LAB: Dữ liệu với NumPy, Pandas và Matplotlib

Mục tiêu

- Làm quen với NumPy: Tạo mảng, thao tác cơ bản và nâng cao với mảng.
- Thực hành xử lý dữ liệu với Pandas: Đọc, phân tích và làm sạch dữ liệu.
- Trực quan hóa dữ liệu bằng Matplotlib: Biểu đồ cơ bản và nâng cao.

Phần 1: NumPy cơ bản

Bài tập 1: Tạo mảng và thao tác cơ bản

1 Tạo một mảng NumPy với các giá trị từ 1 đến 20.

2 Tìm tổng, giá trị lớn nhất, nhỏ nhất và trung bình của mảng.

3 Tạo một mảng 2D (3x5) chứa các số ngẫu nhiên từ 0 đến 100.

4 Lấy hàng thứ 2 và cột thứ 3 của mảng 2D.

In [1]:
```
import numpy as np

# 1. Tạo một mảng NumPy với các giá trị từ 1 đến 20
array_1 = np.array([1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20])
print("Mảng từ 1 đến 20:", array_1)

# 2. Tìm tổng, giá trị lớn nhất, nhỏ nhất và trung bình của mảng
print("Tổng:", sum(array_1))
print("Giá trị lớn nhất:", array_1.max())
print("Giá trị nhỏ nhất:", array_1.min())
print("Trung bình:", array_1.mean())

# 3. Tạo một mảng 2D (3x5) chứa các số ngẫu nhiên từ 0 đến 100
array_2d = np.random.randint(0,101, size=(3,5))
print("Mảng 2D:", array_2d)

# 4. Lấy hàng thứ 2 và cột thứ 3 của mảng 2D
print("Hàng thứ 2:", array_2d[1, :])
print("Cột thứ 3:", array_2d[:, 2])
```

```
Mảng từ 1 đến 20: [ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20]
Tổng: 210
Giá trị lớn nhất: 20
Giá trị nhỏ nhất: 1
Trung bình: 10.5
Mảng 2D: [[80  9 95 13 27]
 [71 10 67 26 75]
 [48 67 69 18 25]]
Hàng thứ 2: [71 10 67 26 75]
Cột thứ 3: [95 67 69]
```

Bài tập 2: Các thao tác nâng cao

1 Tạo một mảng NumPy chứa 20 giá trị ngẫu nhiên từ 0 đến 1.

2 Chuẩn hóa mảng này (đưa các giá trị về khoảng [0, 1]).

3 Tính tích vô hướng (dot product) của hai mảng 1D: [1, 2, 3] và [4, 5,6].

4 Tạo một ma trận 5x5 và tính định thức (determinant) và nghịch đảo của ma trận.

In [2]:
```python
# 1. Tạo một mảng NumPy chứa 20 giá trị ngẫu nhiên từ 0 đến 1
random_array = np.random.rand(20)
print("Mảng ngẫu nhiên từ 0 đến 1:", random_array)

# 2. Chuẩn hóa mảng này (đưa các giá trị về khoảng [0, 1])
array_min = np.min(random_array)
array_max = np.max(random_array)

normalized_array = (random_array-array_min)/ (array_max-array_min)
print("Mảng sau khi chuẩn hóa:", normalized_array)

# 3. Tính tích vô hướng (dot product) của hai mảng 1D
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
dot_product = np.dot(a,b)
print("Tích vô hướng của a và b:", dot_product)

# 4. Tạo một ma trận 5x5 và tính định thức, nghịch đảo
matrix = np.random.randint(1,11, size=(5,5))
print("Ma trận:", matrix)
determinant = np.linalg.det(matrix)
print("Định thức của ma trận:", determinant)
if determinant != 0:
    inverse_matrix = np.linalg.inv(matrix)
    print("Ma trận nghịch đảo:", inverse_matrix)
else:
    print("Ma trận không khả nghịch (định thức = 0).")
```

```
Mảng ngẫu nhiên từ 0 đến 1: [0.17132912 0.02543667 0.58463625 0.36583876 0.27766746
0.97178887
 0.48929877 0.48402362 0.38199992 0.82775548 0.0490899  0.29638597
 0.89514565 0.73699314 0.09798389 0.08624273 0.86163321 0.73622566
 0.7080599  0.89271138]
Mảng sau khi chuẩn hóa: [0.15416296 0.         0.59090007 0.35969916 0.26652951 1.
 0.490158   0.48458381 0.37677648 0.84780149 0.02499411 0.28630916
 0.91901194 0.75189393 0.07665986 0.0642531  0.88359971 0.75108294
 0.72132049 0.91643968]
Tích vô hướng của a và b: 32
Ma trận: [[ 9  9 10  3  6]
 [10  8  4  7  4]
 [ 2  9  5  5 10]
 [10  1  9  5  3]
 [10  6  3  9 10]]
Định thức của ma trận: 19158.99999999999
Ma trận nghịch đảo: [[ 0.14870296 -0.05600501 -0.18586565 -0.08528629  0.14463177]
 [ 0.03799781  0.11555927  0.02656715 -0.08497312 -0.0700976 ]
 [-0.04238217  0.0029751   0.10773005  0.13873375 -0.12511091]
 [-0.29009865  0.20016702  0.19552169  0.17620961 -0.15439219]
 [ 0.10230179 -0.1943734  -0.03836317 -0.06393862  0.17391304]]
```

Phần 2: Pandas cơ bản

Bài tập 3: Làm quen với DataFrame

1 Tạo một DataFrame chứa thông tin sau:

Name	Age	Score
Alice	23	85
Bob	25	90
Charlie	22	78
David	24	92
Eva	21	88

2 Tính giá trị trung bình của cột "Score".

3 Lọc các hàng có "Score" lớn hơn 85.

In [3]:
```python
import pandas as pd

# 1. Tạo DataFrame
data = {
    "Name" : ["Alice", "Bob", "Charlie", "David", "Eva"],
    "Age" : [23,25,22,24,21],
    "Score" : [85,90,78,92,88]
}
df = pd.DataFrame(data)
print("DataFrame:\n", df)
```

```
# 2. Tính giá trị trung bình của cột "Score"
print(df["Score"].mean())

# 3. Lọc các hàng có "Score" lớn hơn 85
filtered_df = df[df["Score"]>85]
print("Các hàng có Score > 85:\n", filtered_df)
```

```
DataFrame:
       Name  Age  Score
0    Alice   23     85
1      Bob   25     90
2  Charlie   22     78
3    David   24     92
4      Eva   21     88
86.6
Các hàng có Score > 85:
     Name  Age  Score
1    Bob   25     90
3  David   24     92
4    Eva   21     88
```

Bài tập 4: Đọc và phân tích dữ liệu từ file

1 Tải file Iris.csv từ Kaggle Iris Dataset.

2 Đọc dữ liệu từ file CSV vào DataFrame.

3 Hiển thị thông tin cơ bản (tổng quan, kiểu dữ liệu, số lượng null).

4 Tính trung bình, lớn nhất, nhỏ nhất của cột sepal_length.

In [4]:
```
import pandas as pd
# Đọc file CSV
iris_df = pd.read_csv('archive\Iris.csv')
print("Thông tin tổng quan về dữ liệu:", iris_df.info())
print("Mô tả dữ liệu:", iris_df.describe())

# Tính toán cơ bản
print("Trung bình sepal_length:", iris_df["SepalLengthCm"].mean())
print("Giá trị lớn nhất sepal_length:", iris_df["SepalLengthCm"].max())
print("Giá trị nhỏ nhất sepal_length:", iris_df["SepalLengthCm"].min())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
Thông tin tổng quan về dữ liệu: None
Mô tả dữ liệu:                Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalW
idthCm
count  150.000000     150.000000    150.000000     150.000000    150.000000
mean    75.500000       5.843333      3.054000       3.758667      1.198667
std     43.445368       0.828066      0.433594       1.764420      0.763161
min      1.000000       4.300000      2.000000       1.000000      0.100000
25%     38.250000       5.100000      2.800000       1.600000      0.300000
50%     75.500000       5.800000      3.000000       4.350000      1.300000
75%    112.750000       6.400000      3.300000       5.100000      1.800000
max    150.000000       7.900000      4.400000       6.900000      2.500000
Trung bình sepal_length: 5.843333333333334
Giá trị lớn nhất sepal_length: 7.9
Giá trị nhỏ nhất sepal_length: 4.3
```

Phần 3: Làm sạch dữ liệu

Bài tập 5: Xử lý dữ liệu thiếu

1 Tạo một DataFrame chứa các giá trị sau:

Name	Age	City	Salary
Alice	23	New York	60000
Bob	NaN	Boston	52000
Charlie	25	NaN	NaN
David	24	Chicago	58000
Eva	22	Boston	NaN

2 Điền giá trị thiếu trong cột Age bằng giá trị trung bình.

3 Xóa các hàng có nhiều hơn 1 giá trị thiếu.

4 Điền giá trị thiếu trong cột Salary bằng 50000.

In [13]:
```python
# Tạo DataFrame chứa dữ liệu thiếu
data_with_missing = {
    "Name": ["Alice", "Bob", "Charlie", "David", "Eva"],
```

```
    "Age": [23, "NaN", 25, 24, 22],
    "City": ["New York", "Boston", "NaN", "Chicago", "Boston"],
    "Salary": [60000, 52000, "NaN", 58000, "NaN"]
}

df_missing = pd.DataFrame(data_with_missing)
print("Dữ liệu ban đầu:\n", df_missing)

# Cách 1: Chuyển sang object trước khi thay thế
df_missing = df_missing.astype(object).replace("NaN", np.nan)

# Chuyển Age sang dạng số
df_missing["Age"] = pd.to_numeric(df_missing["Age"], errors="coerce")

# Điền giá trị thiếu trong cột Age bằng trung bình
df_missing["Age"] = df_missing["Age"].fillna(df_missing["Age"].mean())

# Xóa các hàng có nhiều hơn 1 giá trị thiếu
df_cleaned = df_missing.dropna(thresh=2)

# Điền giá trị thiếu trong cột Salary bằng 50000
df_cleaned["Salary"] = df_cleaned["Salary"].fillna(50000)

print("\nDữ liệu sau khi xử lý:\n", df_cleaned)
```

```
Dữ liệu ban đầu:
       Name  Age      City Salary
0    Alice   23  New York  60000
1      Bob  NaN    Boston  52000
2  Charlie   25       NaN    NaN
3    David   24   Chicago  58000
4      Eva   22    Boston    NaN

Dữ liệu sau khi xử lý:
       Name   Age      City   Salary
0    Alice  23.0  New York  60000.0
1      Bob  23.5    Boston  52000.0
2  Charlie  25.0       NaN  50000.0
3    David  24.0   Chicago  58000.0
4      Eva  22.0    Boston  50000.0
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_11476\266581366.py:13: FutureWarning: Dow
ncasting behavior in `replace` is deprecated and will be removed in a future versio
n. To retain the old behavior, explicitly call `result.infer_objects(copy=False)`. T
o opt-in to the future behavior, set `pd.set_option('future.no_silent_downcasting',
True)`
  df_missing = df_missing.astype(object).replace("NaN", np.nan)
```

Phần 4: Trực quan hóa dữ liệu với Matplotlib

Bài tập 6: Biểu đồ cơ bản

1 Tạo một biểu đồ đường biểu diễn hàm số y = x ** 2 trên khoảng [-10,10]

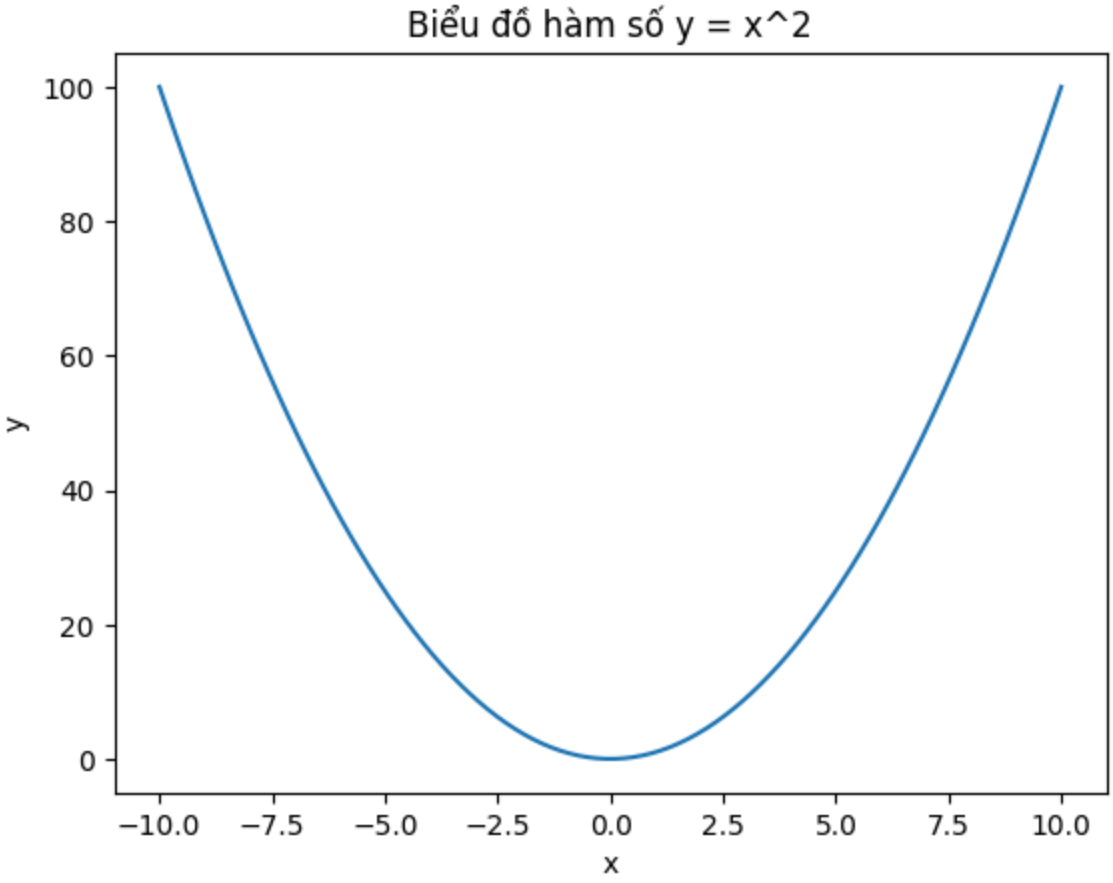
2 Vẽ biểu đồ cột thể hiện điểm số (Score) của các sinh viên từ Bài tập 3.

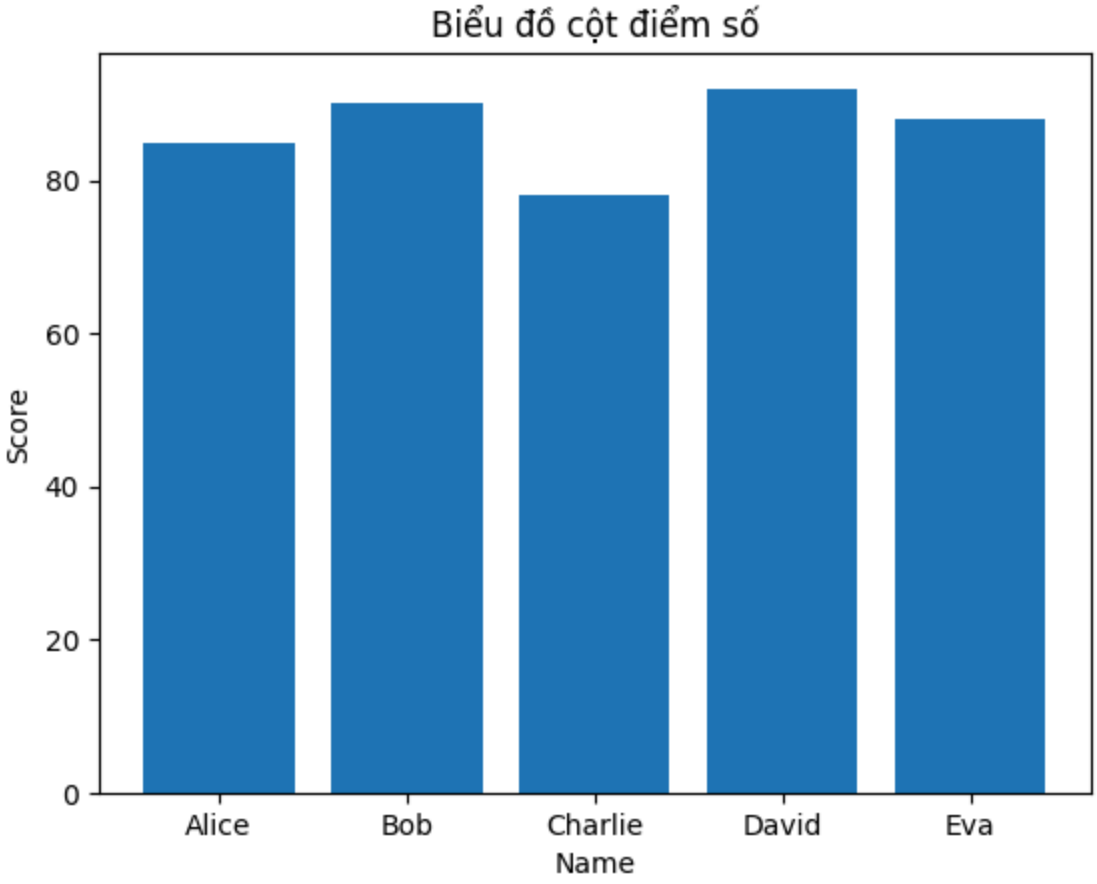
3 Tạo một biểu đồ tròn (pie chart) thể hiện phần trăm mỗi loại hoa trong tập dữ liệu Iris.

In [15]:
```
import matplotlib.pyplot as plt

# 1. Biểu đồ đường hàm số y = x^2
x = np.linspace(-10, 10, 100)
y = x**2
plt.plot(x, y)
plt.xlabel("x")
plt.ylabel("y")
plt.title("Biểu đồ hàm số y = x^2")
plt.show()


# 2. Biểu đồ cột điểm số
data = {
    "Name" : ["Alice", "Bob", "Charlie", "David", "Eva"],
    "Score" : [85,90,78,92,88]
}
df = pd.DataFrame(data)
plt.bar(df["Name"], df["Score"])
plt.xlabel("Name")
plt.ylabel("Score")
plt.title("Biểu đồ cột điểm số")
plt.show()
```



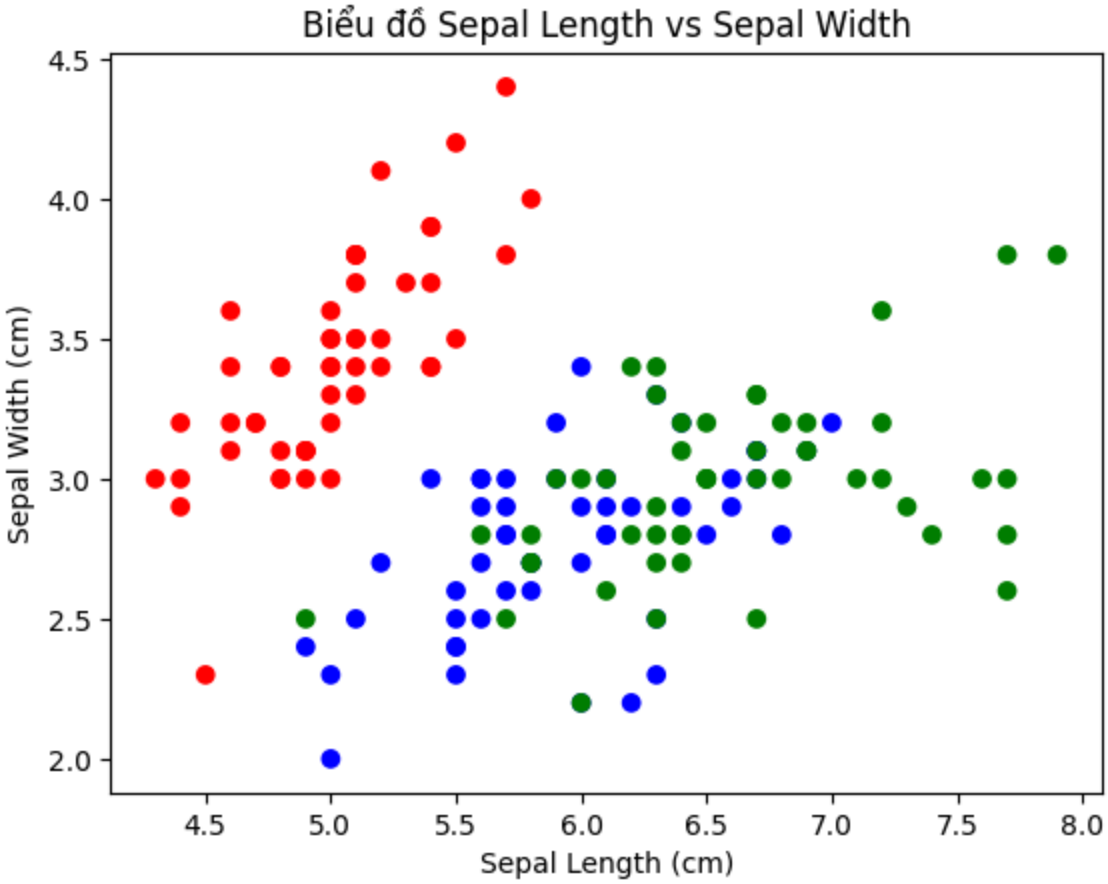


Bài tập 7: Biểu đồ nâng cao

1 Vẽ biểu đồ phân tán (scatter plot) giữa sepal_length và sepal_width của tập dữ liệu Iris.
Dùng màu sắc để phân biệt các loại hoa (species).
2 Thêm tiêu đề, nhãn trục và chú thích cho biểu đồ.

In [17]:
```
# 1. Biểu đồ phân tán với màu sắc theo loại hoa
# Tạo một danh sách ánh xạ màu
colors = {"Iris-setosa": "red", "Iris-versicolor": "blue", "Iris-virginica": "green
iris_df["Color"] = iris_df["Species"].map(colors)

# Vẽ scatter plot
plt.scatter(iris_df["SepalLengthCm"], iris_df["SepalWidthCm"], c=iris_df["Color"])
plt.xlabel("Sepal Length (cm)")
plt.ylabel("Sepal Width (cm)")
plt.title("Biểu đồ Sepal Length vs Sepal Width")
plt.show()
```


Biểu đồ Sepal Length vs Sepal Width
Sepal Width (cm)
Sepal Length (cm)
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0