Control Statement جمل التحكم والحلقات التكرارية

In [181]:
```
x = 0

if (x>0):
  print('Positive!')
elif(x<0):
  print('Negative!')
else:
  print('Zero!')
```

```
Zero!
```

In [188]:
```
fruits = ["apple", "banana", "cherry"]

for x in fruits:
   print(x)
```

```
apple
banana
cherry
```

In [193]:
```
for i in range(1, 5):
  for j in range(1, 5):
    print(f'{i}*{j}={i*j}')
```

```
1*1=1
1*2=2
1*3=3
1*4=4
2*1=2
2*2=4
2*3=6
2*4=8
3*1=3
3*2=6
3*3=9
3*4=12
4*1=4
4*2=8
4*3=12
4*4=16
```

In [189]:
```
fruits = ["apple", "banana", "cherry"]

for x in fruits:
  if x == 'banana':
    break
  print(x)
```

```
apple
```

In [186]:
```
x = 1
while (x<5):
  print(x)
  x +=1
```

```
1
2
3
4
```

Numpy

In [200]:
```
!pip install numpy
```

```
Requirement already satisfied: numpy in c:\users\asaad\anaconda3\lib\site-packa
ges (1.20.1)
```

In [202]:
```
import numpy as np
```

In [212]:
```
import numpy as np

# إنشاء مصفوفة من قائمة
arr = np.array([1, 2, 3, 4, 5])
print(arr)
```

```
[1 2 3 4 5]
```

In [2]:
```
import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]])
print(arr.shape)  # الناتج: (3 ,2)
print(arr.ndim)   # الناتج: 2
```

```
(2, 3)
2
```

In [214]:
```
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])

# الجمع
result = arr1 + arr2
print(result)  # [9 7 5] :الناتج

# الطرح
result = arr1 - arr2
print(result)  # [3- 3- 3-] :الناتج

# الضرب
result = arr1 * arr2
print(result)  # [18 10 4 ] :الناتج

# القسمة
result = arr1 / arr2
print(result)  # [ 0.5  0.4 0.25] :الناتج
```

```
[5 7 9]
[-3 -3 -3]
[ 4 10 18]
[0.25 0.4  0.5 ]
```

In [215]:
```
arr = np.array([1, 2, 3, 4, 5])

print(arr[0])    # 1 :الناتج
print(arr[1:4])  # [4 3 2] :الناتج
```

```
1
[2 3 4]
```

In [211]:
```
c = np.array([[1], [2], [3], [4], [5]])
print(c)

print('\nstatistics over whole array (all elements)')

print('c min', c.min())
print('c max', c.max())
print('c sum', c.sum())
print('c mean', c.mean())
```

```
[[1]
 [2]
 [3]
 [4]
 [5]]

statistics over whole array (all elements)
c min 1
c max 5
c sum 15
c mean 3.0
```

Pandas

In [ ]:
```
import pandas as pd
import numpy as np
```

Series

In [218]:
```
import pandas as pd

# إنشاء Series من قائمة
data = [1, 2, 3, 4, 5]
series = pd.Series(data)
print(series)
```

```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

In [219]:
```
# الوصول إلى القيم بواسطة الفهرس
print(series[2])  # الناتج: 3
```

```
3
```

In [220]:
```
# إنشاء Series مع فهرس مخصص
data = [1, 2, 3, 4, 5]
index = ['a', 'b', 'c', 'd', 'e']
series = pd.Series(data, index=index)
print(series)
```

```
a    1
b    2
c    3
d    4
e    5
dtype: int64
```

In [221]:
```
# تحويل Series إلى مصفوفة NumPy
array_data = series.to_numpy()
print(array_data)
```

```
[1 2 3 4 5]
```

In [223]:
```
s1 = pd.Series([10, 11, 12, 13, 14])
print( s1)

s2 = pd.Series([1, 2, 3], index=['first','second', 'third'])
print(s2)

s3 = pd.Series({'a':3.0, 'b':4, 'c':5})
print(s3)
```

```
0    10
1    11
2    12
3    13
4    14
dtype: int64
first     1
second    2
third     3
dtype: int64
a    3.0
b    4.0
c    5.0
dtype: float64
```

In [224]:
```
print(s3.size)
print(s3.shape)
print(s3.dtype)
print(s3.index)
```

```
3
(3,)
float64
Index(['a', 'b', 'c'], dtype='object')
```

DataFrames

In [225]:
```
import pandas as pd

# إنشاء DataFrame من قائمة
data = {'Name': ['John', 'Anna', 'Peter', 'Linda'],
        'Age': [28, 22, 32, 35],
        'City': ['New York', 'Paris', 'Berlin', 'London']}

df = pd.DataFrame(data)
print(df)
```

```
    Name  Age      City
0   John   28  New York
1   Anna   22     Paris
2  Peter   32    Berlin
3  Linda   35    London
```

In [ ]:
```
```

In [228]:
```
df = pd.DataFrame([ [1, 2, 3], [4, 5, 6]  ])
df
```

Out[228]:

	0	1	2
0	1	2	3
1	4	5	6

In [229]:
```
ar = np.array([ [1, 2, 3], [4, 5, 6], [5, 20, 1] ])
df = pd.DataFrame(ar, columns=['a', 'b', 'c'], index=['r1', 'r2', 'r3'])
df
```

Out[229]:

	a	b	c
r1	1	2	3
r2	4	5	6
r3	5	20	1

In [230]:
```
df['a']
```

Out[230]:
```
r1    1
r2    4
r3    5
Name: a, dtype: int32
```

In [231]:
```
df[['a', 'b']]
```

Out[231]:

	a	b
r1	1	2
r2	4	5
r3	5	20

In [232]:
```
df['d'] = df['c']+1
df
```

Out[232]:

	a	b	c	d
r1	1	2	3	4
r2	4	5	6	7
r3	5	20	1	2

In [233]:
```
df = df *2
df
```

Out[233]:

	a	b	c	d
r1	2	4	6	8
r2	8	10	12	14
r3	10	40	2	4

In [ ]:

In [235]:
```python
import pandas as pd

# قراءة ملف CSV
df = pd.read_csv('data.txt')

# عرض البيانات الأولى
print(df.head())
```

```
    Name  Age      City
0   John   28  New York
1   Anna   22     Paris
2  Peter   32    Berlin
3  Linda   35    London
```

In [236]:
```python
# الوصول إلى عمود معين
print(df['Name'])

# الوصول إلى صف معين
print(df.loc[1])

# الوصول إلى قيمة معينة
print(df.at[2, 'Age'])
```

```
0     John
1     Anna
2    Peter
3    Linda
Name: Name, dtype: object
Name     Anna
Age        22
City    Paris
Name: 1, dtype: object
32
```

In [238]:
```python
# فلترة الصفوف بناءً على شرط
print(df[df['Age'] > 30])
```

```
    Name  Age    City
2  Peter   32  Berlin
3  Linda   35  London
```

In [239]:
```python
# فلترة البيانات بناءً على قيمة معينة
print(df[df['City'] == 'New York'])
```

```
   Name  Age      City
0  John   28  New York
```

In [240]:
```
# تعديل قيمة
df.at[0, 'Age'] = 29

# إضافة صف جديد
new_row = {'Name': 'Mike', 'Age': 27, 'City': 'Sydney'}
df = df.append(new_row, ignore_index=True)
```

In [244]:
```
# حذف عمود
df = df.drop(columns=['City'])

# حذف صف
df = df.drop(2)
df
```

Out[244]:

	Name	Age
0	John	29
1	Anna	22
3	Linda	35
4	Mike	27

In [242]:
```
# المتوسط
print(df['Age'].mean())

# الحد الأدنى والحد الأقصى
print(df['Age'].min(), df['Age'].max())

# العدد الإجمالي للصفوف والأعمدة
print(df.shape)
```

```
28.25
22 35
(4, 2)
```

In [248]:
```
df = pd.read_csv('data.txt', header=0, index_col=0)
df
```

Out[248]:

	Age	City
Name		
John	28	New York
Anna	22	Paris
Peter	32	Berlin
Linda	35	London

In [245]:
```
df = pd.DataFrame([[1,2,3,4,5], [6,7,8,9,10]], index=['x','y'], columns=['a', 'b
df
```

Out[245]:

	a	b	c	d	e
x	1	2	3	4	5
y	6	7	8	9	10

In [246]:
```
df.mean()
df.mean(axis=0)
df.mean(axis=1)
```

Out[246]:
```
x    3.0
y    8.0
dtype: float64
```

In [247]:
```
df.describe()
```

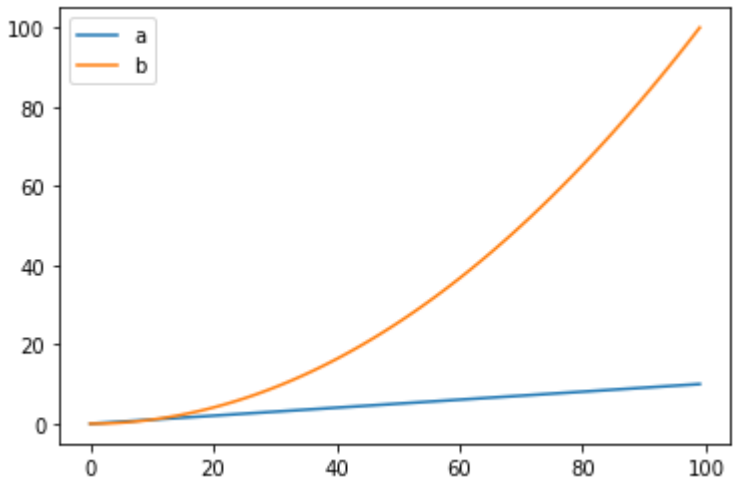
Out[247]:

	a	b	c	d	e
count	2.000000	2.000000	2.000000	2.000000	2.000000
mean	3.500000	4.500000	5.500000	6.500000	7.500000
std	3.535534	3.535534	3.535534	3.535534	3.535534
min	1.000000	2.000000	3.000000	4.000000	5.000000
25%	2.250000	3.250000	4.250000	5.250000	6.250000
50%	3.500000	4.500000	5.500000	6.500000	7.500000
75%	4.750000	5.750000	6.750000	7.750000	8.750000
max	6.000000	7.000000	8.000000	9.000000	10.000000

MatPlotLib

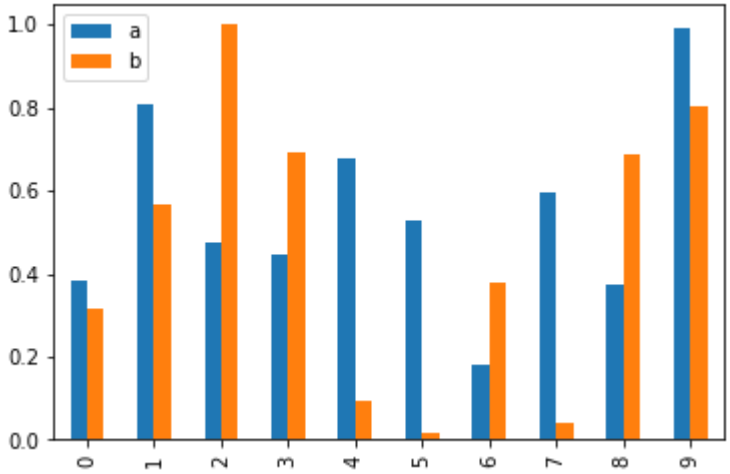
In [249]:
```
import matplotlib.pyplot as plt
x = np.linspace(0, 10, 100)
y = x**2

df = pd.DataFrame(np.array([x, y]).T, columns=['a', 'b'])
df.plot()
```

Out[249]:
```
<AxesSubplot:>
```



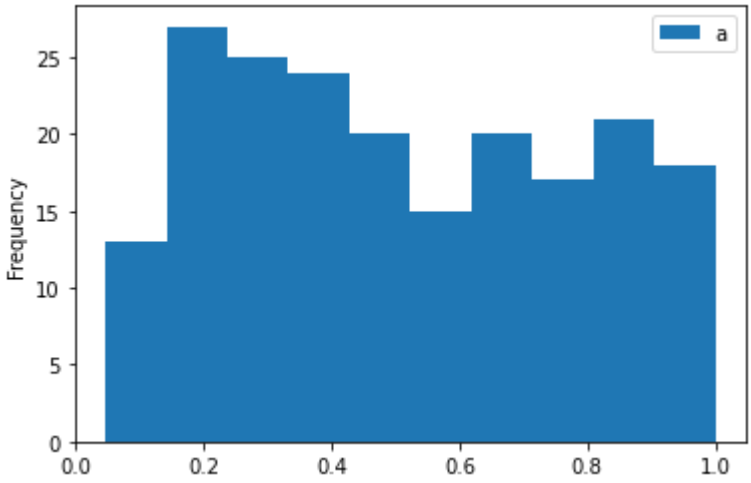
In [250]:
```
x = np.random.rand(10)
y = np.random.rand(10)

df = pd.DataFrame(np.array([x, y]).T, columns=['a', 'b'])
df.plot.bar()
```

Out[250]: <AxesSubplot:>



In [251]:
```
x = np.random.rand(200)

df = pd.DataFrame(x, columns=['a'])
df.plot.hist()
```

Out[251]: <AxesSubplot:ylabel='Frequency'>



In [252]:
```
data = {'First_Score':[100, 90, np.nan, 95],
        'Second_Score': [30, 45, 56, np.nan],
        'Third_Score':[np.nan, 40, 80, 98]}

df = pd.DataFrame(data)
df
```

Out[252]:

	First_Score	Second_Score	Third_Score
0	100.0	30.0	NaN
1	90.0	45.0	40.0
2	NaN	56.0	80.0
3	95.0	NaN	98.0

In [253]:
```
df.isnull()
```

Out[253]:

	First_Score	Second_Score	Third_Score
0	False	False	True
1	False	False	False
2	True	False	False
3	False	True	False

In [254]:
```
df.notnull()
```

Out[254]:

	First_Score	Second_Score	Third_Score
0	True	True	False
1	True	True	True
2	False	True	True
3	True	False	True

In [255]:
```
df.fillna(0, inplace=False)
```

Out[255]:

	First_Score	Second_Score	Third_Score
0	100.0	30.0	0.0
1	90.0	45.0	40.0
2	0.0	56.0	80.0
3	95.0	0.0	98.0

In [256]:
```
df.First_Score.fillna(int(df.First_Score.mean()), inplace=True)
```

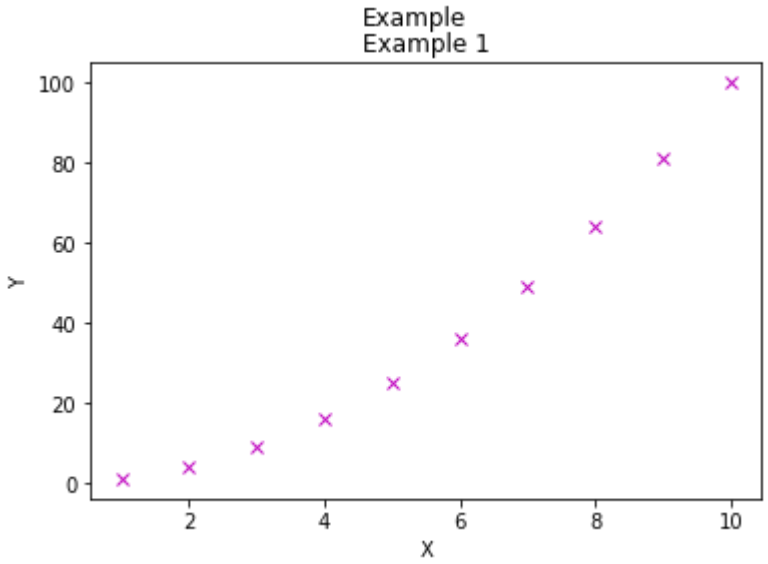
In [257]:
```
df.replace(np.nan, 98, inplace=True)
df
```

Out[257]:

	First_Score	Second_Score	Third_Score
0	100.0	30.0	98.0
1	90.0	45.0	40.0
2	95.0	56.0	80.0
3	95.0	98.0	98.0

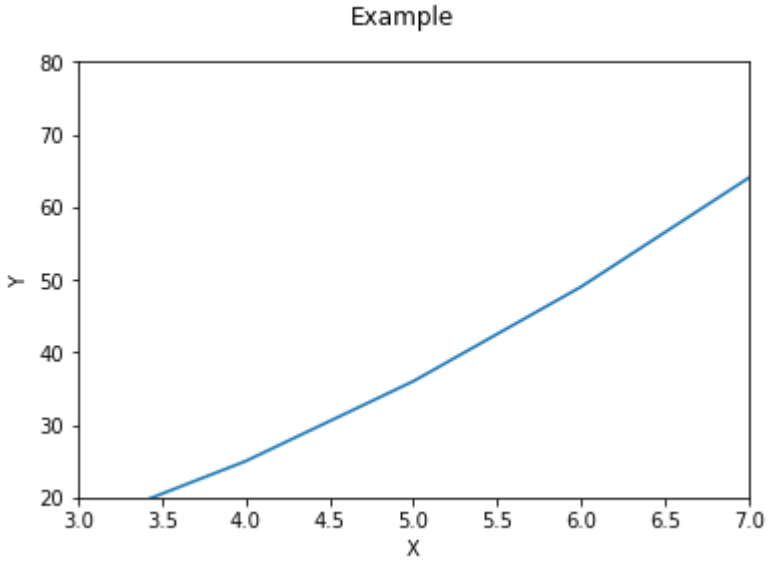
In [258]:

```
x = [1,2,3,4,5,6,7,8,9,10]
y = [1,4,9,16,25,36,49,64,81,100]
r = plt.plot(x, y, 'mx')
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Example 1')
plt.suptitle('Example')
plt.show()
```



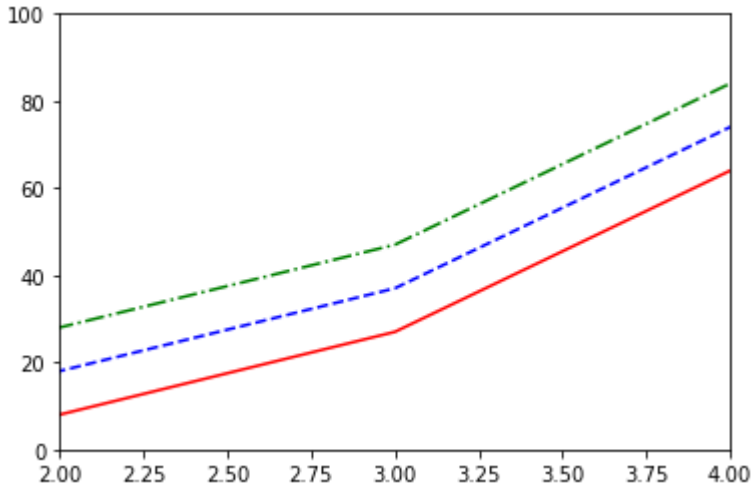
In [259]:

```
r = plt.plot([1,4,9,16,25,36,49,64,81,100])
plt.xlabel('X')
plt.ylabel('Y')
plt.suptitle('Example')
plt.axis([3, 7, 20, 80])
plt.show()
```

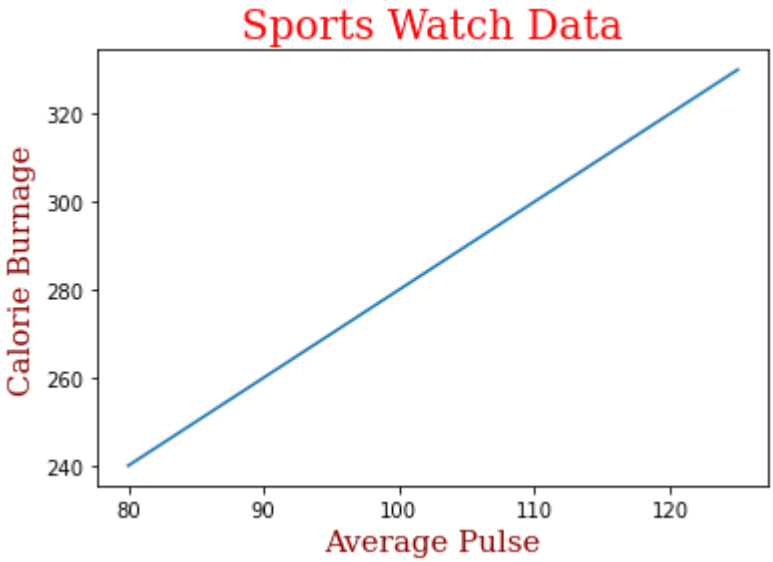


In [260]:
```
x = [1,2,3,4,5,6,7,8,9,10]
y2 = [1,8,27,64,125,216,343,512,729,1000]
y3 = [10,18,37,74,135,226,353,522,739,1010]
y4 = [20,28,47,84,145,236,363,532,749,1020]
plt.plot(x, y2,'r-', x, y3, 'b--', x, y4, 'g-.')
plt.xlim((2, 4))
plt.ylim((0, 100))
```

Out[260]: (0.0, 100.0)

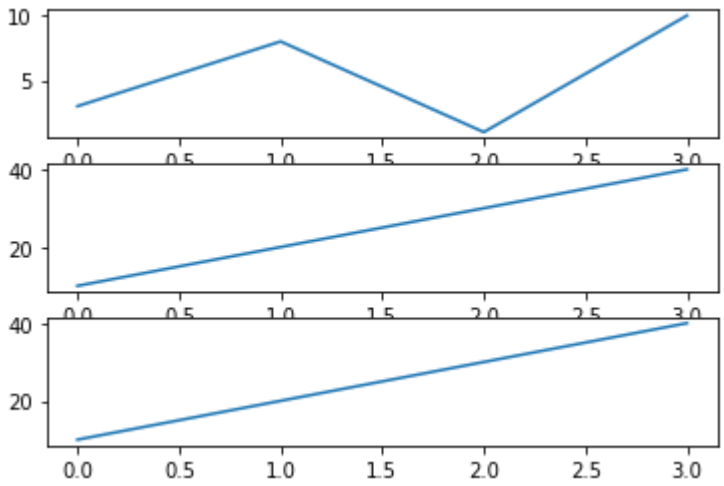


In [262]:
```
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

f1 = {'family':'serif', 'color':'red', 'size':20}
f2 = {'family':'serif', 'color':'darkred', 'size':15}

plt.title("Sports Watch Data", fontdict=f1)
plt.xlabel("Average Pulse", fontdict=f2)
plt.ylabel("Calorie Burnage", fontdict=f2)

plt.plot(x, y)
plt.show()
```



In [263]:

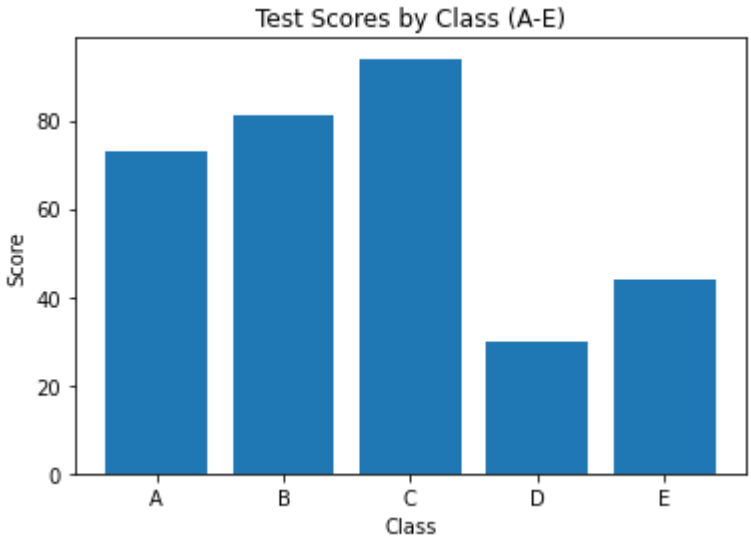
```
#plot 1:
x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])

plt.subplot(3, 1, 1)
plt.plot(x,y)# plot the data


#plot 2:
x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])

plt.subplot(3, 1, 2)
plt.plot(x,y)


#plot 3:
x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])

plt.subplot(3, 1, 3)
plt.plot(x,y)

plt.show()
```



In [264]:

```
groups = [0, 1, 2, 3, 4]
groups_titles = ['A', 'B', 'C', 'D', 'E']

plt.bar(groups, [73, 81, 94, 30, 44])
plt.xticks(groups, groups_titles)
plt.xlabel('Class')
plt.ylabel('Score')
plt.title('Test Scores by Class (A-E)')
```

Out[264]: Text(0.5, 1.0, 'Test Scores by Class (A-E)')



In [266]:

```
g1 = (77, 58, 84, 62)
g2 = (99, 92, 88, 80)
g3 = (85, 81, 79, 80)

plt.subplots()
index = np.arange(4)
bar_width = 0.25

plt.bar(index, g1, bar_width, color='r', label='Group 1')
plt.bar(index+bar_width, g2, bar_width, color='b', label='Group 2')
plt.bar(index+ 2*bar_width, g3, bar_width, color='g', label='Group 3')

plt.xlabel('Score')
plt.ylabel('Class')
plt.title('Test Scores by Class (A-E)')
plt.xticks(index+bar_width, ['A', 'B', 'C', 'D'])
plt.legend()
```

Out[266]: <matplotlib.legend.Legend at 0x227dbe2b9d0>