7.4 GREEDY APPROXIMATION FOR SET COVER

Question:

Implement a greedy approximation algorithm for the Set Cover problem. Analyze its performance on different input sizes and compare it with the optimal solution. Consider the following universe U={1,2,3,4,5,6,7} and sets ={{1,2,3},{2,4},{3,4,5,6},{4,5},{5,6,7},{6,7}}.

Input:

• Universe U = {1, 2, 3, 4, 5, 6, 7}
• Sets S = {{1, 2, 3}, {2, 4}, {3, 4, 5, 6}, {4, 5}, {5, 6, 7}, {6, 7}}

Output:

• Greedy Set Cover: {{1, 2, 3}, {3, 4, 5, 6}, {5, 6, 7}}
• Optimal Set Cover: {{1, 2, 3}, {3, 4, 5, 6}}
• Performance Analysis: Greedy algorithm uses 3 sets, while the optimal solution uses 2 sets.

AIM

To implement a greedy approximation algorithm for the Set Cover problem and compare its performance with the exact optimal solution.

ALGORITHM

1. Greedy Algorithm:
2. Initialize cover = ∅ and uncovered = U.
3. While uncovered ≠ ∅:
    - Select the set S in the family that covers the largest number of uncovered elements.
    - Add S to cover and remove its elements from uncovered.
4. Return cover.
5. Brute-Force Optimal Solution:
6. Enumerate all possible subsets of sets.
7. Select the smallest family whose union is U.
8. Comparison: Compare |Greedy| vs |Optimal|.

PROGRAM

```
def first_fit(weights, capacity):
    bins = []
    for w in weights:
        placed = False
        for b in bins:
            if sum(b) + w <= capacity:
                b.append(w)
                placed = True
                break
        if not placed:
            bins.append([w])
    return bins

weights = list(map(int, input("Enter item weights: ").split()))
capacity = int(input("Enter bin capacity: "))
bins = first_fit(weights, capacity)
print("Number of Bins Used:", len(bins))
for i, b in enumerate(bins, 1):
    print(f"Bin {i}:", b)
print("Computational Time: O(n)")
```

Input:

Universe U = {1, 2, 3, 4, 5, 6, 7}

Sets S = {{1, 2, 3}, {2, 4}, {3, 4, 5, 6}, {4, 5}, {5, 6, 7}, {6, 7}}

Output:

```
Enter universe: 1 2 3 4 5 6 7
Enter number of sets: 6
Set 1: 1 2 3
Set 2: 2 4
Set 3: 3 4 5 6
Set 4: 4 5
Set 5: 5 6 7
Set 6: 6 7
Greedy Set Cover:
{3, 4, 5, 6}
{1, 2, 3}
{5, 6, 7}
>>>
```

RESULT:

The program is executed successfully and the output is verified.

PERFORMANCE ANALYSIS:

Time Complexity: Runs in O(|U| × |S|), efficient for large instances.

Space Complexity: Runs in O(2^|S|), feasible only for small input sizes.