93	NaN	正常考试	94	94	NaN	专业基础课	56
1	2022-2023	1	赵一	男	13232157x0	园林工程制图	滨海农业学院	NaN	必修	80	NaN	72	NaN	正常考试	76	76	NaN	专业基础课	64
2	2022-2023	2	赵一	男	13232410x0	园林设计初步	滨海农业学院	考试	必修	NaN	NaN	60	81.0	正常考试	71	71	NaN	专业基础课	56
3	2022-2023	1	赵一	男	13233222x0	计算机辅助园林设计	滨海农业学院	NaN	必修	90	NaN	70	NaN	正常考试	良好	85	NaN	教学实验与实训	32
4	2022-2023	1	赵一	男	13281130x0	园林专业导论	滨海农业学院	NaN	必修	95	NaN	93	NaN	正常考试	94	94	NaN	专业基础课	16

In [27]:

```
# （1）对表格增加一列"学分"
df['学分'] = df['总学时']/16
df.head()
```

Out[27]:

	学年	学期	姓名	性别	课程代码	课程名称	开课学院	考核方式	课程性质	平时成绩	期中成绩	期末成绩	实验成绩	成绩性质	总评成绩	折算成绩	补考成绩	课程类别	总学时	学分
0	2022-2023	2	赵一	男	13222301x0	测量学	滨海农业学院	考试	必修	95.50	NaN	93	NaN	正常考试	94	94	NaN	专业基础课	56	3.5
1	2022-2023	1	赵一	男	13232157x0	园林工程制图	滨海农业学院	NaN	必修	80	NaN	72	NaN	正常考试	76	76	NaN	专业基础课	64	4.0
2	2022-2023	2	赵一	男	13232410x0	园林设计初步	滨海农业学院	考试	必修	NaN	NaN	60	81.0	正常考试	71	71	NaN	专业基础课	56	3.5
3	2022-2023	1	赵一	男	13233222x0	计算机辅助园林设计	滨海农业学院	NaN	必修	90	NaN	70	NaN	正常考试	良好	85	NaN	教学实验与实训	32	2.0
4	2022-2023	1	赵一	男	13281130x0	园林专业导论	滨海农业学院	NaN	必修	95	NaN	93	NaN	正常考试	94	94	NaN	专业基础课	16	1.0

In [28]:

```
#（2）对每一个学生，计算其已获得的学分和不及格学分，将这些数据添加到表格中
# 步骤1：在原数据帧中，求每一项成绩是否获得学分

df["折算成绩"]=df["折算成绩"].astype(np.float64)
ind = df["补考成绩"]=="缺考"
df.loc[ind,"补考成绩"] = -1  # 将缺考换成-1
df["补考成绩"] = df["补考成绩"].fillna(0) # 将NaN换成0
df["补考成绩"] = df["补考成绩"].astype(np.float64)
df["最终成绩"] = df[["折算成绩", "补考成绩"]].T.max()
ifornot = df["最终成绩"]>=60
get = df["学分"]*ifornot # 获得学分
noget = df["学分"]*(~ifornot)  # 未获得学分
```

In [29]:
```
# (2)步骤2：对每个学生已获得的学分和未获得的学分分别求和
names = df["姓名"].unique()
allget = np.zeros(len(names))
allnoget = np.zeros(len(names))
sex = []
for i, name in enumerate(names):
    ind = df["姓名"] == name
    allget[i] = sum(get[ind])
    allnoget[i] = sum(noget[ind])
```

In [30]:
```
# (2)步骤3：生成一个新数据帧，包含姓名和性别两列及已获学分、未获学分
ind = [list(df["姓名"]).index(name) for name in names]
sex = df["性别"][ind]
stus = pd.DataFrame({"姓名":names, "性别":sex, "已获学分":allget, "未获学分":allnoget})
stus.head()
```

Out[30]:

	姓名	性别	已获学分	未获学分
0	赵一	男	142.3125	0.0
68	赵二	男	118.9375	0.0
125	赵三	男	114.3125	0.0
180	赵四	男	115.3125	0.0
234	赵五	女	118.3750	0.0

In [31]:
```
# (3)对每一个学生的成绩，按课程类别汇总，求各类课程的平均分、学分加权平均分，添加到表格中
cls = df["课程类别"].unique()
avg = np.zeros((len(names),len(cls)))
stus[cls] = avg # 在stus中添加新列
avg_w = []
```

In [32]:
```
# 求各类课程的平均分，放在stus中
for name in names:
    tmp = df[df["姓名"]==name]
    # 求学分加权平均分
    avg_w.append(np.sum(tmp["最终成绩"]*tmp["学分"])/np.sum(tmp["学分"]))
    # 求各类课程的平均分，
    for c in cls:
        if c in tmp[tmp['姓名']==name]["课程类别"].values:
            stus.loc[stus['姓名']==name,c]=tmp[tmp["课程类别"]==c]["最终成绩"].mean()
stus['学分加权平均分'] = avg_w
stus.head()
```

Out[32]:

	姓名	性别	已获学分	未获学分	专业基础课	教学实验与实训	通识教育课	通识教育核心课	思想政治理论课	专业课	通识教育拓展课	通识实践与创新训练	课程与专业实习	公共课	学分加权平均分
0	赵一	男	142.3125	0.0	89.166667	90.00	87.153846	85.333333	88.666667	89.333333	95.600000	88.333333	88.333333	0.0	88.283267
68	赵二	男	118.9375	0.0	82.916667	89.50	78.583333	86.750000	85.272727	82.166667	88.833333	90.000000	0.000000	0.0	82.088807
125	赵三	男	114.3125	0.0	80.583333	89.50	82.461538	86.000000	88.363636	84.000000	91.000000	90.000000	0.000000	0.0	82.283762
180	赵四	男	115.3125	0.0	77.333333	87.75	79.583333	79.333333	90.181818	80.571429	85.000000	90.000000	0.000000	0.0	78.821680
234	赵五	女	118.3750	0.0	73.000000	87.50	85.750000	86.333333	90.272727	81.500000	96.166667	85.000000	0.000000	93.0	80.942978

In [33]:

```
# (4)计算学生每门课程的绩点，再以该课程的学分计算平均学分绩点
df['绩点'] = [x/10-5 if x>=60 else 0 for x in df['最终成绩']]
gpa_avg = []
for name in names:
    tmp = df[df["姓名"]==name]
    gpa_avg.append(np.sum(tmp["绩点"]*tmp["学分"])/np.sum(tmp["学分"]))
stus['平均学分绩点'] = gpa_avg
# 按平均学分绩点排序
stus = stus.sort_values(by="平均学分绩点",ascending=False)
# 将成绩表保存为"scores.xlsx"
stus.to_excel("scores.xlsx")
stus.head()
```

Out[33]:

	姓名	性别	已获学分	未获学分	专业基础课	教学实验与实训	通识教育课	通识教育核心课	思想政治理论课	专业课	通识教育拓展课	通识实践与创新训练	课程与专业实习	公共课	学分加权平均分	平均学分绩点
1085	李五	男	116.5000	0.0	89.666667	84.75	86.750000	92.500000	90.545455	89.500000	89.400000	95.000000	0.000000	0.0	88.875536	3.887554
0	赵一	男	142.3125	0.0	89.166667	90.00	87.153846	85.333333	88.666667	89.333333	95.600000	88.333333	88.333333	0.0	88.283267	3.828327
801	孙五	男	116.5625	0.0	89.833333	85.00	86.583333	87.500000	88.909091	89.000000	85.400000	95.000000	0.000000	0.0	88.080965	3.808097
406	钱三	女	121.6250	0.0	82.916667	89.50	83.083333	92.750000	89.636364	85.428571	95.142857	90.000000	0.000000	0.0	85.173176	3.517318
1535	吴三	男	120.9375	0.0	86.250000	88.00	84.416667	91.000000	84.727273	83.285714	88.400000	85.000000	0.000000	0.0	84.799483	3.479948