Week 5 HW: Knapsack Problem Comprehension

Published | Assign To | Edit | ⋮

Objectives:

- Explain algorithms and efficiency
- Understand and use python unit tests
- Understand and use recursion and dynamic programming in python
- Experiment using LLMs to code and test
- Analyze the use of LLMs to code and test

For this homework assignment you will explore solutions to a variation on the classic knapsack problem and the use of LLMs for coding and testing. As you work through each section, answer the questions in the following doc: **HW5 Knapsack Contraints Assignment Template.docx (https://canvas.vt.edu/courses/204793/files/37872809?wrap=1)**

You are allowed to use resources and LLMs but do be sure to document their usage. If you write or use any scripts for your testing then include those in your descriptions.

Think about the following problem description

Given a set of N items, each with a weight and a value. The goal is to select a subset of these items to maximize the total value without exceeding a given capacity W. However, the problem can have additional complexities.

A. Conflicting Items: Certain pairs of items conflict with each other. If you select one item from a conflicting pair, you cannot select the other.

B. Bonuses for Item Sets: Selecting specific combinations of items grants additional bonus value.

Constraints

▾ **Conflicting Items Solution**

A. Conflicting Items

A request to an LLM provided the following solution for the knapsack problem with constraint A listed

above, conflicting items, Write tests to break it and describe if and where it breaks. As needed, search online and use LLMs to learn how to test and solve it.

1)(30pts) Is it correct? See if you can find a test case that breaks it. If so, explain how you found it and include a screenshot of it failing in vscode. (50-100 words). *Note that the test case should demonstrate a logical error (not handling cases such as negative numbers or empty input).*

```
def knapsack_with_conflicts(weights, values, conflicts, W):
    n = len(weights)

    # Initialize a DP table
    dp = [[0] * (W + 1) for _ in range(n + 1)]

    # Fill the DP table
    for i in range(1, n + 1):
        for w in range(W + 1):
            # Check if the current item conflicts with any previously selected item
            conflict = False
            for j in range(i):
                if (i - 1, j) in conflicts or (j, i - 1) in conflicts:
                    conflict = True
                    break

            if conflict:
                dp[i][w] = dp[i - 1][w]
            else:
                if weights[i - 1] <= w:
                    dp[i][w] = max(dp[i - 1][w], dp[i - 1][w - weights[i - 1]] + values[i - 1])
                else:
                    dp[i][w] = dp[i - 1][w]

    return dp[n][W]

# Example usage
weights = [2, 3, 4, 5]
values = [3, 4, 5, 6]
conflicts = [(0, 2), (1, 3)] # Indices of conflicting items
W = 5

result = knapsack_with_conflicts(weights, values, conflicts, W)
print(result)
```

Bonus Items Solution

B. Bonuses for Items Set

A request to an LLM provided the following solution for the knapsack problem with constraint B listed above, bonus value for including items from a set. Write tests to break it and describe if and where it breaks. As needed, search online and use LLMs to learn how to test and solve it.

2)(30pts) Is it correct? See if you can find a test case that breaks it. If so, explain how you found it and include a screenshot of it failing in vscode. (50 - 100 words). *Note that the test case should demonstrate a logical error (not handling cases such as negative numbers or empty input).*

```
def knapsack_with_bonuses(weights, values, W, bonuses):
    n = len(weights)

    # Initialize a DP table
    dp = [[0] * (W + 1) for _ in range(n + 1)]

    # Fill the DP table
    for i in range(1, n + 1):
        for w in range(W + 1):
            if weights[i - 1] <= w:
                dp[i][w] = max(dp[i - 1][w], dp[i - 1][w - weights[i - 1]] + values[i - 1])
            else:
                dp[i][w] = dp[i - 1][w]

    # Apply bonuses
    for items, bonus in bonuses:
        total_weight = sum(weights[i - 1] for i in items)
        total_value = sum(values[i - 1] for i in items) + bonus

        for w in range(W, total_weight - 1, -1):
            dp[n][w] = max(dp[n][w], dp[n][w - total_weight] + total_value)

    return dp[n][W]

# Example usage
weights = [2, 3, 4, 5]
values = [3, 4, 5, 6]
W = 10
bonuses = [([1, 4], 5)] # Bonus for selecting items with indices 1 and 4

result = knapsack_with_bonuses(weights, values, W, bonuses)
print(result)
```

Reading

Reading and Reflection

Read the following essay by Joel Chan for University of Maryland's Introduction to Programming for Information Science: **How (Not) to Code with LLMs as a Beginning Programmer (https://canvas.vt.edu/courses/204793/pages/how-not-to-code-with-llms-as-a-beginning-programmer)**

Answer the following questions based on this assignment, previous assignments in the course, or other experiences that you describe.

3) (10pts)((Describe a time when an LLM provided code that you did not understand. What were your next steps? How does this relate to Joel Chan's essay? (50-100 words)

4) (10pts) Describe a time when an LLM provided code that did not work. How did you realize it didn't work? How does this relate to Joel Chan's essay? (50-100 words)

5) (10pts) Describe the concept of likely vs correct code. Provide at least one example. How does this relate to Joel Chan's essay? (50-100 words)

6) (10pts) Describe the concept of steering. What are some examples of steering an LLM effectively to generate or fix code? What must be true fo the LLM user to steer effectively?(50-100 words)

Points	100
Submitting	a file upload
File Types	pdf

Due	For	Available from	Until
Feb 21	CS_2104_13319_202501	-	Feb 22 at 11:59pm
Feb 21	CS_2104_13317_202501	-	Feb 22 at 11:59pm
Feb 21	CS_2104_13318_202501	-	Feb 22 at 11:59pm
Feb 21	CS_2104_13316_202501	-	Feb 22 at 11:59pm
Feb 23 at 1am	1 student	-	Feb 23 at 1am

Week 5 HW: Knapsack Problem Comprehension Rubric

You've already rated students with this rubric. Any major changes could affect their assessment results.

Criteria	Ratings	Pts

1a correct or failed test case	**10 pts** **Full Marks** Response correlates with screenshot and behavior checks out when run manually.		**0 pts** **No Marks**	10 pts
1b describe process	**10 pts** **Full Marks**	**5 pts** **Partical Marks** Somewhat describe process	**0 pts** **No Marks**	10 pts
1c screenshot of failing code	**10 pts** **Full Marks**		**0 pts** **No Marks**	10 pts
2a correct or failed test case	**10 pts** **Full Marks** Response correlates with screenshot and behavior checks out when run manually.		**0 pts** **No Marks**	10 pts
2b describe process	**10 pts** **Full Marks** Descriptions are thorough and are logical	**5 pts** **Partical Marks** Somewhat describe process	**0 pts** **No Marks** No descriptions or fail to describe process at any level.	10 pts
2c screenshot of failing code	**10 pts** **Full Marks**		**0 pts** **No Marks**	10 pts

3 Description of LLM code didn't understand	**10 pts** **Full Marks** Provide a concrete example, Describe next steps, Connect back to Joel Chan's Essay	**7 pts** **Partial Marks** Missing 1/3 parts: Provide a concrete example, Describe next steps, Connect back to Joel Chan's Essay	**3 pts** **Minimal Marks** Missing 2/3 parts: Provide a concrete example, Describe next steps, Connect back to Joel Chan's Essay	**0 pts** **No Marks**	10 pts
4 Description of LLM code that didn't work	**10 pts** **Full Marks** Provide a concrete example, How did you realize, Connect back to Joel Chan's Essay	**7 pts** **Partial Marks** Missing 1/3 parts: Provide a concrete example, How did you realize, Connect back to Joel Chan's Essay	**3 pts** **Minimal Marks** Missing 2/3 parts: Provide a concrete example, How did you realize, Connect back to Joel Chan's Essay	**0 pts** **No Marks**	10 pts
5 Describe likely vs correct code	**10 pts** **Full Marks** Describe the concept, Provide one example, relate back to Joel Chan's essay	**7 pts** **Partial Marks** 2/3 correct: Describe the concept, Provide one example, relate back to Joel Chan's essay	**3 pts** **Minimal Marks** 1/3 correct :Describe the concept, Provide one example, relate back to Joel Chan's essay	**0 pts** **No Marks**	10 pts
6 Describe the concept of steering	**10 pts** **Full Marks** Describe the concept, Provide more than one example, describe what must be true to steer effectively	**7 pts** **Partial Marks** 2/3 correct: Describe the concept, Provide more than one example, describe what must be true to steer effectively	**3 pts** **Minimal Marks** 1/3 correct :Describe the concept, Provide more than one example, describe what must be true to steer effectively	**0 pts** **No Marks**	10 pts

Total Points: 100			