

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 using namespace std;
5
6 vector<vector<int>> getKnapsackItems(vector<vector<int>> knapsackValues,
7                                   vector<vector<int>> items, int weight) {
8
9     // O(nc) time | O(nc) space
10    vector<vector<int>> knapsackProblem(vector<vector<int>> items, int capacity,
11                                       vector<vector<int>> knapsackValues(items.size() + 1,
12                                       vector<int>(capacity + 1, 0));
13
14    for (int i = 1; i < items.size() + 1; i++) {
15        int currentWeight = items[i - 1][1];
16        int currentValue = items[i - 1][0];
17        for (int c = 0; c < capacity + 1; c++) {
18            if (currentWeight > c) {
19                knapsackValues[i][c] = knapsackValues[i - 1][c];
20            } else {
21                knapsackValues[i][c] =
22                    max(knapsackValues[i - 1][c],
23                      knapsackValues[i - 1][c - currentWeight] + currentValue);
24            }
25        }
26    }
27
28    return getKnapsackItems(knapsackValues, items,
29                             knapsackValues[items.size()][capacity]);
30 }
31
32 vector<vector<int>> getKnapsackItems(vector<vector<int>> knapsackValues,
33                                   vector<vector<int>> items, int weight) {
34     vector<vector<int>> solution = {{}, {}};
35     int i = knapsackValues.size() - 1;
36
37     while (i > 0) {
38         int currentWeight = items[i - 1][1];
39         int currentValue = items[i - 1][0];
40         if (knapsackValues[i][currentWeight] == knapsackValues[i - 1][currentWeight]) {
41             continue;
42         }
43         solution[i] = {currentValue, currentWeight};
44         i--;
45     }
46
47     return solution;
48 }
```

Solution 1

Solution 2

Solution 3

```
1 #include <vector>
2 using namespace std;
3
4 vector<vector<int>> knapsackProblem(vector<vector<int>> items, int capacity) {
5     // Write your code here.
6     // Replace return below.
7     return {
8         {10}, // total value
9         {1, 2}, // item indices
10    };
11 }
```

Our Tests

Custom Output

Submit Code

1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.

2

3 #include <vector>

4 using namespace std;

5

6 vector<vector<int>> getKnapsackItems(vector<vector<int>> knapsackValues,

7 vector<vector<int>> items, int weight) {

8

9 // O(nc) time | O(nc) space

10 vector<vector<int>> knapsackProblem(vector<vector<int>> items, int capacity,

11 vector<vector<int>> knapsackValues(items.size() + 1,

12 vector<int>(capacity + 1, 0));

13

14 for (int i = 1; i < items.size() + 1; i++) {

15 int currentWeight = items[i - 1][1];

16 int currentValue = items[i - 1][0];

17 for (int c = 0; c < capacity + 1; c++) {

18 if (currentWeight > c) {

19 knapsackValues[i][c] = knapsackValues[i - 1][c];

20 } else {

21 knapsackValues[i][c] =

22 max(knapsackValues[i - 1][c],

23 knapsackValues[i - 1][c - currentWeight] + currentValue);

24 }

25 }

26 }

27

28 return getKnapsackItems(knapsackValues, items,

29 knapsackValues[items.size()][capacity]);

30 }

31

32 vector<vector<int>> getKnapsackItems(vector<vector<int>> knapsackValues,

33 vector<vector<int>> items, int weight) {

34 vector<vector<int>> solution = {{}, {}};

35 int i = knapsackValues.size() - 1;

36

37 while (i > 0) {

38 int currentWeight = items[i - 1][1];

39 int currentValue = items[i - 1][0];

40 if (knapsackValues[i][currentWeight] == knapsackValues[i - 1][currentWeight]) {

41 continue;

42 }

43 solution[i] = {currentValue, currentWeight};

44 i--;

45 }

46

47 return solution;

48 }

Custom Output

Submit Code

Run or submit code when you're ready.