

Name: Maurya Patel R

En-No: 22162101014

Batch: 51

Batch: CBA

Institute of Computer Technology
B. Tech Computer Science and Engineering

Sub: Algorithm Analysis and Design

Practical 9

- A thief is robbing a store and can carry a maximal weight of W into his knapsack. There are n items available in the store and weight of i^{th} item is w_i and its profit is p_i . What items should the thief take?
- In this context, the items should be selected in such a way that the thief will carry those items for which he will gain maximum profit. Hence, the objective of the thief is to maximize the profit.
- Implement Program for fractional knapsack using Greedy design technique.

Note: First solve the example:

$W=60$

| Item | A | B | C | D |
|--------|-----|-----|-----|-----|
| Profit | 280 | 100 | 120 | 120 |
| Weight | 40 | 10 | 20 | 24 |

Sample Input:-

$p=[280,100,120,120]$

$w=[40,10,20,24]$

$W=60$

Sample Output:-

Profit [100, 280, 120, 120]

Weight [10, 40, 20, 24]
Ratio [10.0, 7.0, 6.0, 5.0]
[1, 1, 0.5, 0]
Total profit : 440.0

Code:

.py:

```
from flask import Flask, render_template, request

app = Flask(__name__)

# Function to perform fractional knapsack with sorted profits
def fractional_knapsack(p, w, W):
    n = len(p)
    # List to store (profit per weight, profit, weight)
    items = [(p[i] / w[i], p[i], w[i]) for i in range(n)]

    # Sort based on profit per weight in descending order
    items.sort(reverse=True, key=lambda x: x[0])

    total_profit = 0
    selected_items = [0] * n # Array to store the fraction of each item
    for i in range(n):
        profit_per_weight, profit, weight = items[i]
        if weight <= W:
            W -= weight
            total_profit += profit
            selected_items[i] = 1 # Take the entire item
        else:
            selected_items[i] = W / weight # Take a fraction of the item
            total_profit += profit * (W / weight)
            break

    return selected_items, total_profit, items

@app.route('/', methods=['POST', 'GET'])
def index():
    if request.method == 'POST':
        if 'profits' in request.form and 'weights' in request.form and 'capacity'
in request.form:
            # Retrieve input values from the form
            profits = request.form['profits'] # Example input: "280,100,120,120"
            weights = request.form['weights'] # Example input: "40,10,20,24"
            W = int(request.form['capacity']) # Max capacity as an integer
```

```

        # Split the input strings by commas and convert each value to int
        p = [int(x) for x in profits.split(',')]
        w = [int(x) for x in weights.split(',')]

        # Call the knapsack function
        selected_items, total_profit, items = fractional_knapsack(p, w, W)

        # Return the result to the front end
        return render_template('index.html',
                               selected_items=selected_items,
                               total_profit=total_profit,
                               items=items)

    else:
        return "Please provide all the required inputs."

    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=True)

```

.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Fractional Knapsack</title>
    <style>
        /* General styles for the page */
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
            text-align: center;
        }

        h1 {
            color: #333;
        }

        h2, h3 {

```

```
        color: #555;
    }

    form {
        margin: 20px auto;
        padding: 20px;
        border-radius: 8px;
        background-color: #fff;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        width: 300px;
    }

    label {
        font-size: 14px;
        color: #333;
    }

    input[type="text"], input[type="number"] {
        padding: 8px;
        margin: 10px 0;
        width: 100%;
        box-sizing: border-box;
        border: 1px solid #ccc;
        border-radius: 4px;
    }

    button {
        padding: 10px 15px;
        background-color: #4CAF50;
        color: white;
        border: none;
        border-radius: 4px;
        cursor: pointer;
        font-size: 16px;
    }

    button:hover {
        background-color: #45a049;
    }

    /* Table styles */
    table {
        width: 70%;
        margin: 20px auto;
        border-collapse: collapse;
```

```

    }

    th, td {
        padding: 10px;
        text-align: left;
        border-bottom: 1px solid #ddd;
    }

    th {
        background-color: #f4f4f4;
        font-weight: bold;
    }

    tr:nth-child(even) {
        background-color: #f9f9f9;
    }

    /* List styles */
    ul {
        text-align: left;
        margin: 0;
        padding: 0;
        list-style-type: none;
        display: inline-block;
        max-width: 500px;
        text-align: left;
    }

    li {
        margin: 5px 0;
        padding: 5px;
    }

    /* Footer or message styles */
    p {
        font-size: 16px;
        color: #333;
    }
</style>
</head>
<body>
    <h1>Fractional Knapsack Problem</h1>
    <form method="POST" action="/">
        <label for="profits">Profits (comma-separated):</label>

```

```

        <input type="text" id="profits" name="profits"
placeholder="280,100,120,120" required><br><br>

        <label for="weights">Weights (comma-separated):</label>
        <input type="text" id="weights" name="weights" placeholder="40,10,20,24"
required><br><br>

        <label for="capacity">Max Capacity (W):</label>
        <input type="number" id="capacity" name="capacity" placeholder="60"
required><br><br>

        <button type="submit">Calculate</button>
    </form>

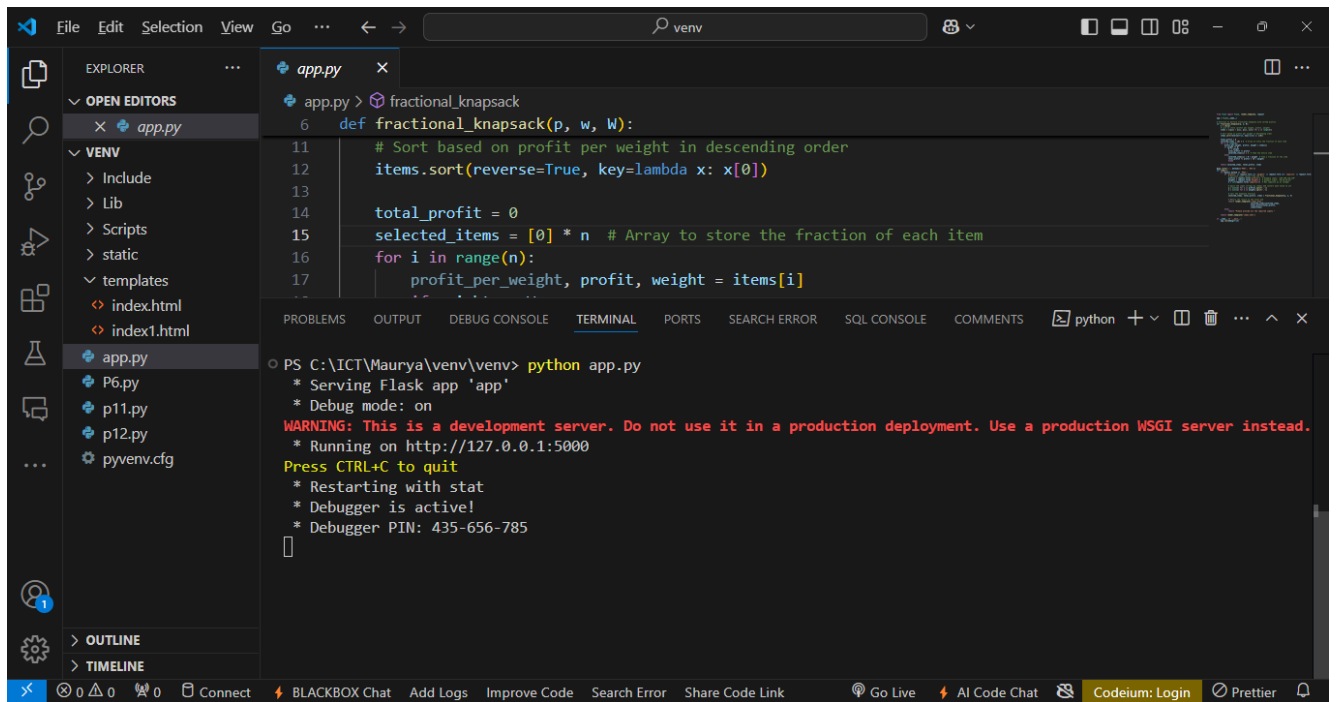
    {% if selected_items %}
    <h2>Results:</h2>
    <p>Total Profit: {{ total_profit }}</p>

    <h3>Items (sorted by profit/weight):</h3>
    <table>
        <tr>
            <th>Profit</th>
            <th>Weight</th>
            <th>Profit/Weight</th>
        </tr>
        {% for item in items %}
        <tr>
            <td>{{ item[1] }}</td>
            <td>{{ item[2] }}</td>
            <td>{{ item[0] }}</td>
        </tr>
        {% endfor %}
    </table>

    <h3>Selected Items (Fractions):</h3>
    <ul>
        {% for fraction in selected_items %}
        <li>{{ fraction }}</li>
        {% endfor %}
    </ul>
    {% endif %}
</body>
</html>

```

Screenshots:



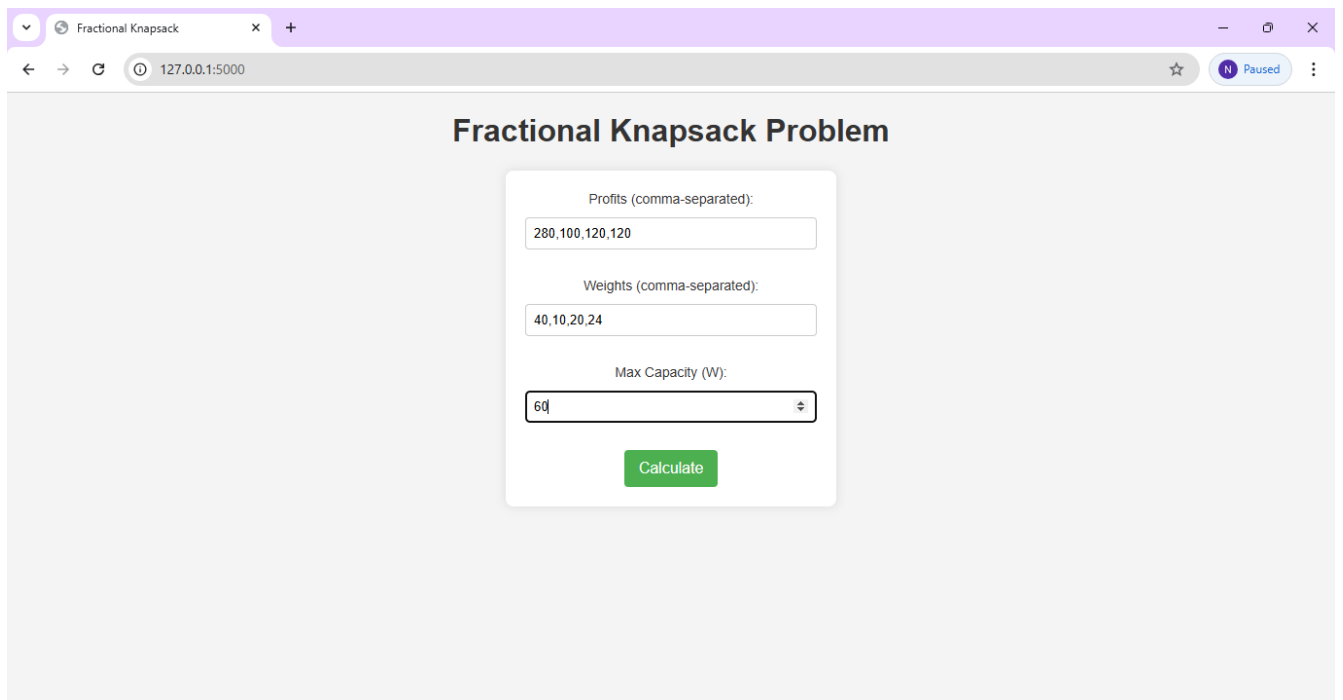
The screenshot shows a VS Code editor with a Python file named `app.py` open. The file contains a function `fractional_knapsack(p, w, W)` that implements the greedy algorithm for the fractional knapsack problem. The function sorts items by profit per weight in descending order and then iterates through them, adding fractions of items until the knapsack is full.

```
6 def fractional_knapsack(p, w, W):
11     # Sort based on profit per weight in descending order
12     items.sort(reverse=True, key=lambda x: x[0])
13
14     total_profit = 0
15     selected_items = [0] * n # Array to store the fraction of each item
16     for i in range(n):
17         profit_per_weight, profit, weight = items[i]
```

The terminal output shows the command `python app.py` being executed. The output indicates that the Flask app is running on `http://127.0.0.1:5000` and provides instructions for debugging.

```
PS C:\ICT\Maurya\venv\venv> python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 435-656-785
```

Output:



The screenshot shows a web browser with the title "Fractional Knapsack Problem". The interface includes three input fields for "Profits (comma-separated)", "Weights (comma-separated)", and "Max Capacity (W)". The "Profits" field contains the value "280,100,120,120", the "Weights" field contains "40,10,20,24", and the "Max Capacity (W)" field contains "60". A green "Calculate" button is located below the input fields.

Profits (comma-separated):
280,100,120,120

Weights (comma-separated):
40,10,20,24

Max Capacity (W):
60

Calculate

Fractional Knapsack

127.0.0.1:5000

Paused

Calculate

Results:

Total Profit: 440.0

Items (sorted by profit/weight):

| Profit | Weight | Profit/Weight |
|--------|--------|---------------|
| 100 | 10 | 10.0 |
| 280 | 40 | 7.0 |
| 120 | 20 | 6.0 |
| 120 | 24 | 5.0 |

Selected Items (Fractions):

1
1
0.5
0