

CSE 2320-003 Lab Assignment 1

Due February 12

Goals:

1. Understanding of binary search.
2. Understanding of indirection, maps/permutations, and swapping.

Requirements:

1. Write a C program to maintain n counters indexed by $0 \dots n-1$. n will be the first input value and all counters are initially valued as zero. The following operations will then appear, one per line, in the input:
 - a. 0 - terminate execution.
 - b. 1 - print the counters in ascending index value order as (`index`, `count`) pairs. ($O(n)$ time)
 - c. 2 - print the counters in ascending counter value order as (`index`, `count`) pairs. ($O(n)$ time)
 - d. 3 i - add one to the counter indexed by i . ($O(\log n)$ time)
 - e. 4 i - subtract one from the counter indexed by i . ($O(\log n)$ time)
 - f. 5 i j - determine the number of counters whose values are no smaller than i and no larger than j . ($O(\log n)$ time)

The input will be read from standard input (`stdin`) as either keyboard typing or as a shell redirect (`<`) from a file. Prompts/menus are completely unnecessary!

2. Submit your C program on Canvas by 12:45 pm on February 12. Comments at the beginning of the source file should include: your name, your ID number, and the command used to compile your code on Omega (5 point penalty for non-compliance).

Getting Started:

1. Review binary search and obtain a copy of <http://ranger.uta.edu/~weems/NOTES2320/binarySearchRange.c>. Code similar to this will be useful in implementing operations 3, 4, and 5.
2. Your program should dynamically allocate three tables - `map`, `index`, and `count`. (If you wish, the last two tables may be implemented as an array of structs.) `index[i]` indicates which of the n counters has its value presently stored as `count[i]`. `map[i]` is used to find the counter with index i , i.e. it is always true that `index[map[i]]==i`.

Operation 2 may be coded as:

```
for (i=0;i<n;i++)
    printf("%d %d\n",index[i],count[i]);
```

Operation 1 may be coded as:

```
for (i=0;i<n;i++)
    printf("%d %d\n",i,count[map[i]]);
```

3. You should implement and completely debug operation 3 before implementing operation 4.
4. Your code must satisfy the indicated time bounds by using binary search when possible. DO NOT USE SORTING!