NumPy 치트시트

NumPy란?

NumPy(Numerical Python)는 Python에서 과학 계산을 위한 핵심 라이브러리로, 다차원 배열 객체와 이를 처리하기 위한 다양한 함수를 제공합니다.

```
import numpy as np
```

배열 생성 (Array Creation)

- 리스트로부터 생성: `np.array([1, 2, 3])`
- 특정 값으로 채운 배열 생성:
  - `np.zeros((2, 3))`: 2x3 크기의 0으로 채워진 배열
  - `np.ones((2, 3))`: 1로 채워진 배열
  - `np.full((2, 3), 7)`: 7로 채워진 배열
- 연속된 값으로 생성:
  - `np.arange(10, 30, 5)`: 10부터 30 전까지 5씩 증가하는 배열
  - `np.linspace(0, 2, 9)`: 0과 2 사이를 9개의 균일한 간격으로 나눈 배열
- 랜덤 배열:
  - `np.random.rand(2, 2)`: 0과 1 사이의 균등 분포
  - `np.random.randn(2, 2)`: 표준 정규 분포

배열 속성 및 형태 변경

- `arr.shape`: 배열의 차원 (튜플)
- `arr.ndim`: 차원의 수
- `arr.size`: 전체 요소의 수
- `arr.dtype`: 요소의 데이터 타입
- `arr.reshape(new_shape)`: 배열의 형태를 변경
- `arr.T`: 배열을 전치(Transpose)

인덱싱 및 슬라이싱

- 기본 인덱싱: `arr[2]`, `arr[1, 2]`
- 슬라이싱: `arr[0:5]`, `arr[:, 1]` (모든 행의 1번 열)
- 불리언 인덱싱 (Boolean Indexing):

```
arr = np.array([1, 2, 3, 4])
arr[arr > 2]  # array([3, 4])
```

- 팬시 인덱싱 (Fancy Indexing): `arr[[0, 1, 3]]` (0, 1, 3번 인덱스의 요소 선택)

배열 연산

- 요소별 연산 (Element-wise Operations):
  - `arr + 5`, `arr * 2`
  - `arr1 + arr2`, `arr1 * arr2`
- 브로드캐스팅 (Broadcasting): 크기가 다른 배열 간의 연산을 가능하게 함.
- 행렬 곱:
  - `A.dot(B)`
  - `A @ B` (Python 3.5+)

주요 함수 (Ufuncs)

- 집계 함수:
  - `arr.sum()`, `arr.min()`, `arr.max()`, `arr.mean()`, `arr.std()`
  - 축 지정: `arr.sum(axis=0)` (열 기준 합계)
- 수학 함수:
  - `np.sqrt(arr)`: 제곱근
  - `np.exp(arr)`: 지수 함수
  - `np.log(arr)`: 로그 함수
  - `np.sin(arr)`, `np.cos(arr)`: 삼각 함수
- 비교 함수: `np.maximum(arr1, arr2)`: 요소별 최댓값

배열 결합 및 분할

- `np.concatenate([arr1, arr2], axis=0)`: 축을 따라 배열 결합
- `np.vstack((arr1, arr2))`: 수직으로 쌓기 (행 추가)
- `np.hstack((arr1, arr2))`: 수평으로 쌓기 (열 추가)
- `np.split(arr, 3)`: 배열을 3개로 분할
- `np.hsplit(arr, 3)`, `np.vsplit(arr, 2)`: 수평/수직 분할

파일 입출력

- 텍스트 파일 저장/로드:
  - `np.savetxt('my_array.txt', arr, delimiter=',')`
  - `np.loadtxt('my_array.txt', delimiter=',')`
- 바이너리 파일 저장/로드:
  - `np.save('my_array.npy', arr)`
  - `np.load('my_array.npy')`