Дәріс 12

Тақырыбы: Екі өлшемді массив

Оқытудың әдістемесі мен формасы: Баяндау, дәріс

Сабақ мақсаты: Екіөлшемді массивтермен жұмыс істеудің негізгі принциптерімен танысу.

Негізгі түсініктер: массив, индекс, екіөлшемді массив

Әдебиеттер: [6,7]

Массивтің элементтеріне арифметикалық амалдар, оны енгізу және шығару жеке элементтері бойынша орындалады.

Мысалы, массивпен жұмыс істеу кезінде көбінесе қолданылатын әрекеттер мынадай:

- массивті енгізу;
- массивті шығару;
- массивтен элемент іздеу;
- массивке амалдар қолдану.

Екі өлшемді массивті сипаттау үшін әр өлшемін квадрат жақшада жазамыз. Мысалы: int matr [6][8]; - бұл массив 7 жол және 9 бағанадан тұрады. Екі өлшемді массивке қатынау үшін оның индексіне нұсқаймыз, мысалы: matr [i][j].

Мысалы, кесте түрінде:

```
1   2   3
4   5   6
7   8   9
```

Бұл 3 жол × 3 бағаннан тұратын екі өлшемді массив.

2. Python тілінде екі өлшемді массив құру тәсілдері

2.1. Тізімдердің тізімі (List of Lists)

Python тілінде арнайы “массив” типі жоқ, сондықтан көбінесе **тізім (list)** қолданылады.

```
# Екі өлшемді массив құру
matrix = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]

print(matrix)
```

Элементке жүгіну:

```
print(matrix[0][0])   # 1
print(matrix[2][1])   # 8
```

2.2. Екі өлшемді массивті цикл арқылы толтыру

```
rows = 3
cols = 4

matrix = [[0 for j in range(cols)] for i in range(rows)]
print(matrix)
```

Мысалы, мәндерді енгізу:

```
for i in range(rows):
    for j in range(cols):
        matrix[i][j] = i + j
```

3. Екі өлшемді массив элементтерін шығару

3.1. Бір жолда (әр элементті шығару)

```
for i in range(len(matrix)):
    for j in range(len(matrix[i])):
        print(matrix[i][j], end=' ')
    print()  # келесі жолға өту
```

Шығару нәтижесі:

```
0 1 2 3
1 2 3 4
2 3 4 5
```

4. NumPy кітапханасы арқылы екі өлшемді массив

Python-да үлкен және жылдам есептеулер үшін **NumPy** модулі жиі қолданылады.

4.1. NumPy арқылы 2D массив құру

```
import numpy as np

a = np.array([[1, 2, 3],
              [4, 5, 6],
              [7, 8, 9]])

print(a)
```

4.2. Негізгі қасиеттері

```
print(a.shape)    # (3, 3)
print(a.ndim)     # өлшем саны: 2
print(a.size)     # элементтер саны: 9
```

4.3. Элементтерге қол жеткізу

```
print(a[1][2])   # 6
print(a[0, 1])   # 2
```

4.4. Мысал: кездейсоқ 2D массив

```
import numpy as np

matrix = np.random.randint(0, 10, size=(3, 4))
print(matrix)
```

5. Практикалық мысалдар

Мысал 1: Массив элементтерінің қосындысы

```
total = 0
for i in range(len(matrix)):
    for j in range(len(matrix[i])):
        total += matrix[i][j]
print("Қосындысы:", total)
```

Мысал 2: Негізгі диагональ элементтерін шығару

```
for i in range(len(matrix)):
    print(matrix[i][i], end=' ')
```

Мысал 3: Максималды элементті табу

```
max_val = matrix[0][0]
for i in range(len(matrix)):
    for j in range(len(matrix[i])):
        if matrix[i][j] > max_val:
            max_val = matrix[i][j]
print("Ең үлкен элемент:", max_val)
```

6. Қорытынды

Артықшылықтары	Қолданылуы
Кестелік деректерді ыңғайлы сақтайды	Матрицалар, суреттер, кестелер
Элементтерге индекспен жылдам қол жеткізу	Есептеу, статистика, деректер талдау
NumPy арқылы жоғары өнімділік	Ғылыми есептер, машиналық оқыту