Computer Science I Program #3: Winning the Lottery (Linked Lists)

Please check Webcourses for the Due Date

Read all the pages before starting to write your code

The state of Florida has come up with a new way to determine lottery winners. The system is outlined as follows:

Phase 1

Each person eligible to win is placed in one of ***G*** groups. For each group, a process is used to narrow down the possible number of winners. The process is as follows:

Let a group initially start with ***p*** people. We can number the people within a group from 1 to ***p*** and assume they are arranged in a circle, with 1, then 2, …, then p, followed by 1. Each group has its own skip number, ***s***, and its own threshold number, ***t***, where $t < p$. To eliminate some people in a group from possibility of winning the lottery, start at the person labeled 1 and skip over ***s*** people in line. Then, remove the following person. (Thus the first person removed is always person number ***s***+1.) Then repeat the process. Since the line is circular, once you pass the highest numbered person left in line, you'll continue to the lowest numbered person left in line. Continue eliminating people in this fashion until there are precisely ***t*** people left.

Phase 2

Of all of the people left, the winner of the lottery will be the lowest numbered person from any group. If there are multiple people from different groups with the same lowest number, then the person from the lowest group number is chosen as the lottery winner.

The state started forcing people to come out and form lines doing the process as described, but this was extremely time consuming and they got many complaints from the losers who were forced to stand out in a big hot field for many hours before they got eliminated (some of those officials in Tallahassee aren't fast counting around circles…)

Now, the state has decided to hire you to simulate the process and determine who the winner will be without actually forcing all participants from standing around in circles! Don't let the state officials down!

The Problem

Given a list of the number of groups, the initial sizes of each group, the skip value for each group and the threshold value for each group, determine the winner of the lottery (group number and person number). In addition to determining the lottery winner, you must print a log for each group of which people get eliminated (in the order that they are eliminated from contention of winning the lottery.)

The Input (to be read from standard input)
The first line will contain a single positive integer, n ($n \leq 25$), representing the number of input cases to process.

The first line of each input case will contain a single positive integer G ($G \leq 10$), representing the number of groups for the lottery selection.

The following G lines will each contain information about each of the G groups. The i^{th} of these lines will have information about group number i ($1 \leq i \leq G$). The i^{th} of these lines will have three space separated integers: p_i ($2 \leq p_i \leq 10^5$), s_i ($0 \leq s_i < p_i$) and t_i, ($1 \leq t_i < p_i$), representing the number of people in group i, the skip number for group i, and the threshold for group i, the number of people who will remain for contention to win the lottery after Phase 1 finishes for that group.

The sum of $s_i(p_i - t_i)$ over all groups will be 10^7 or less. (This means that over all groups, as you simulate Phase 1, you won't have to move forward from node to node more than 10^7 times.)

The Output (to be printed to standard out)
For each group, print the following header on a line by itself:

```
Group #x:
```

where x is the 1-based number of the group.

Follow this with $p_i - t_i$ integers, one per line, representing the people eliminated from contention of winning the lottery in the order in which they were eliminated.

Print the information for each group in order by group number. Don't put any blank lines between the output for different groups.

Follow this with a single line in the following format:

```
Lottery winner is person k from group g.
```

where k is the number of the person (1-based) and g is the group number (also 1-based) of the person who was selected to win the lottery.

The total output for all cases won't exceed 10^6 lines.

Sample Input

```
1
5
10 2 2
8 1 1
7 2 3
5 1 3
9 1 4
```

Sample Output

```
Group #1:
3
6
9
2
7
1
8
5
Group #2:
2
4
6
8
3
7
5
Group #3:
3
6
2
7
Group# 4:
2
4
Group #5:
2
4
6
8
1
Lottery winner is person 1 from group 2.
```

Implementation Restrictions/ Run-Time/Memory Restrictions

1. Although there is a much faster way to get the final answer, ***you must implement a linked list of integers*** to simulate the elimination process for each group. You must actually "walk" around the linked list, going node by node, to do the eliminations.

2. You must create your own node struct. It should be similar to the one covered in class.

3. You must write standard linked list functions (insert, delete) that operate on arbitrary circular linked lists. The delete may have to be written a bit differently than what was shown in class because you won't know in advance what value to delete, but you will be able to obtain a pointer to the node to delete (and/or the node before the one to delete).

4. Your code must be modular. For each group, the same set of functions should be called, and these functions should in turn call the same linked list functions.

5. You must free all dynamically allocated memory at the appropriate time during execution. (Memory for nodes representing all people eliminated during phase 1 should be freed as soon as it's determined that person is eliminated. Memory for nodes representing people making it past Phase 1 should be freed at the very end of the processing of each case.)

Deliverables

You must submit four files over WebCourses:

1) A source file, *lottery.c*. Please use stdin, stdout. There will be an automatic 10% deduction if you read input from a file or write output to a file.

2) A file describing your testing strategy, *lastname_Testing.doc(x) or lastname_Testing.pdf*. This document discusses your strategy to create test cases to ensure that your program is working correctly. If you used code to create your test cases, just describe at a high level, what your code does, no need to include it.

3) Files *lottery.in* and *lottery.out*, storing both the test cases you created AND the corresponding answers, respectively.