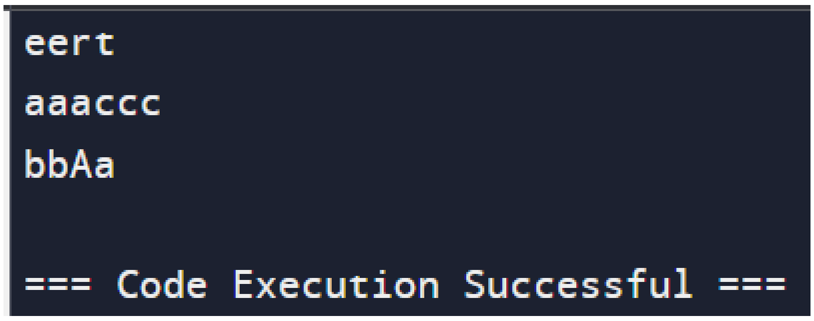
51. Sort Characters By Frequency Given a string s, sort it in decreasing order based on the frequency of the characters. The frequency of a character is the number of times it appears in the string. Return the sorted string. If there are multiple answers, return any of them. Example 1: Input: s = "tree" Output: "eert" Explanation: 'e' appears twice while 'r' and 't' both appear once. So 'e' must appear before both 'r' and 't'. Therefore "eetr" is also a valid answer

PROGRAM:-

```
from collections import Counter
import heapq

def frequencySort(s):
    # Step 1: Count the frequency of each character
    freq = Counter(s)

    # Step 2: Build a max heap based on frequency
    max_heap = [(-count, char) for char, count in freq.items()]
    heapq.heapify(max_heap)

    # Step 3: Build the result string
    result = []
    while max_heap:
        count, char = heapq.heappop(max_heap)
        result.append(char * -count)

    return ''.join(result)

# Example usage:
s = "tree"
print(frequencySort(s))  # Output: "eert" or "eetr"

# Another example:
s = "cccaaa"
print(frequencySort(s))  # Output: "cccaaa" or "aaaccc"

# Yet another example:
s = "Aabb"
print(frequencySort(s))  # Output: "bbAa" or "bbaA"
```

OUTPUT:-



```
eert
aaaccc
bbAa


=== Code Execution Successful ===
```

TIME COMPLEXITY:-O(n+m log n)