Experiment Name: Maximizing Value using Fractional Knapsack with Greedy Method

Problem Statement:

You are given:

- A list of `n` items, each with a `value[i]` and `weight[i]`
- A knapsack with maximum weight capacity `W`

Goal: To maximize the total value by **possibly taking fractions** of items.

🧭 Step-by-Step Process:

Step 1: Input the data

- Number of items `n`
- Arrays of `value[i]` and `weight[i]` for each item
- Knapsack capacity `W`

Step 2: Calculate value-to-weight ratio

For each item, calculate:

```
ratio[i] = value[i] / weight[i]
```

Step 3: Sort items

Sort all items in **descending order of ratio**.

👉 **This ensures you take items that give the most value per weight first.**

Step 4: Start filling the knapsack

Initialize:

```
total_value = 0
remaining_capacity = W
```

Loop through the sorted items:

- If the full item fits (`weight[i] <= remaining_capacity`):
    - Take the whole item:

      ```
      total_value += value[i]
      remaining_capacity -= weight[i]
      ```

- Else:
    - Take the **fraction** of the item that fits:

      ```
      fraction = remaining_capacity / weight[i]
      total_value += value[i] * fraction
      remaining_capacity = 0
      break
      ```

Step 5: Output the result

Print:

```
Maximum total value = total_value
```

📌 Example:

- Items:
  Value = {60, 100, 120}
  Weight = {10, 20, 30}
- Capacity = 50

Ratios:

- Item 1: 60/10 = 6.0
- Item 2: 100/20 = 5.0
- Item 3: 120/30 = 4.0

Sorted by ratio:
Item 1, Item 2, Item 3

Greedy Selection:

- Take Item 1 (10) → value = 60
- Take Item 2 (20) → value = 100
- Take 2/3 of Item 3 (20/30) → value = 80

Total = 60 + 100 + 80 = 240