CSO-102 : Data Structures – Tasks 1 & 2.

Sheldon Cooper is back with a brand new podcast of "**Fun with Flags**".

He records the podcast for N minutes. During each minute Sheldon talks about the flag of any one country. Because of his habit of undermining others' intelligence in comparison to his, he thinks that it will be difficult for his viewers to remember the names of the countries. Hence, he decides to replace each country name by a unique positive integer called the **index** of the country. For example, India (1), Brazil (6), Mexico (3) is valid whereas India (1), Brazil (1), Mexico (3) is not valid because India and Brazil are both represented by the same index 1.

Each minute Sheldon inputs the index of the country he talked about in an array (of size N) for later statistical analysis. But Howard has already tweaked his laptop with a weird algorithm to wrong Sheldon's statistical analysis. **Because of this, Sheldon's N-sized array now works as 2 stacks. Whenever Sheldon inputs an index, Howard's algorithm pushes the index in any 1 of the stacks. Also, in between, Howard's algorithm can pop any index from any stack.**

At the end of N minutes, when Sheldon's podcast ends, he wants his laptop to print the **mean, median, mode and standard deviation** of all the indices he talked about. But Howard's algorithm does not allow that! **Instead, his laptop first pops and prints all the elements of stack 1 and then pops and prints all the elements of stack 2,** leaving Sheldon confused.

Your task is output both the results – Sheldon's (actual mean, median, mode and standard deviation) and Howard's (the contents of stack 1 followed by stack 2). Look at the output section below for more details.

Input:

The first line contains an integer N (number of minutes for which the podcast runs) (1 <= N <= 100).

The second line contains an integer M (number of operations performed by Howard's algorithm) (1 <= M <= 200).

The next M lines contains 3 integers – Operation (1 for push, 2 for pop), Stack number (1 for stack 1, 2 for stack 2), Index (An integer in case of push, -1 in case of pop).

Note that he may talk about the flag of the same country more than once at different minutes. Also note that you don't have to print anything when you pop during any of the above M operations.

It is guaranteed that there will be no underflow or overflow conditions.

Output:

In the first line output 4 real numbers – the actual mean, median, mode and standard deviation that should have been printed if Howard's algorithm was not running.

In the second line pop and print all the remaining elements in stack 1. (Blank line in case of no elements in stack 1).

In the third line pop and print all the remaining elements in stack 2. (Blank line in case of no elements in stack 2).

Instructions:

1. You have to implement both the stacks in the same array.
2. Make a struct for the stacks with 3 variables – Array pointer, top1 for stack 1, top2 for stack 2.
3. Initialize the array using malloc(..) with N*sizeof(int).
4. It's true that structs in C can't have functions in them. You have to implement 4 functions outside the struct – push1(..), pop1(..), push2(..), pop2(..) for respective operations in stack 1 and stack 2.
5. For calculating mean, median, mode and standard deviation you can use another array to simply store all the elements, since stack operations on the array in the struct will modify it.
5. You have to format the output as expected.
6. Write the code preferably in C. If you use C++, you are not allowed to use the stack library.

Sample Input:

```
4
7
1 1 10
1 1 12
2 1 -1
1 2 4
1 2 3
2 2 -1
1 1 4
```

Sample Output:

```
6.6 4 4 3.66
4 10
4
```

Explanation:

After each of the operations – (left to right => top to bottom)

Stack 1 – 10	Stack 2 – null
Stack 1 – 12, 10	Stack 2 – null
Stack 1 – 10	Stack 2 – null
Stack 1 – 10	Stack 2 – 4
Stack 1 – 10	Stack 2 – 3, 4
Stack 1 – 10	Stack 2 – 4
Stack 1 – 4, 10	Stack 2 - 4