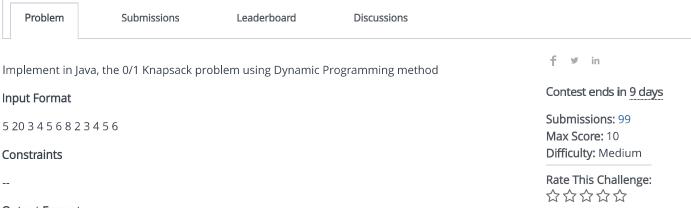
More



Knapsack-Dynamic_programming



Output Format

Optimal Solution: 26 The objects picked up into knapsack are: 5 4 3 2 1

Sample Input 0

Sample Output 0

```
Optimal Solution: 26
The objects picked up into knapsack are:
5 4 3 2 1
```

```
Java 7
                                                                                                          *
1 ▼import java.util.Scanner;
2 ▼class DKnapsack {
3
        int c,n,p[],w[],v[][];
        public DKnapsack(int n,int c,int[] p,int[] w)
5
6
            super();
7
            this.n=n;
8
            this.c=c;
9
            this.w=w;
10
            this.p=p;
            this.v=new int[n+1][c+1];
11 🔻
12
13
        void compute()
14 ▼
15 ▼
            for(int i=0;i<=n;++i){</pre>
                 for(int j=0;j<=c;++j){
16 ▼
17 ▼
                     if(i==0||j==0){
```

```
18
                         v[i][j]=0;
19
                     }
20 🔻
                     else if(j-w[i] >= 0)
21 🔻
                         v[i][j]=max(v[i-1][j],p[i]+v[i-1][j-w[i]]);
22 🔻
23
                     }
                     else if(j-w[i]<0)</pre>
24 ▼
25 ▼
                     {
                         v[i][j]=v[i-1][j];
26 🔻
27
                     }
28
                 }
29
30 🔻
            System.out.println("Optimal Solution: "+v[n][c]);
31
            traceback();
32
33 🔻
        void traceback(){
            System.out.println("The objects picked up into knapsack are:");
34
35
            int i=n;
36
            int j=c;
            while(i>0)
37
38 ▼
            {
39
                 if(v[i][j]!=v[i-1][j])
40
                     System.out.print(i+" ");
41
42 •
                     j=j-w[i];
43
                     i--;
44
                 }
45
                 else {
46
                     i--;
47
                 }
            }
48
49
        }
50
        private int max(int i,int j){
51
            if(i>j)return i;
52
            else return j;
53
        }
54 }
55 ▼public class KpDynamic{
        public static void main(String[] args){
56 ₹
57
            int c,n;
            Scanner input=new Scanner(System.in);
58
            //System.out.println("Enter number of objects");
59
60
            n=input.nextInt();
61 ▼
            int[] p=new int[n+1];
62
            int[] w=new int[n+1];
63
            int i;
            //System.out.println("Enter capacity of Knapsack");
64
65
            c=input.nextInt();
            //System.out.println("Enter profit for each "+n+" objects");
66
67
            for(i=1;i<=n;i++)
68 ▼
                 p[i]=input.nextInt();
            //System.out.println("Enter weight for each "+n+" objects");
69
70
            for(i=1;i<=n;i++)
71
                 w[i]=input.nextInt();
72
            DKnapsack dk=new DKnapsack(n,c,p,w);
73
            dk.compute();
        }
74
75
   }
76
                                                                                                  Line: 1 Col: 1
```

<u>**1**</u> <u>Upload Code as File</u> ☐ Test against custom input

Run Code

Submit Code

Testcase 0 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin) 5 20 3 4 5 6 8 2 3 5 6 Your Output (stdout) Optimal Solution: 26 The objects picked up into knapsack are: 5 4 3 2 1 **Expected Output** Optimal Solution: 26 The objects picked up into knapsack are: 5 4 3 2 1

Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy |