Programming Design, Spring 2013
Homework 05

Instructor: Ling-Chieh Kung
Department of Information Management
National Taiwan University

To submit your work, please upload the following two files to the online grading system PDOGS at http://stella.im.ntu.edu.tw/online-judgement/.

1. Your source file (the .cpp file) for Problem 1. This should be submitted to the PD004 section.

NO hard copy and NO late submission. The due time of this homework is ***1:00pm, March 25, 2013***.

Problem 1

(100 points) Please write a C++ program according to the following instructions.

What should your program do

The input contains several lines of numbers. In each line, an instance of the knapsack decision problem introduced in class is defined. An instance is defined with:

1. A positive integer number B: the knapsack capacity,
2. A positive integer n: the number of items, and
3. A sequence of n integer numbers w_1, w_2, ..., w_n: the weights of items.[1]

The parameters for an instance will be given in a line in the following way: First B, then a space, then n, then a space, the the sequence of w_1, w_2, ..., w_n separated by spaces. For example, the following input

```
11 4 2 3 5 6
```

describes a knapsack instance that the knapsack capacity is 11, the number of items is 4, and the weights of these items are 2, 3, 5, and 6, respectively. You may assume that the weights are all different and presented in an ascending order. Each line of the testing data represents a problem instance. Please also assume that $B \leq 1000$ and $n \leq 50$ so that you have a predetermined bounds for initializing your array.

Your program should solve these knapsack instance following the algorithm described in class. After solving the problem, you need to display

1. Whether there is a subset of items that can exactly fill the knapsack and
2. If yes, which items should be included in the subset.

You should first output an integer for the first question, where `0` means no and `1` means yes. In the former case, the output ends and then you complete this line by placing a newline object/character at the end of this line. In the latter case, however, you need to display which items to select by listing their indices from the largest one to the smallest one, separated by spaces.

[1] If the capacity of a weight is fractional, we may always makes it an integer by multiplying a suitable integer value to all problem parameters as long as the fractional number is rational. Therefore, the assumption that all parameters are integers is without loss of generality for practical problems.

For an instance, it is possible that there are multiple ways to fill the knapsack. For example, for the instance given above, one way is to include the first, second, and fourth items and another way is to include the third and the fourth items. Therefore, when you run the algorithm and want to mark one subproblem as yes, mark it as the item should be selected even if the item does not need to be selected. With this in mind, for the instance given above the items to be included will be the third and fourth items. The line of output is thus

```
1 4 3
```

The first number 1 indicates that the knapsack can be filled exactly, and the remaining numbers indicate which items should be included.

Below are some more examples:

- Input: `16 4 2 3 5 6`. Output: `1 4 3 2 1`.
- Input: `15 4 2 3 5 6`. Output: `0`.
- Input: `100 5 10 20 30 40 50`. Output: `1 5 4 1`.

Grading criteria

- 70% of your grades for this program will be based on the correctness of your output. The online grading system will input 35 sets of testing data and then check your outputs. You may only see the grades of running your program on these data but cannot see the inputs and outputs. These 35 sets count for 70 points, i.e., 2 points for each set.
- 30% of your grades for this program will be based on how you write your program, including the logic and format. Please try to write a robust, efficient, and easy-to-read program.