CSE2011 Problem Solving, 2016 Spring

Instructor: Jinkyu Lee

Homework 4 (80 points): 2015/4/28

Due date: 2016/5/9 23:59 (submission to icampus)

Problem 4-1: zero-sum

- For given one-dimensional array, develop an efficient way to find all consecutive sections, each of whose sum is zero.
- Example: 1 -2 3 -4 5 -3 2 1
- Then, the output is (1,6), (4,7), (6,8).

In your code

- No code

In your report

- Explain your solution idea (algorithm).
- Show another example how your idea (algorithm) works.

Problem 4-2: Shoemaker's problem (page 94)

- A shoemaker has N orders from customers which he must satisfy. The shoemaker can work on only one job in each day, and jobs usually take several days. For the ith job, the integer Ti (1 ≤ Ti ≤ 1, 000) denotes the number of days it takes the shoemaker to finish the job.

- But popularity has its price. For each day of delay before starting to work on the ith job, the shoemaker has agreed to pay a fine of Si (1 ≤ Si ≤ 10, 000) cents per day. Help the shoemaker by writing a program to find the sequence of jobs with minimum total fine.

Input

- The input begins with a single positive integer on a line by itself indicating the number of the test cases, followed by a blank line. There is also a blank line between two consecutive cases.

- The first line of each case contains an integer reporting the number of jobs N, where 1 ≤ N ≤ 1000. The ith subsequent line contains the completion time Ti and daily penalty Si for the ith job.

Output

- For each test case, your program should print the sequence of jobs with minimal fine. Each job should be represented by its position in the input. All integers should be placed on only one output line and each pair separated by one space. If multiple solutions are possible, print the first one in lexicographic order.

- The output of two consecutive cases must be separated by a blank line.

Sample input

1

```
4
3 4
1 1000
2 2
5 5
```

Sample output

```
2 1 3 4
```

In your code

- Implement a C program for Homework 4-2, but you should use template_HW4-2.c using the backtracking technique.
- You can edit the body of five functions: is_a_solution, process_solution, construct_cadidates, calculate_fine and backtrack functions. You cannot modify any other parts than the body of the five functions; for example, no additional global variable, no additional function, no parameter change, and no edit for main function.
- TA will copy your code "from here" "to here" to the template code; other parts will not be evaluated.
- Insert comments to contents you created or modified.
- TA will test your program in http://ideone.com/

In your report

- Explain your solution idea (algorithm) and code.