Assignment 6: 0-1 Knapsack Problem

Implement the Knapsack problem shown in the class.

Assignment deliverables are two items
1. ACM Problem: (25%)
http://uva.onlinejudge.org/index.php?option=onlinejudge&page=show_problem&problem=931
2. Class Assignment (75%)

Direction

****Please make sure the Class Name, Method Names and the parameter Sequence are correct in your code in order to pass test cases.**

Create a class named as **`KnapSackDP`** which will contain two methods named **`knapSack`** and **`findItems`**.
Constructor of **`KnapSackDP`** will take two integer array as input.
1. int [ ] value : this will contain benefit of the given elements.
2. int [ ] weight: this will contain weight of given elements.
So the constructor will look as follows

```
public KnapSackDP(int [] value,int [] weight)
{

}
```

Note that length of value and weight array will be same. This will be passed from Driver Class.

The method `knapSack` will take two integer numbers.
1. int n: number of elements
2. int W: maximum weights that can be carried.

so the method will look as follows

```
public int knapSack(int n,int W)
{

}
```

This method will return the the maximum benefit value of KnapSack.
Another method named `findItems` that will return an Array list containing the list of elements chosen for the knapSack. So the method will look like as follows.

```
public ArrayList<Integer> findItems()
{

}
```

Way to solutions

Follow the class lectures and resources listed in the Reading Materials section at the end of the document.

Sample Test Data

Test Case 1:

```
Input:
Weights:
517 570 354 846 634 986 751 209 477 790 139 681 891 36 761 45 560 482 748
663
Benefits:
893 248 613 139 370 94 515 644 180 221 382 312 629 372 447 149 843 486 149
443
Number of Elements: 19
Knapsack Capacity: 3713

Output:
Item  Profit      Weight
0     893         517
1     248         570
2     613         354
3     139         846
4     370         634
5     94          986
6     515         751
7     644         209
8     180         477
9     221         790
10    382         139
11    312         681
12    629         891
13    372         36
14    447         761
15    149         45
16    843         560
17    486         482
18    149         748
19    443         663
Total Benefit: 4633
Items in knapsack: [2, 6, 7, 10, 12, 13, 15, 16, 17]
```

Test Case2:

```
 Input:
Weights:
749 904 950 433 230 452 847 348 12 357 335
Benefits:
2 607 832 433 515 298 953 794 280 851 211
Number of Elements: 10
Knapsack Capacity: 1872

Output:
Item  Profit      Weight
0     2           749
1     607         904
2     832         950
3     433         433
4     515         230
5     298         452
6     953         847
7     794         348
8     280         12
9     851         357
10    211         335
Total Benefit: 3393
Items in knapsack: [4, 6, 7, 8, 9]
```

Test Case 3:

```
Input:
Weights:
400 301 331 969 726 508 965 936 58 139 429 52 177 572 684
Benefits:
168 133 145 806 914 420 123 501 988 688 70 65 812 613 565 389
Number of Elements: 15
Knapsack Capacity: 2670

Output:
Weight and Benefit array sizes don’t match
```

Evaluation

Submission will be evaluated using following criteria (with distribution of marks)

Criteria	Marks
Created necessary java files (according to directions)	10
Implementation of knapSack Method	30
Understanding the bottom up concept	30
Implementation of findItems	30
Total	**100**

For Incorrect Implementations:

Implementation of knapSack method*	0-25
Understanding the bottom up concept *	0-20
Implementation of findItems*	0-25

* Marking depends on evaluator.

***Notes on copying other's code**: If anybody found copying solution code from other student(s), all of them will be penalized. Penalty includes*

- ➢ ***Assigning ZERO as mark for the solution submitted***
- ➢ ***Assigning ZERO as mark for the best submission among all other submissions. (assigned at the end of semester)***

Copying solutions from the Internet will also incur similar penalties.

Submission & Contact

Submission Type	Individual
Submission Deadline (Full Credit):	June 1, 2014

Students are encouraged to use Google Group for contacting to resolve their issues (instead of personal mails.)

Group Address: https://groups.google.com/forum/#!forum/algorithmfiesta2014

Reading Materials

Introduction to Algorithm by Thomas Cormen
Slides and pdf in TSR/CSE/RIK/CSE221