**1. 어레이 array**

동일한 데이터타입의 변수들이 연속적으로 메모리에 할당된 자료구조.

여러개의 값을 저장하기 위한 구조.

특징)

- 인덱스를 통해 해당 데이터 배열요소(element)에 접근할 수 있다.

- 고정된 크기를 가지며 변경할 수 없다.

- 동일한 데이터타입 요소를 가질 수 있다.

- 메모리의 연속된 위치에 즉, 순서대로 저장된다.

- 인덱스를 이용한 요소접근이 빠르다.

- 임의의 위치에 데이터를 추가,삭제시, 많은 시간과 비용이 든다.

: 3개의 칸에서 2번째에 “ㄱ”을 추가하려면 끝에 공간을 하나 더 만들고(4칸) 2번째와 3번째칸을 오른쪽으로 밀고 2번째에 집어넣어야하며, 삭제할때도 그 위치의 요소를 삭제 후, 한칸씩 왼쪽으로 밀고 빈공간을 없애야한다.

**2. hash (hash table)**

hash란, 데이터를 key와 value의 한쌍으로 이루어, 검색과 저장이 아주 빠르게 진행되게 하기 위한, 데이터를 다루는 기법 중 하나이다.

- key는 데이터를 검색하는데 사용되고, 이는 배열의 인덱스로 변환되기 때문에 검색과 저장의 평균시간복잡도는 O(1)에 수렴한다.

- 프로그램 언어의 연관배열(associative array)등에 많이 사용.

**해쉬 접근방식)**

**-** 키를 가지고 해쉬함수를 통해 해쉬코드를 계산한다.

- 해쉬값을 모듈러 연산을 통해 나온값을 인덱스로 지정하여 배열공간의 해당인덱스에 저장한다.

: 모듈러 연산 = 두 정수를 나눌 때 나머지를 이용하여 정의하는 방법.

: **여기서는 key -> 해쉬함수 -> 해쉬코드를 배열공간 수 만큼 나누어 나머지를 인덱스로 사용하여 넣음.**

- 입력key가 들어오면 인덱스를 계산하고 해당위치로 찾아가서 값이 동일하면 value를 반환한다.

- 만약, 다수의 데이터가 같은 인덱스가 나온다면, 리스트 형태로 옆으로 나열하여 저장.



**- 만약 Dan의 성별을 알고싶으면 key인 Dan의 해시값을 구하여 배열의 상자 수인 5로 모듈러 연산을 하고, 결과는 4가 나온다. 배열의 4번인덱스에 저장된 데이터의 키가 Dan과 일치하면 value값(M)을 출력한다.**

**- 만약, Ally가 입력key이면 3번 인덱스로 가서 Joe를 선두로하는 리스트를 선형탐색 한다.**

**: 이렇게 해시값이 충돌(인덱스가 같을 때)할 때에는 리스트를 이용하고 있어, 저장할 데이터 수가 정해져있지 않더라도 유연하게 대응가능하다.**

**: 이렇게 리스트를 사용하는 방법을 “연쇄법”이라한다.**

**- 해시 테이블에 사용하는 배열의 크기가 너무 작으면 충돌이 많아지고, 선형탐색 빈도가 높아진다. 반대로 배열 크기가 너무크면 데이터가 없는 상자가 많아져, 메모리를 낭비하게된다.**

3. 어레이와 리스트 차이

**1) 선언조건**

- 리스트는 숫자형, 문자열 등 모든 자료를 타입을 보존하여 가질 수 있습니다.

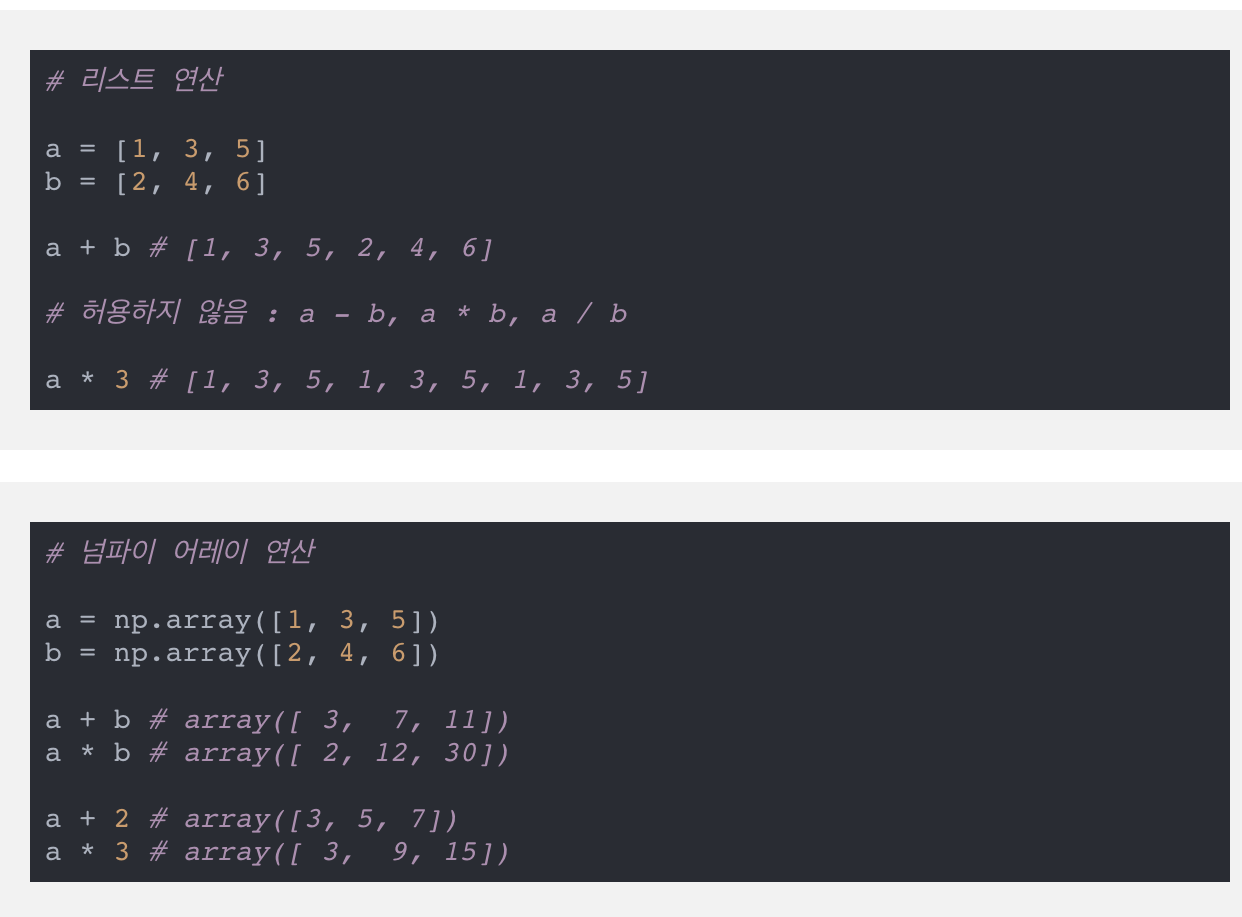
(여러 가지 자료형 허용)

반면, 넘파이 어레이는 숫자형과 문자열이 섞이면 모두 문자열로 전환됩니다.

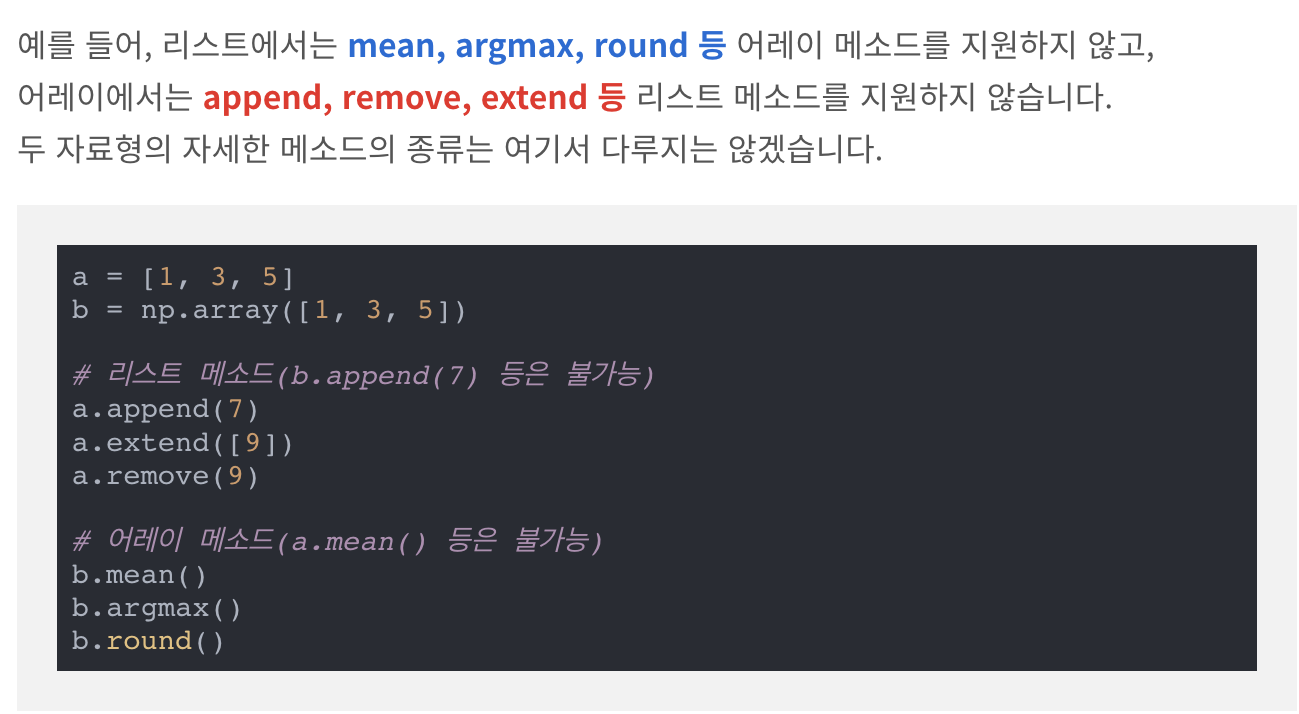
(한 가지 자료형만 허용)

**2) 연산**

리스트 : 리스트간의 연산 또한 허용하지 않고, 원소 복사를 의미



**3. 메소드 종류**



**4. 연산속도**

