DAA Mini Project

Student Name: Gurjyot Singh
UID: 24MCC20026
Branch: MCA (CCD)
Section/Group: 1/A
Semester: 1
Date of Performance: 04/11/24
Subject Name: DAA Lab
Subject Code: 24CAP 612

Project Title : Budget Optimization System using Knapsack and Greedy Algorithms

Aim/ Overview of the project:

'The Budget Optimization System' is developed to help people improve their personal financial situation through the proper managing of the spending and the prerequisites for it. The suggested approaches that the project uses are such algorithmic methods as the 0/1 Knapsack problem and Greedy algorithms to give effective recommendations concerning the budget.

Objectives:
Expense Tracking: Let users enter and group their regular expenses and the amounts of money which they have to pay for various needs per month.

Budget Optimization:
Implement 0/1 Knapsack algorithm to understand costs and make recommendations on ways to save money, whilst considering necessity facts. To this end, this dynamic programming approach enables users to make correct spending decisions in accordance to the set financial objectives.
Append a Greedy algorithm which coarsely iteratively peels off unnecessary expenses one by one. Based on the priority or flexibility of the expenses, the system gives fast suggestions toward the target saving aim.
Data Visualization: The spending pattern of the user can be depicted with the help of bar charts and pie charts, and help the users to get an idea of how they are spending their money and control over their money spending habits.

User-Friendly Interface: Make sure the system's user interface is friendly and well designed, meaning the users of the system will enter their financial data and view the subsequent suggestion(s).

Significance:
A typical problem with which the project deals is the problem of money management; the use of algorithmic methods provides personalized suggestions. Dynamic programming and applying greedy approaches allow users to attain optimal results regarding their savings and, therefore, improve the Users' financial state and their budgeting skills.

Task to be done

a) **Implement Expense Tracking**:
- Develop a function to categorize and display expenses.

b) **Optimize Budget using 0/1 Knapsack Algorithm**:
- Implement the dynamic programming solution to analyze expenses and suggest adjustments that maximize savings.

c) **Expense Reduction using Greedy Algorithm**:
- Create a function that sorts expenses based on priority and iteratively reduces non-essential costs to meet savings goals.

d) **Data Visualization**:
- Use Matplotlib to generate bar charts and pie charts for visual representation of expenses and savings.

e) **User Interface Development**:
- Design a simple and intuitive command-line or GUI interface for easy user interaction.

Code for Experiment/ Practical

```
import matplotlib.pyplot as plt

user_data = {
    "income": 3000,
    "expenses": {
        "rent": 1000,
        "groceries": 300,
        "utilities": 200,
        "entertainment": 150,
        "transport": 100,
    },
    "savings_goal": 500
}

expense_priorities = {
    "rent": 10,
```

```
    "groceries": 8,
    "utilities": 7,
    "entertainment": 4,
    "transport": 5
}
def optimize_budget(expenses, priorities, savings_goal):
    expense_list = [(k, v, priorities[k]) for k, v in expenses.items()]
    n = len(expense_list)
    dp = [[0] * (savings_goal + 1) for _ in range(n + 1)]

    for i in range(1, n + 1):
        for w in range(savings_goal + 1):
            category, cost, priority = expense_list[i - 1]
            if cost <= w:
                dp[i][w] = max(priority + dp[i - 1][w - cost], dp[i - 1][w])
            else:
                dp[i][w] = dp[i - 1][w]

    w = savings_goal
    adjustments = []
    for i in range(n, 0, -1):
        if dp[i][w] != dp[i - 1][w]:
            category, cost, priority = expense_list[i - 1]
            adjustments.append((category, cost))
            w -= cost
     return adjustments
```

```
def greedy_expense_reduction(expenses, priorities, savings_goal):
    sorted_expenses = sorted(expenses.items(), key=lambda x: priorities[x[0]])
    remaining_goal = savings_goal
    reductions = []
    for category, amount in sorted_expenses:
        if remaining_goal <= 0:
            break
        reducible_amount = min(amount, remaining_goal)
        reductions.append((category, reducible_amount))
        remaining_goal -= reducible_amount
    return reductions
def display_suggestions(adjustments, expenses, savings_goal, method="Knapsack"):
    print(f"\n=== {method} Budget Optimization Suggestions ===")
    total_savings = sum(amount for _, amount in adjustments)

    if total_savings >= savings_goal:
        print(f"To reach your savings goal of ${savings_goal}, consider adjusting these expenses:")
        for category, amount in adjustments:
            print(f"- Reduce {category} by ${amount}")
    else:
        print("Your expenses are too high to meet the savings goal with adjustments alone.")
def visualize_expenses(expenses):
    categories = list(expenses.keys())
    amounts = list(expenses.values())
```

```
    plt.figure(figsize=(10, 5))
    plt.bar(categories, amounts, color='skyblue')
    plt.xlabel('Expense Categories')
    plt.ylabel('Amount Spent ($)')
    plt.title('Monthly Expenses')
    plt.show()

    plt.figure(figsize=(8, 8))
    plt.pie(amounts, labels=categories, autopct='%1.1f%%', startangle=140)
    plt.title('Expense Distribution')
    plt.show()

def main():
    print("=== Personal Finance Management System ===")
    print("Income: $", user_data["income"])
    print("Expenses:")
    for category, amount in user_data["expenses"].items():
        print(f"  {category}: ${amount}")
    print("Savings Goal: $", user_data["savings_goal"])

    knapsack_adjustments = optimize_budget(user_data["expenses"], expense_priorities,
user_data["savings_goal"])
    display_suggestions(knapsack_adjustments, user_data["expenses"], user_data["savings_goal"],
method="Knapsack")
        greedy_adjustments = greedy_expense_reduction(user_data["expenses"], expense_priorities,
user_data["savings_goal"])
```

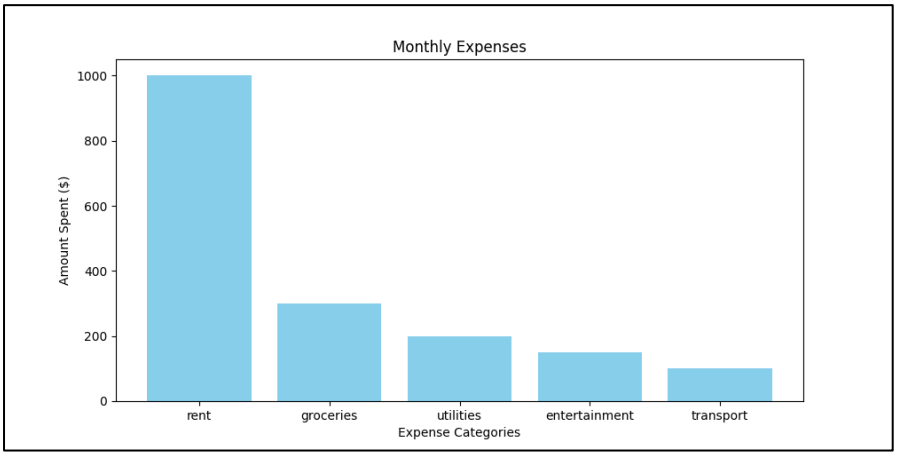
```
    display_suggestions(greedy_adjustments, user_data["expenses"], user_data["savings_goal"],
method="Greedy")
        visualize_expenses(user_data["expenses"])
if __name__ == "__main__":
    main()
```

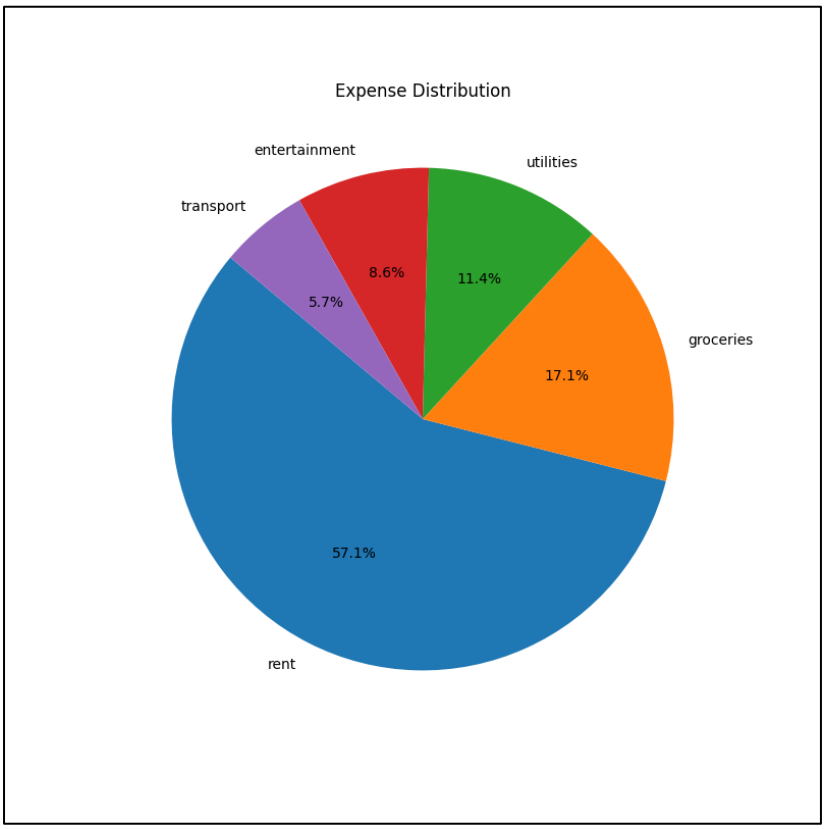
Output

```
[Running] python -u "c:\Users\sgurj\OneDrive\Desktop\DAA Algos\daaproject.py"
=== Personal Finance Management System ===
Income: $ 3000
Expenses:
  rent: $1000
  groceries: $300
  utilities: $200
  entertainment: $150
  transport: $100
Savings Goal: $ 500

=== Knapsack Budget Optimization Suggestions ===
Your expenses are too high to meet the savings goal with adjustments alone.

=== Greedy Budget Optimization Suggestions ===
To reach your savings goal of $500, consider adjusting these expenses:
- Reduce entertainment by $150
- Reduce transport by $100
- Reduce utilities by $200
- Reduce groceries by $50
```





Learning Outcomes

- Understanding of Algorithmic Techniques: The 0/1 Knapsack problem and Greedy algorithms have given me a firm foundation for the algorithms and sharpened my understanding for making it possible to implement the course concepts into the actual financial optimization problem.

- Practical Experience in Dynamic Programming: In knapsack algorithm, because of the handling of dynamic programming, I have been able to gain practical experience in how best one can solve optimization issues through the management of sub-problems.

- Application of Data Visualization: I have also enhanced my knowledge on data visualization by using Matplotlib to produce appropriate charts that will help the user understand spending habits with regards to finances and assist him/her make wise decisions.

- User-Centric Design and Functionality: Design perspective I have understood how the User Experience by setting up a friendly interface for entering the financial data and presenting the Optimization Propositions in a comprehensible method that increases the efficiency of the application.