**REPORT**

**[ 과제 : NP문제지(1) ]**



|  |  |
| --- | --- |
| 과 목 명 | 파이썬과학프로그래밍기초 |
| 교 수 명 | 김 병 정 |
| 학 번 | 20237107 |
| 작 성 자 | 하 태 영 |
| 제 출 일 | 2025.05.11 |

**한림대학교**

텍스트, 스크린샷, 폰트, 대수학이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

np1 = np.arange(0, 10) # 0 ~ 9까지의 정수를 배열로 만들고, np1에 저장

print(np1) # np1 출력

np2 = np.array(list(reversed(np1))) # np1을 뒤집고, 리스트화하여 배열로 np2에 저장

print(np2) # np2 출력

폰트, 텍스트, 타이포그래피, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 영수증, 폰트, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0 ~ 10까지 10개의 랜덤 정수타입으로 np1에 저장

np1 = np.random.uniform(0, 10, 10).astype('int')

print(np1) # np1 출력

np2 = np1[np.where(np1%2==0)] # np1의 배열에서 짝수만 골라서 np2에 저장

print(np2) # np2 출력

폰트, 타이포그래피, 텍스트, 친필이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.**

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# .T = 전치 연산 (행과 열을 바꿈)

# 0 ~ 23까지의 정수로 3행 8열을 만들고, 행과 열을 전치하여 np1에 저장

np1= np.arange(0,24).reshape(3,8).T

print(np.transpose(np1)) # np1의 행과 열을 전치하여 3행 8열로 출력

텍스트, 폰트, 타이포그래피, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**텍스트, 스크린샷, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.**

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# .T = 전치 연산 (행과 열을 바꿈)

# 0 ~ 23까지의 정수로 배열을 만들고, 3행 8열의 전치인 8행 3열로 만들어서 ndarray에 저장

ndarray = np.arange(0, 24).reshape(3, 8).T

print(ndarray) # ndarray 출력

텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**텍스트, 스크린샷, 폰트, 영수증이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.**

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0 ~ 1 사이의 난수를 3행 3열로 만들어주고, 소수점 2자리까지의 값들을 np1에 저장

np1 = np.round(np.random.random((3, 3)), 2)

print(np1) # np1 출력

텍스트, 폰트, 타이포그래피, 그래픽이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.**

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0 ~ 23의 정수를 3개의 블록, 블록 1개당 4행 2열로 만들고 aaa에 저장

aaa = np.arange(0,24).reshape(3,4,2)

print(aaa) # aaa 출력

텍스트, 폰트, 스크린샷, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.**

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0 ~ 23의 정수를 3개의 블록, 블록 1개당 4행 2열로 만들고 ndarray에 저장

ndarray = np.arange(0, 24).reshape(3, 4, 2)

print(ndarray) # ndarray 출력

텍스트, 폰트, 스크린샷, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 방법1 : LC 이용

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0 ~ 17의 정수를 3행, 6열 형태의 배열로 생성 후 np1에 저장

np1 = np.arange(0, 18).reshape(3, 6)

# 1. i는 np1의 행(3)만큼 반복, j는 np1의 열(6)만큼 반복

# 2. np1의 원소가 3의 배수이거나 5의 배수이면

# 3. i와 j를 튜플로 result에 저장

result = [(i, j) for i in range(np1.shape[0]) for j in range(np1.shape[1])

if np1[i, j] % 3 == 0 or np1[i, j] % 5 == 0]

print(result) # result 출력

텍스트, 폰트, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 방법2 : np.where() 이용

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0 ~ 17의 정수를 3행, 6열 형태의 배열로 생성 후 np1에 저장

np1 = np.arange(0, 18).reshape(3, 6)

# np1의 3의 배수이거나 5의 배수인 경우 idx배열에 저장

idx = np.where((np1 % 3 == 0) | (np1 % 5 == 0))

# idx의 행과 열을 zip으로 묶어서 반복하여 i, j를 정수 형변환한 후 튜플의 형태로 result에 저장

result = [(int(i), int(j)) for i, j in zip(idx[0], idx[1])]

print(result) # result 출력



텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 방법1) 3중 반복문

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 2개의 블록을 만들고, 블록 1개당 3행 4열에 0 ~ 23의 정수를 넣고 np1에 저장

np1 = np.arange(0, 24).reshape(2, 3, 4)

# 1. i는 np1의 블록(2)만큼, j는 np1의 행(3)만큼, K는 np1의 열(4)만큼 반복

# 2. np1의 원소가 3의 배수이건 5의 배수이면

# 3. i, j, k를 튜플로 result에 저장

result = [(i, j, k) for i in range(np1.shape[0])

for j in range(np1.shape[1])

for k in range(np1.shape[2])

if np1[i, j, k] % 3 == 0 or np1[i, j, k] % 5 == 0]

print(result) # result 출력



# 방법2 : np.where() 이용

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 2개의 블록을 만들고, 블록 1개당 3행 4열에 0 ~ 23의 정수를 넣고 np1에 저장

np1 = np.arange(0, 24).reshape(2, 3, 4)

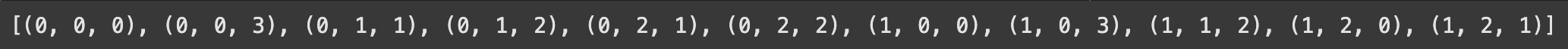
# np1의 3의 배수이거나 5의 배수인 경우 idx배열에 저장

idx = np.where((np1 % 3 == 0) | (np1 % 5 == 0))

# idx의 블록, 행, 열을 zip으로 묶어서 반복하여 i, j, k를 정수 형변환한 후 튜플의 형태로 result에 저장

result = [(int(i), int(j), int(k)) for i, j, k in zip(idx[0], idx[1], idx[2])]

print(result) # result 출력



텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

np1 = np.arange(0, 24).reshape(2, 3, 4)

np2 = np1.sum(axis=0) # 블록 기준의 합

np3 = np1.sum(axis=1) # 행 기준의 합

np4 = np1.sum(axis=2) # 열 기준의 합

print(np1) # np1 출력

텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0 ~ 1 사이 난수를 소수점 2자리 값으로 np1에 저장

np1 = np.round(np.random.random((3, 6)), 2)

max = np.max(np1, axis=1) # np1 행 기준으로 최대값을 max에 저장

min = np.min(np1, axis=1) # np1 행 기준으로 최소값을 min에 저장

print(max) # max 출력

print(min) # min 출력

텍스트, 폰트, 타이포그래피, 그래픽이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 대수학이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0 ~ 1 사이 난수를 소수점 2자리 값으로 np1에 저장

np1 = np.round(np.random.random((3, 6)), 2)

max = np.max(np1, axis=0) # np1 열 기준으로 최대값을 max에 저장

min = np.min(np1, axis=0) # np1 열 기준으로 최소값을 min에 저장

print(max) # max 출력

print(min) # min 출력

텍스트, 폰트, 그래픽, 타이포그래피이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 영수증이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np

# np1 배열 생성

np1 = np.array([0, 5, 2, 6, 8, 6, 3, 6, 3, 9])

# 첫 번째 6과 마지막 6 사이의 값들 추출 (6은 포함되지 않음)

np2 = np1[np.where(np1 == 6)[0][0]+1:np.where(np1 == 6)[0][-1]]

np2 = np2[np.where(np2 != 6)]

# 결과 출력

print(np2)

텍스트, 폰트, 스크린샷, 그래픽이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np이름으로 불러오기

# 정수형 8행, 8열 배열을 0으로 채우고 np1에 저장

np1 = np.zeros((8, 8), dtype=int)

np1[0, :] = 1 # np1의 첫 번째 행의 값을 1로 저장

np1[-1, :] = 1 # np1의 마지막 행의 값을 1로 저장

np1[:, 0] = 1 # np1의 첫 번째 열의 값을 1로 저장

np1[:, -1] = 1 # np1의 마지막 열의 값을 1로 저장

print(np1) # np1 출력

스크린샷, 폰트, 사각형, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np이름으로 불러오기

def myfct(row, col): # 사용자 정의 함수 myfct(행, 열)

# row, col를 정수형 타입 배열을 만들고, 0으로 채워 arr에 저장

arr = np.zeros((row, col), dtype=int)

arr[0, :] = 1 # arr의 첫 번째 행의 값을 1로 저장

arr[-1, :] = 1 # arr의 마지막 행의 값을 1로 저장

arr[:, 0] = 1 # arr의 첫 번째 열의 값을 1로 저장

arr[:, -1] = 1 # arr의 마지막 열의 값을 1로 저장

return arr # arr 반환

np1 = myfct(3, 8) # myfct(3, 8) 결과를 np1에 저장

print(np1) # np1 출력

폰트, 스크린샷, 타이포그래피, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np

a = np.array([1, 2, 3])

b = np.array([4, 5, 6])

c = np.array([[1, 2, 3],[4, 5, 6]])

d = np.array([[7, 8, 9],[10, 11, 12]])

# np.hstack([]) = 수평(horizontal, 가로)으로 붙여서 하나의 배열로 만든다.

# a, b를 가로로 붙여서 1차원 배열을 만든다.

print("1-1. np.hstack([a,b])")

print(np.hstack([a,b]))

print()

# c, d를 가로로 붙여서 1차원 배열을 만든다.

print("1-2. np.hstack([c,d])")

print(np.hstack([c,d]))

print()

# a, d를 가로로 붙여서 1차원 배열을 만든다.

# 행 개수가 다르기 때문에 오류이다.

print("1-3. np.hstack([a,d]) \n","행 개수가 다르기 때문에 오류이다.")

print()

# a와 마지막 원소를 제외한 b를 가로로 붙여서 1차원 배열을 만든다.

print("1-4. np.hstack([a,b[:-1]]) \n", np.hstack([a,b[:-1]]))

print()

# c와 마지막 원소를 제외한 d를 가로로 붙여서 1차원 배열을 만든다.

# 행 개수가 다르기 때문에 오류이다.

print("1-5. np.hstack([c,d[:,1:]]) \n", "행 개수가 다르기 때문에 오류이다.")

print()

# np.r\_[] = 1차원이면 가로로 붙이고, 2차원 이상이면 아래 행으로 붙여서 하나의 배열로 만든다.

# a, b를 가로로 붙여서 1차원 배열을 만든다.

print("1-6. np.r\_[a, b] \n", np.r\_[a, b])

print()

# a, b를 행 방향(아래)으로 붙여서 2차원 배열을 만든다.

print("1-7. np.r\_[[a], [b]] \n", np.r\_[[a], [b]])

print()

# a, c를 행 방향(아래)으로 붙여서 2차원 배열을 만든다.

print("1-8. np.r\_[[a], c] \n", np.r\_[[a], c])

print()

# a, c를 행 방향(아래)으로 붙여서 2차원 배열을 만든다.

print("1-9. np.r\_[c, d] \n", np.r\_[c, d])

print()

# np.c\_[] = 열 방향(가로)으로 붙여서 1차원 배열로 만드는 함수

# 배열 c, b 열 방향(가로)으로 붙여서 1차원 배열로 만든다.

print("1-10. np.c\_[c, d] \n", np.c\_[c, d])

print()

# np.concatenate((), axis=0)

# 결과가 1차원 배열이면 가로로 붙이고, 2차원 배열이면 세로로 붙인다.

# a, b를 가로로 붙여서 1차원 배열을 만든다.

print("1-11. np.concatenate((a,b), axis=0) \n", np.concatenate((a,b), axis=0))

print()

# 0 ~ 5까지 0.5씩 증가하여 가로로 붙인다.

print("1-12. np.r\_[0:5:0.5] \n", np.r\_[0:5:0.5])

print()

# 0 ~ 5까지 0.5씩 증가하여 세로로 붙인다.

print("1-13. np.c\_[0:5:0.5] \n", np.c\_[0:5:0.5])

print()

# np.vstack() = 수직으로 붙여서 하나의 배열로 만드는 함수

# a, b를 수직으로 붙여서 1차원 배열로 만든다.

print("2-1. np.vstack([a,b]) \n", np.vstack([a,b]))

print()

# c, d를 수직으로 붙여서 1차원 배열로 만든다.

print("2-2. np.vstack([c,d]) \n", np.vstack([c,d]))

print()

# a, c를 수직으로 붙여서 1차원 배열로 만든다.

print("2-3. np.vstack([a,c]) \n", np.vstack([a,c]))

print()

# a, b를 수직으로 붙여서 2차원 배열로 만든다.

print("2-4. np.r\_[[a], [b]] \n", np.r\_[[a], [b]])

print()

# a, c를 수직으로 붙여서 2차원 배열로 만든다.

print("2-5. np.r\_[[a], c] \n", np.r\_[[a], c])

print()

# c, d를 수직으로 붙여서 1차원 배열로 만든다.

print("2-6. np.r\_[c, d] \n", np.r\_[c, d])

print()

# a, b를 행 기준(axis=0)으로 붙여서 1차원 배열을 만든다.

print("2-7. np.concatenate((a,b), axis=0) \n", np.concatenate((a,b), axis=0))

print()

print("2-8. np.concatenate((a,b), axis=1) \n", "1차원 배열은 열이 없기 때문에 열 기준(axis=1)으로 붙일 수 없다.")

print()

# a, b를 행 기준으로 붙이고 2차원 배열을 만든다.

print("2-9. np.concatenate(([a],[b]), axis=0) \n", np.concatenate(([a],[b]), axis=0))

print()

# a, b를 열 기준으로 붙이고 2차원 배열을 만든다.

print("2-10. np.concatenate(([a],[b]), axis=1) \n", np.concatenate(([a],[b]), axis=1))

print()

# np.column\_stack() = 여러 배열을 열 방향(가로, axis=1)으로 붙여 2차원 배열을 만드는 함수

# a, b가 1차원 배열이면 각각 열 벡터로 변환하여 2차원 배열을 만든다.

print("3-1. np.column\_stack([a,b]) \n", np.column\_stack([a, b]))

print()

# c, d가 2차원 배열이면, 그대로 열 방향(가로)으로 붙여 2차원 배열을 만든다.

print("3-2. np.column\_stack([c,d]) \n", np.column\_stack([c, d]))

print()

# np.c\_[] = 여러 배열을 열 방향(가로, axis=1)으로 붙여 2차원 배열을 만드는 도구

# a, b가 1차원 배열이면, 각각 열 벡터로 변환되어 2차원 배열이 된다.

print("3-3. np.c\_[a, b] \n", np.c\_[a, b])

print()

# c, d가 2차원 배열이면, 그대로 열 방향(가로)으로 붙여 2차원 배열을 만든다.

print("3-4. np.c\_[c, d] \n", np.c\_[c, d])

print()

# a\_reshaped와 b\_reshaped를 열 방향(가로, axis=1)으로 붙여서 하나의 2차원 배열로 만든다.

print("3-5. np.concatenate((a\_reshaped, b\_reshaped), axis=1 \n", "a\_reshaped, b\_reshaped가 선언되지 않아 오류")

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다. 텍스트, 폰트, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 방법1 : zeros() 이용 ( 0 배열을 만들고, 1로 값을 치환하는 방법)

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 정수형 3행, 8열 배열을 0으로 채우고 np1에 저장

np1 = np.zeros((3, 8), dtype=int)

np1[0, :] = 1 # np1의 첫 번째 행의 값을 1로 저장

np1[-1, :] = 1 # np1의 마지막 행의 값을 1로 저장

np1[:, 0] = 1 # np1의 첫 번째 열의 값을 1로 저장

np1[:, -1] = 1 # np1의 마지막 열의 값을 1로 저장

print(np1)

폰트, 텍스트, 타이포그래피, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 방법2 : 리스트를 이용 (LC 2차원 행렬을 만드는 방법)

rows, cols = 3, 8 # 행은 3, 열은 8

np1 = [

# 한 행(row)을 만듭니다. 각 열(column)마다

# 행 인덱스가 0이거나 마지막이거나 열 인덱스가 0 이거나 마지막이면 1, 그렇지 않으면 0 값을 결정한다.

[1 if r== 0 or r == rows-1 or c == 0 or c == cols-1 else 0 for c in range(cols)]

# 위 행을 전체 행(rows)만큼 반복해서 2차원 리스트를 만듭니다.

for r in range(rows)

]

# np1 리스트의 각 행(row)을 출력합니다.

for row in np1:

print(row) # 한 행씩 출력

폰트, 스크린샷, 타이포그래피, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 방법3 : vstack 을 이용

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

rows, cols = 3, 8 # 행은 3, 열은 8

# 정수형 1행, 8열 배열을 1로 채우고 top\_bottom 저장

top\_bottom = np.ones((1, cols), dtype=int)

# 정수형 1행, 8열 배열을 0으로 채우고 middle 저장

middle = np.zeros((1, cols), dtype=int)

middle[0, 0] = 1 # 1행 1열은 1로 저장

middle[0, -1] = 1 # 1행 마지막열은 1로 저장

# top\_bottom, middle, top\_bottom을 수직으로 붙여서 np1에 저장

np1 = np.vstack([top\_bottom, middle, top\_bottom])

print(np1) # np1 출력

폰트, 텍스트, 타이포그래피, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 방법4 : hstack 을 이용

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

rows, cols = 3, 8 # 행은 3, 열은 8

# 정수형 3행, 1열 배열을 1로 채우고 side 저장

side = np.ones((rows, 1), dtype=int)

# 정수형 3행 6열 배열을 0으로 채우고 middle 저장

middle = np.zeros((rows, cols-2), dtype=int)

# side, middle, side 배열을 수평으로 붙여서 np1에 저장

np1 = np.hstack([side, middle, side])

np1[0, :] = 1 # np1의 1행은 전부 1로 저장

np1[-1, :] = 1 # np1의 마지막행은 전부 1로 저장

print(np1) # np1 출력

폰트, 텍스트, 타이포그래피, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

# 방법5: concatenate 이용

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

rows, cols = 3, 8 # 행은 3, 열은 8

# 정수형 3행, 1열 배열을 1로 채우고 side 저장

side = np.ones((rows, 1), dtype=int)

# 정수형 3행 6열 배열을 0으로 채우고 middle 저장

middle = np.zeros((rows, cols-2), dtype=int)

# side, middle, side 배열을 행 기준으로 붙여서 np1에 저장

np1 = np.concatenate([side, middle, side], axis=1)

np1[0, :] = 1 # np1의 1행은 전부 1로 저장

np1[-1, :] = 1 # np1의 마지막행은 전부 1로 저장

print(np1) # np1 출력

폰트, 텍스트, 스크린샷, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 영수증, 폰트, 화이트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 영수증, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0~9까지 숫자 중에서 5개를 임의로 뽑아서 np1에 저장

np1 = np.random.choice(np.arange(10), size=5, replace=True)

# np.pad로 앞뒤에 0 추가

np2 = np.pad(np1, (1, 1), mode='constant', constant\_values=0)

print(np1) # np1 출력

print(np2) # np2 출력

폰트, 텍스트, 번호, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0~9까지 숫자 중에서 무작위로 12개(3x4)를 뽑아 배열 생성

# replace=True: 중복 허용

np1 = np.random.choice(np.arange(10), size = (3, 4), replace=True)

# np1 배열 주변을 4로 패딩

# 상하좌우 1줄씩 추가 (패딩 후 크기: 5x6)

# mode='constant': 고정값 채우기

# constant\_values=4: 채울 값 지정

np2 = np.pad(np1, ((1, 1), (1, 1)), mode='constant', constant\_values=4)

print(np1) # np1 출력

print(np2) # np2 출력

텍스트, 폰트, 스크린샷, 타이포그래피이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0~9까지 숫자 중에서 무작위로 30개(5x6)를 뽑아 배열 생성

# replace=True: 중복 허용

np1 = np.random.choice(np.arange(15), size = (5, 6), replace=True)

# 가장 가까운 값으로 상하좌우 1줄씩 패딩

pad\_np1 = np.pad(np1, ((1,1),(1,1)), mode='edge')

# np1과 같은 크기의 0.0(실수형) 배열을 np2에 생성

np2 = np.zeros\_like(np1, dtype=float)

for i in range(5): # i는 0~4까지 반복

for j in range(6): # j는 0~5까지 반복

# (i, j)를 좌상단으로 하는 3x3 영역(부분 배열)을 mask에 저장

mask = pad\_np1[i:i+3, j:j+3]

# mask 배열들의 평균값을 np2에 저장

np2[i, j] = np.mean(mask)

# 소수점 2자리로 출력 (과학적 표기법 억제)

np.set\_printoptions(precision=2, suppress=True)

print(np1) # np1 출력

# np.array2string : 배열을 문자열로 변환

# formatter를 사용해 모든 실수를 항상 소수점 둘째자리까지(빈자리는 0으로) 문자열로 변환

print(np.array2string(np2, formatter={'float\_kind': lambda x: "%.2f" % x}))

텍스트, 폰트, 스크린샷, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import copy # copy 라이브러리 불러오기

# np1: 2차원 배열의 리스트 값 생성 (Nested List Comprehension 사용)

# i(0~2)가 행을, j(0~3)가 열을 생성하며, 각 요소는 i\*4 + j +1로 계산

np1 = [[i \* 4 + j + 1 for j in range(4)] for i in range(3)]

# np2: np1의 깊은 복사 후 중심부(2행 2-3열) 값을 0으로 변경

np2 = copy.deepcopy(np1)

np2[1][1:3] = [0, 0] # 2행(인덱스 1)의 2-3열(인덱스 1:3)을 0으로 할당

# 결과 출력

for k in np1:

print(k)

print()

for k in np2:

print(k)

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# np1: 2차원 배열의 리스트 값 생성 (Nested List Comprehension 사용)

# i(0~2) 행을 j(0~3) 열을 생성하며, 각 요소는 i\*4+j+1로 계산하여 np1에 저장

np1 = [[i \* 4 + j + 1 for j in range(4)] for i in range(3)]

# np1의 상하좌우 행과 열의 1줄씩 가장자리와 같은 값으로 추가하여 np2에 저장

np2 = np.pad(np1, pad\_width=((1, 1), (1, 1)), mode='edge')

# 결과 출력

for k in np1:

print(k)

print(np2)

텍스트, 폰트, 스크린샷, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# i(0~2) 행을 j(0~3) 열을 생성하며, 각 요소는 i\*4+j+1로 계산하여 np1에 저장

np1 = [[i \* 4 + j + 1 for j in range(4)] for i in range(3)]

result = [] # 시계방향 테두리 값을 저장할 리스트

sum = 0 # 테두리 값의 합계 계산용 변수

# 1. 맨 위 행(0행) 전체 추가 (왼→오)

result += np1[0]

# 2. 오른쪽 열(3열) 추가 (위→아래, 1행~2행)

for i in range(1, 3):

result.append(np1[i][3])

# 3. 맨 아래 행(2행) 역순 추가 (오→왼, 2열~0열)

result += np1[2][2::-1]

# 4. 왼쪽 열(0열) 추가 (아래→위, 1행만 추가)

for i in range(1, 0, -1): # i=1만 반복 (0행은 이미 추가됨)

result.append(np1[i][0])

# np1 배열 출력

for row in np1:

print(row)

# result 출력

print(result)

# 합계 계산 및 출력

for num in result:

sum += num

print(sum)

텍스트, 폰트, 스크린샷, 타이포그래피이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트, 영수증이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0~12의 값을 3행 4열로 만들어 np1에 저장

np1 = np.arange(12).reshape(3, 4)

row\_sums = np1.sum(axis=1) # np1의 행 기준의 합

col\_sums = np1.sum(axis=0) # np1의 열 기준의 합

np2 = np.vstack([np1, col\_sums]) # np1의 열의 합을 np1에 수직으로 붙여서 np2에 저장

total\_sum = np1.sum() # np1의 전체 합을 total\_sum에 저장

# row\_sums에 total\_sum을 추가하고, 수직행렬로 만듬

row\_sums = np.append(row\_sums, total\_sum).reshape(-1, 1)

# np2에 row\_sums를 수평으로 붙여서 np2에 저장

np2 = np.hstack([np2, row\_sums])

# 결과 출력

print(np1)

print(np2)

텍스트, 폰트, 스크린샷, 타이포그래피이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

import numpy as np # numpy 라이브러리를 np라는 이름으로 불러오기

# 0~23까지의 수를 2블록 3행 4열 3차원 배열로 생성

# ▶ reshape(2,3,4): (블록, 행, 열)

np1 = np.arange(24).reshape(2, 3, 4)

print(np1)

# 블록별 더할 값 [2, 4]를 3차원 형태(2,1,1)로 변환

# ▶ -1: 차원 자동 계산 (2개 블록)

# ▶ 1x1: 브로드캐스팅을 위해 행/열 차원 확장

add\_values = np.array([2, 4]).reshape(-1, 1, 1)

# 브로드캐스팅으로 각 블록에 다른 값 더하기

# ▶ 첫 번째 블록: 모든 요소 +2

# ▶ 두 번째 블록: 모든 요소 +4

np2 = np1 + add\_values

print(np2)

텍스트, 폰트, 스크린샷, 타이포그래피이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.