22. 07 . 19 火 _ 난수 매트릭스

정렬 (Sorting) : 특정 기준에 따라 원소를 나열 하는 것

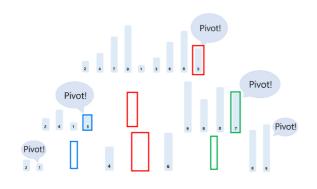
	1	2	3	4	5
	6	7	8	9	10
	11	12	13	14	15
	16	17	18	19	20
열	21	22	23	24	25
최소값	1	2	3	4	5
최대값	21	22	23	24	25
중간값	11	12	13	14	15
행					
최소값	최대값	중간값			
1	5	3			
6	10	8			
11	15	13			
16	20	18			
21	25	23			
전체			•		
최소값	1]			
최대값	25]			
중간값	13	1			

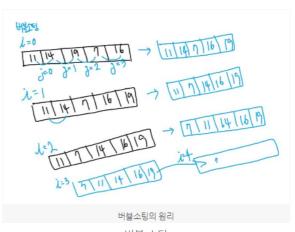
중간값을 칮		- 17	로그램 결		141	
			포그림 를	TA EX A	INI	
						_
	0	1	2	3	4	5
		6	7	8	9	10
		11	12	13	14	15
		16	17	18	19	20
	열	21	22	23	24	25
	최소값	1	2	3	4	5
	최대값	21	22	23	24	25
	중간값	11	12	13	14	15
	최소깂	최대값	중간값			
	1	5	3			
	6	10	8			
	11	15	13			
	16	20	18			
	21	25	23			
	AL AII					
	선제					
	최소값	1	-			
	최대값	25	_			
	중간값	13				

문제푸는 2가지 방법

- 1. sort API 사용 하는 방법
- 2. 구글링 → 1)버블 소팅 , 2)퀵소팅

3.





버블 소팅

```
숫자를 두개씩 비교해서
   [5, 2, 3, 4, 1]
                                                   이매진쌤의 파이썬 연구
                                                  blog.naver.com/leeaain
                       큰 숫자는 뒤로 보낸다.
   [2, 5, 3, 4, 1]
   [2, 3, (5, 4), 1]
                      첫번째 턴에서 가장 큰 숫자가 맨 뒤로 밀려나게 된다.
   [2, 3, 4, (5, 1)]
                      맨 뒷자리는 이미 확고하게 가장 큰 수 이다.
   [2, 3, 4, 1, X]
결과
                      그러니 또 비교해볼 필요가 없다.
   [2, 3, 4, 1, 5]
                      따라서 두번째 턴에서는 제외한다.
   [2, 3, 4, 1, 5]
                      그 앞자리도 나머지 중에서 가장 큰 수로 확정되으므로
   [2, 3, 1, 1, 1, 1]
                      다음 턴부터 제외한다.
   [2, 3, 1, X, X]
                      그 앞자리도 남은 수 중 가장 큰 수로 확정되었다.
   [2, 1, 3, 4, 5]
                     다음 턴부터 제외한다.
결과
   [1, 2, 3, 4, 5]
                      이제 마지막 남은 두 개의 숫자만 비교하여 자리를 바꾸어주면 된다.
격과
   ----모든 정렬이 끝났다.
    결과: [1, 2, 3, 4, 5]
```

1.

```
list = [ 5, 9 , 4, 8, 1]

# 정렬 (Sorting) : 특정 기준에 따라 원소를 나열 하는 것.

# 숫자를 정렬한다----> 수의 크기를 기준으로 원소를 나열 하는 것.

# 1) 오름차순 -> 1, 4, 5, 8, 9

list.sort()
print(list)

# 2) 내림차순 -> 9, 8, 5, 4, 1
list.sort(reverse=True)
print(list)

# 두 가지 방법

# 1) Sort API 사용 하는 방법

# 2) 구글링 -> 버블 소팅, 퀵 소팅
```

2. 재일_난수 매트릭스

```
import random

MyList = [] # 무작위로 받는 리스트
SortingList = [] # 정렬 리스트
```

```
# 2차원 리스트
row_List = [[],[],[],[]] # [ [0] , [1] , [2] , [3] , [4] ] -> 5개
# 1. while 로 같은거 안들어 오게 계속 무한 반복
while len(MyList)< 25 : # 0 ~24 개
   # 1,50 사이 수를 무작위로 돌리기
   Element_1 = random.randint(1,50)
   # 리스트안에 요소가 안들어 있으면 넣기
   if not Element_1 in MyList :
      MyList.append(Element_1)
# 2. 1,50사이 순서대로 크기 잡아가면서 정렬
# == MyList.sort()
for index in range (1,50+1):
   if index in MyList :
      SortingList.append(index)
# 3. 화면 출력용 5 X 5
for value in range (len(SortingList)) :
   # 원소 추출
   Element_2 = SortingList[value]
   print(Element_2, "\t", end= "" )
   # 띄워쓰기
   if (value+1) %5 == 0 :
      print()
# 4. 2차원 리스트
Count = 0 # 전체 카운트
Sorting_count = 0 # 솔팅 카운트
# 리스트 안에 리스트가 5개 까지 반복
while Count < len(SortingList) :</pre>
   # 요소 뽑기
   \verb"row_List[Sorting_count].append(SortingList[Count])"
   if len(row_List[Sorting_count]) == 5 :
      Sorting_count += 1
   Count += 1
# 5. 출력부
# 5.1 열
print()
print("열")
# 최소 값
print("최소값","\t",end="" )
for index in range (len(row_List[0])) :
  print(row_List[0][index],"\t" , end="")
print()
#최대 값
print("최대값","\t",end="" )
for index in range (len(row_List[4])) :
   print(row_List[4][index],"\t" , end="")
print()
# 중간 값
print("중간값","\t",end="" )
for index in range (len(row_List[4])) :
   print(row_List[len(row_List)//2][index],"\t" , end="")
print()
# 5.2 행
print()
print("행")
print("최소값", "\t\t", "최대값", "\t", "중간값")
```

```
# 2차원 리스트 ( 5 X 5 를 행,열 로 활용 )
                                       # 0 1 2 3 4
for row in range (len(row_List)) :
    for col in range (len(row_List)) : # 0 1 2 3 4
        # 최소값
        if col == 0:
           print (row_List[row][col],"\t\t",end=" ")
        # 최대 값
        elif col == 2 :
           # 숫자 조절
           col = 4
           print (row_List[row][col],"\t\t",end=" ")
        # 중간 값
        elif col == len(row_List)-1 :
           # 숫자 조절
           col = 2
           print (row_List[row][col],"\t\t",end=" ")
# 5.3 전체
print()
print("最小値 : ",SortingList[0])
print("中間値 : ",SortingList[12])
print("最大値 : ",SortingList[len(SortingList)-1])
```

3. 거품 정렬 (버블 소팅)

```
def bubbleSort(x):
  length = len(x)-1
  for i in range(length):
    for j in range(length-i):
        if x[j] > x[j+1]:
            x[j], x[j+1] = x[j+1], x[j]
    return x
```

4.