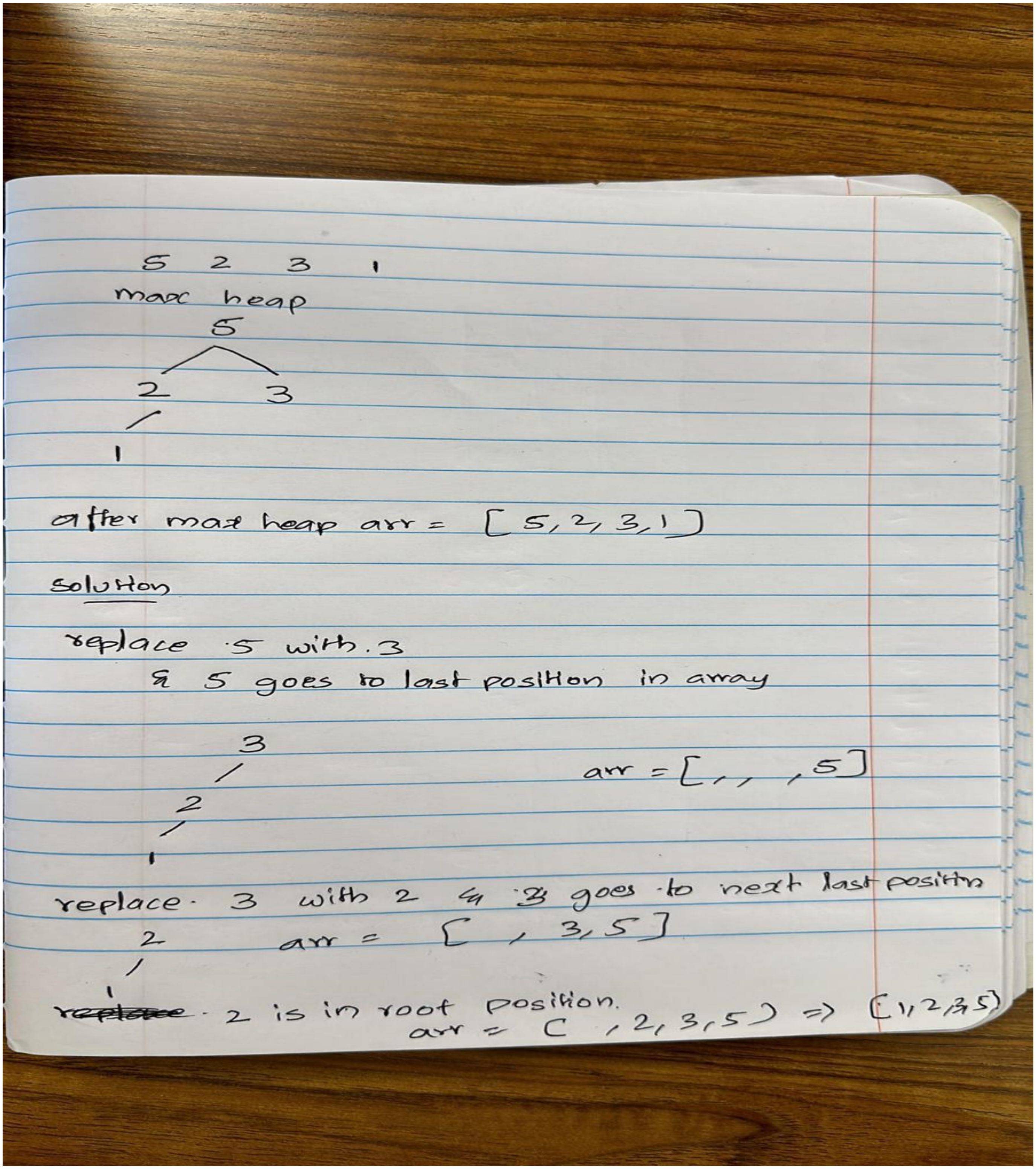
Week9hw2

Manual solution



5 2 3 1

max heap

5

2 3

1

after max heap arr = [5,2,3,1]

solution

replace 5 with 3

& 5 goes to last position in array

3

2

1

arr = [, , ,5]

replace 3 with 2 & 3 goes to next last positin

2 arr = [, 3,5]

1

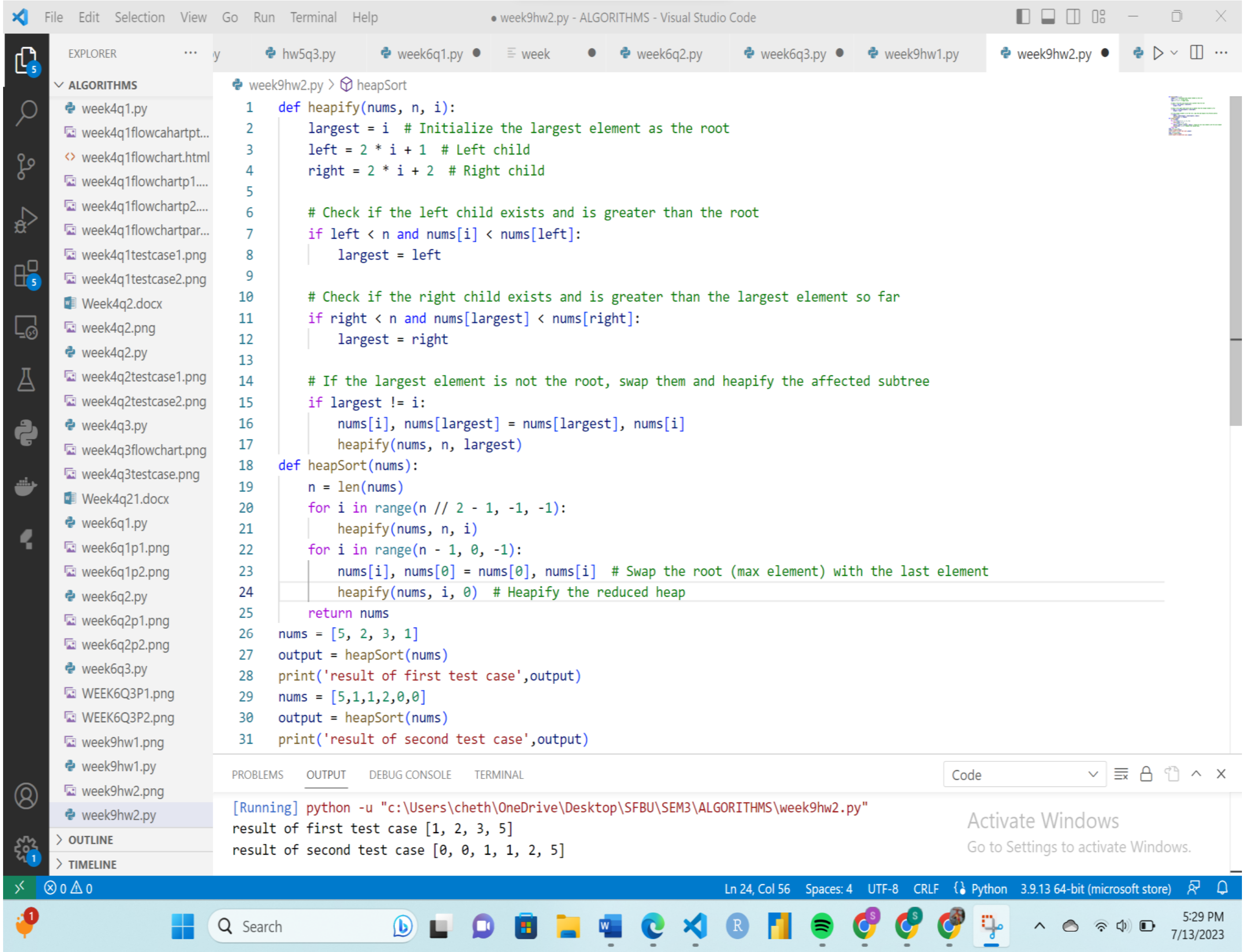
~~replace~~ 2 is in root position.

arr = (,2,3,5) => [1,2,3,5)

Chatgpt code

```
def heapify(nums, n, i):
    largest = i  # Initialize the largest element as the root
    left = 2 * i + 1  # Left child
    right = 2 * i + 2  # Right child

    # Check if the left child exists and is greater than the root
    if left < n and nums[i] < nums[left]:
        largest = left

    # Check if the right child exists and is greater than the largest element so far
    if right < n and nums[largest] < nums[right]:
        largest = right

    # If the largest element is not the root, swap them and heapify the affected subtree
    if largest != i:
        nums[i], nums[largest] = nums[largest], nums[i]
        heapify(nums, n, largest)


def heapSort(nums):
    n = len(nums)

    # Build a max heap
    for i in range(n // 2 - 1, -1, -1):
        heapify(nums, n, i)

    # Extract elements one by one from the heap
    for i in range(n - 1, 0, -1):
        nums[i], nums[0] = nums[0], nums[i]  # Swap the root (max element) with the last element
        heapify(nums, i, 0)  # Heapify the reduced heap

    return nums
nums = [5, 2, 3, 1]
output = heapSort(nums)
print(output)
```

test case



```python
def heapify(nums, n, i):
    largest = i  # Initialize the largest element as the root
    left = 2 * i + 1  # Left child
    right = 2 * i + 2  # Right child

    # Check if the left child exists and is greater than the root
    if left < n and nums[i] < nums[left]:
        largest = left

    # Check if the right child exists and is greater than the largest element so far
    if right < n and nums[largest] < nums[right]:
        largest = right

    # If the largest element is not the root, swap them and heapify the affected subtree
    if largest != i:
        nums[i], nums[largest] = nums[largest], nums[i]
        heapify(nums, n, largest)
def heapSort(nums):
    n = len(nums)
    for i in range(n // 2 - 1, -1, -1):
        heapify(nums, n, i)
    for i in range(n - 1, 0, -1):
        nums[i], nums[0] = nums[0], nums[i]  # Swap the root (max element) with the last element
        heapify(nums, i, 0)  # Heapify the reduced heap
    return nums
nums = [5, 2, 3, 1]
output = heapSort(nums)
print('result of first test case',output)
nums = [5,1,1,2,0,0]
output = heapSort(nums)
print('result of second test case',output)
```

```
[Running] python -u "c:\Users\cheth\OneDrive\Desktop\SFBU\SEM3\ALGORITHMS\week9hw2.py"
result of first test case [1, 2, 3, 5]
result of second test case [0, 0, 1, 1, 2, 5]
```