| Examination: Quiz 1  Duration: 30 mins | Semester: Fall 2024  Full Marks: 15 |
| --- | --- |

CSE 220: Data Structures

| Name: | ID: | Section: |
| --- | --- | --- |

**Allowed:** numpy array, .shape, range(), len(). print()

**Not Allowed:** python list or any other data structure except Array, .append(), max(), min(), negative indexing

**Q1)** Bob has a list of numbers stored in a two-dimensional (2D) array. He wants to sort these numbers in ascending order (from smallest to largest) using an out-of-place solution.

1. First, he flattens the array into a single linear array.
2. He creates a new empty linear array to store the sorted numbers.
3. Then, he repeatedly finds the maximum number from the flattened array, places it in the appropriate position in the new array, and removes it from the original flattened array.
4. Bob continues this process until the flattened array is empty and the new array is filled with sorted numbers. Remember, each time Bob removes an element from the original flattened array, he adjusts the array to maintain its property.
5. Finally, he reshapes the sorted 1D array back into the original 2D structure.

Write a function that takes a 2D array as input, performs Bob's sorting idea, and returns the sorted array. The output should be a sorted 2D array of the same dimensions.

def sortArray(arr):

# write code

return sortedArr

Sample Input and Output:

Input: [[7, -1], [8, 16]], After Flattening: [7, -1, 8, 16]

Output: [[-1, 7], [8, 16]]

Input: [[4, 3, 6], [-2, 5, 7]], After Flattening: [4, 3, 6, -2, 5, 7]

Output: [[-2, 3, 4], [5, 6, 7]]

| Examination: Quiz 1  Duration: 30 mins |  | Semester: Fall 2024  Full Marks: 15 |
| --- | --- | --- |

CSE 220: Data Structures

| Name: | ID: | Section: |
| --- | --- | --- |

**Allowed:** numpy array, .shape, range(), len(). print()

**Not Allowed:** python list or any other data structure except Array, .append(), max(), min(), negative indexing

**Q1)** Bob has a list of numbers stored in a two-dimensional (2D) array. He wants to sort these numbers in

descending order (from largest to smallest) using an out-of-place solution.

1. First, he flattens the array into a single linear array.
2. He creates a new empty linear array to store the sorted numbers.
3. Then, he repeatedly finds the minimum number from the flattened array, places it in the appropriate position in the new array, and removes it from the original flattened array.
4. Bob continues this process until the flattened array is empty and the new array is filled with sorted numbers. Remember, each time Bob removes an element from the original flattened array, he adjusts the array to maintain its property.
5. Finally, he reshapes the sorted 1D array back into the original 2D structure.

Write a function that takes a 2D array as input, performs Bob's sorting idea, and returns the sorted array. The output should be a sorted 2D array of the same dimensions.

def sortArray(arr):

# write code

return sortedArr

Sample Input and Output:

Input: [[7, -1], [8, 16]], After Flattening: [7, -1, 8, 16]

Output: [[16, 8], [7, -1]]

Input: [[4, 3, 6], [-2, 5, 7]], After Flattening: [4, 3, 6, -2, 5, 7]

Output: [[7, 6, 5], [4, 3, -2]]