리스트를 이용한 배열 연산

2019년 2학기

리스트를 이용한 2차원 배열 연산

1. 문제

아래와 같이 두개의 행렬을 리스트를 이용해서 각각 만들고,

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \quad , \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

(1) 리스트를 출력하시오.

(2) numpy 로 형변환 해서 출력하시오.

(3) 두개의 행렬곱을 수행하시오.

실행결과

```
[[1, 2, 3], [4, 5, 6]]
[[7], [8], [9]]
------------------------------
[[1 2 3]
 [4 5 6]]
[[7]
 [8]
 [9]]
------------------------------
[[ 50]
 [122]]
```

2. 문제

다음과 같은 조건을 만족하는 isNumber() 함수를 만드시오

실행결과

```
print(isNumber(3)) #True
print(isNumber('3')) #False
print(isNumber([1,2,3])) #Ture
print(isNumber([1,2,'3'])) #False
print(isNumber((1,2,3))) #Ture
print(isNumber((1,2,'3'))) #False
```


3. 문제

다음과 같은 조건을 만족하는 ltall() 함수를 리스트 shift 를 이용해서 만드시오

실행결과

```
print(ltall([1,1,1,1])) #True
print(ltall([1,1,1,2])) #False
print(ltall([1,1,1])) #True
print(ltall([1,1,2])) #False
```

4. 문제

다음과 같은 조건을 만족하는 ltall() 함수를 list reverse 를이용해서 만드시오

실행결과

```
print(ltall([1,1,1,1,1])) #True
print(ltall([1,1,2,1,1])) #False
print(ltall([1,1,1])) #True
print(ltall([1,1,2])) #False
```


5. 문제

다음과 같이 행렬의 크기를 출력해주는 lt_shape() 함수를 만드시오.

추가조건

행렬의 크기가 잘못된 경우 'Matrix shape Error ' 를 출력하도록 하시오.

실행결과

```
ltA = [[1,2,3],[4,5,6],[1,2,3],[4,5,6]]
print(lt_shape(ltA)) #(4,3)
ltA = [[1,2,3],[4,5,6],[1,2,3],[5,6]]
print(lt_shape(ltA)) #'Matrix shape Error '
```

6. 문제

아래와 같이 행렬을 입력받고, 행과 열을 치환하는 lt_Trans() 함수를 만드시오.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

조건

- 2차원 행렬의 크기 정보를 구하는 함수 lt_shape() 를 만들어서 이용하시오.
- 새로운 행렬을 리스트 append() 함수를 이용해서 만드시오.

실행결과

```
lt1 = [[1,2,3],[4,5,6]]
print(lt1) #[[1, 2, 3], [4, 5, 6]]
lt2 = lt_Trans(lt1)
print(lt2) #[[1, 4], [2, 5], [3, 6]]
```


◆ HW

아래와 같이 행렬을 입력받고, 행과 열을 치환하는 lt_Trans2() 함수를 만드시오.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

조건

- 새로운 행렬을 LC 을 이용해서 만드시오.

실행결과

```
lt1 = [[1,2,3],[4,5,6]]
print(lt1) #[[1, 2, 3], [4, 5, 6]]
lt2 = lt_Trans(lt1)
print(lt2) #[[1, 4], [2, 5], [3, 6]]
```

7. 문제

아래와 같이 1차원 행렬의 엘리멘터리 곱을 수행하는 lt_mul() 함수를 만드시오.

실행결과

```
lt1 = [1,2,3]
lt2 = [4,5,6]
print(lt_mul(lt1,lt2)) #[4, 10, 18]
```


8. 문제

아래와 같이 2차원 행렬 두개의 내적을 구하는 함수 lt_dot() 함수를 만드시오.

추가조건

- 행렬의 요소 검사 후 예외처리
- 행렬의 크기가 잘못된 경우 예외 처리

실행결과

```
lt1 = [[1,2],[3,4]]
lt2 = [[5],[6]]

print(lt_dot(lt1,lt2)) #[[17], [39]]
```