Programming Design, Spring 2013
Homework 08

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. A CPP file for Problem 1 (to the PD006 section).
2. A CPP file for Problem 2 (to the PD007 section).

NO hard copy and NO late submission. The due time of this homework is ***1:00pm, April 22, 2013***.

Problem 1

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

What should your program do

The input contains several lines of integers. In each line, there are $n+2$ values labeled as n, x_1, x_2, ..., x_n, and t. The first value, n, tells you there are $n+2$ values in this line. The next n values, $x_1, x_2, ..., x_n$, form the search space. You need to determine whether the last ($(n+2)$th) value, t, exists in the search space. The values forming the search space will be nonrepeating and sorted in the ascending order (i.e., $x_1 < x_2 < \cdots < x_n$). You are required to implement a binary search algorithm so that for each line of input, your program will output according to the following rules:[1]

1. If t does not exist in the search space (i.e., $t \neq x_i$ for all $i = 1, ..., n$), output `0` and then a newline character or object.
2. If t exist in the search space, output the position of the value in the search space. More precisely, if $t = x_k$, output k and then a newline character or object.

For example, the following input

```
7 2 4 7 9 12 19 30 12
```

contains $n + 2 = 9$ values. The first value 7 means there are seven values forming the search space. The next $n = 7$ values (2, 4, ..., and 30), which form the search space, are nonrepeating and sorted in the ascending order. The task is to check whether the last value 12 exists in the search space. As it does, the output should be 5 (followed by a newline character or object), because 12 is equal to the fifth value in the search space.

Below are some more examples:

- Input: `6 1 2 3 4 5 6 7`. Output: `0`.
- Input: `6 2 6 8 41 260 976 6`. Output: `2`.
- Input: `1 1 1`. Output: `1`.

Please note that there is no restriction on n, so the number of values given to you in each line may be unreasonably large (however, we promise that n can be stored in an `int`-type varaible). Therefore, you need to use dynamic memory allocation to store input values.

[1] You are allowed to modify the example program "PDSp13_hw08_binarySearch", which is exactly the same as the one discussed in class.

Grading criteria

First of all, the TAs will open your CPP file and check whether you implement the binary search algorithm. If not, you will get zero point with no exception (i.e., no matter how many points you get through the online grading system). If yes, your grades will be determined in the following way:

- 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.
- 20% 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.
- 10% of your grades will be given to you if you correctly apply dynamic memory allocation.

Problem 2

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

What should your program do

The input contains several lines of integers. In each line, there are $n+2$ values labeled as n, x_1, x_2, ..., x_n, and t. The first value, n, tells you there are $n+2$ values in this line. The next n values, $x_1, x_2, ..., x_n$, form the search space. You need to determine whether the last ($(n+2)$th) value, t, exists in the search space. The n values forming the search space may be repeating. Moreover, they are unsorted. You are required to implement a search algorithm so that for each line of input, your program will output according to the following rules:

1. If t does not exist in the search space (i.e., $t \neq x_i$ for all $i = 1, ..., n$), output `0` and then a newline character or object.
2. If t exist in the search space, output ALL the positions of the value in the search space, separated by white spaces, in the ascending order. More precisely, if $t = x_{k_1} = x_{k_2} = \cdots = x_{k_m}$, output k_1, k_2, ..., k_m, and then a newline character or object.

For example, the following input

```
7 2 12 7 19 12 19 30 12
```

contains $n+2=9$ values. The first value 7 means there are seven values forming the search space. The next $n=7$ values (2, 4, ..., and 30), which form the search space, are repeating and unsorted. The task is to check whether the last value 12 exists in the search space. As it does and it appears twice, the output should be

```
2 5
```

(followed by a newline character or object), because 12 is equal to the second and fifth value in the search space. Again, there is no restriction on n, so you need to use dynamic memory allocation to store input values.

Below are some more examples:

- Input: `6 1 2 1 4 1 6 3`. Output: `0`.
- Input: `6 122 16 8 41 20 42 16`. Output: `2`.
- Input: `8 1 1 1 1 4 2 3 1 1`. Output: `1 2 3 4 8`.

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.
- 20% 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.
- 10% of your grades will be given to you if you correctly apply dynamic memory allocation.