EndSem Lab Exam Set 1

Problem 1: Alternating Subarrays (40 points)

You are given a 0-indexed integer array `nums`. A subarray `s` of length `m` is called **alternating** if:

- `m` is greater than 1.
- `s[1] = s[0] + 1`.
- The 0-indexed subarray `s` looks like `[s[0], s[1], s[0], s[1], ..., s[(m-1) % 2]]`. In other words:
  - `s[1] - s[0] = 1`
  - `s[2] - s[1] = -1`
  - `s[3] - s[2] = 1`
  - `s[4] - s[3] = -1`
  - ...up to `s[m - 1] - s[m - 2] = (-1) ^ m`.

Return the maximum length of all alternating subarrays present in `nums` or `-1` if no such subarray exists.

Input format

First number i.e. n is the length of array and then there are n integers.

Examples

Example 1:

```
Input: nums = 3 4 5 6
Output: 2
Explanation:
[4, 5] and [5, 6] are the only two alternating subarrays. They are both of length 2.
```

Example 2:

```
Input: nums = 2 4 67
Output: -1
Explanation:
None possible so output is -1
```

Constraints

- $2 \leq$ `nums.length` ≤ 1000
- $1 \leq$ `nums[i]` $\leq 10^7$

Problem 2: Sensor Sorting Challenge (60 points)

A research lab relies on a temperature sensor to monitor its environment. The sensor logs temperature readings (one per line) into a file named sensor_data.txt. Unfortunately, the readings are unsorted, making analysis difficult. Your task is to write a C program to:

1. Read the temperature readings from sensor_data.txt using file handling.
2. Dynamically store the readins in a linked list
3. Sort the readings in ascending order
4. Display the sorted readings on the console

Input Format:

An input file named `input.txt` containing a list of numbers (positive and negative) representing temperature readings. Each line contains a single reading.

Output Format:

List of sorted temperature readings, one per line.

Example:

Initial File: `input.txt`

```
25.3
18.2
-10.5
30.1
0.0
```

Expected Output:

```
-10.500000 0.000000 18.200000 25.300000 30.100000
```

Constraints:

- The file can have up to 10,000 readings in the range -100.0 to 100.0 degrees Celsius.
- If the file is missing or empty, display "File does not exist" as its error message.
- Use only linked lists for storing data. Arrays are not allowed.
- display the output in %.6lf format.

Submission instructions:

- Boilerplate code is available, you're tasked to fill in the code for the functions.
- Do not make any changes in the boilerplate code.

Problem 3: Ring Calculator (80 points)

A new discovery, the ring calculator represents integers as beads in a circular ring. Further, at any given time a single bead is selected. There are two types of operations performed on the calculator. Arithmetic operations merge two beads to create a new bead. Shift operations change which bead is selected.

Initially there are N beads on the ring. If N is odd they contain the integers {-N/2 ... N/2}. If N is even they contain the integers {-N/2 ... (N/2)-1}. (ie. they contain N consecutive integers starting at -N/2). The bead containing 0 is the initially selected bead. The beads are initially stored in a clockwise (CW) ascending order, with the maximum bead just counter-clockwise (CCW) of the minimum bead.

The input file "input.txt" contains (N-1)+M lines. There are N-1 lines of arithmetic operations, and M lines of shift operations.

- An arithmetic operation is a line containing a single character belonging to {A, M, S}. These correspond to the 'A'dd, 'M'ultiply, and 'S'ubtract operations respectively. To perform each operation, consider that bead B1 is the selected bead and B2 is the bead immediately CW of B1. Let the result Y = B1 O B2 where 'O' is the operation given in the file. Then the two beads B1 and B2 are merged on the ring to form a single bead with value Y.
- A shift operation contains a single character (L or R) and positive integer (k) separated by a space. 'R'ight specifies a clockwise movement and 'L'eft specifies a counter-clockwise movement. To perform the operation, the new selected bead on the ring is the one with relative position specified by the operation. For example, given an operation 'R 3', the new selected bead is 3 beads clockwise to the previous selected bead. This operation allows us to perform arithmetic operations on different beads along the ring.

After performing all the operations from the file on the ring calculator, you will be left with a single node. Print the value of the final node, modulo 1e9+7.

File Format:

- File contains (N-1) + M lines
- There are (N-1) lines of arithmetic operations consisting of a single character (A/M/S)
- There are M lines of shift operations, consisting of a single character (L/R) followed by an integer k.

Constraints

- 1 <= N <= 10^6
- 0 <= M <= 10^5
- 1 <= K <= 10^5 for each of the M lines
- The sum of K over all M <= 10^6

Output Format:

Print 1 line of the number on the final bead. The output is an integer in the range {0 ... 1e6+6}

Example Input 1: (in input.txt)

```
M
A
S
```

Example Output 1:

```
1000000006
```

Example Input 2: (in input.txt)

```
A
L 1
M
R 1
A
S
```

Example Output 1:

```
1
```

Explanation 1

There are 3 arithmetic operations. Therefore N = 4. The ring is [-2, -1, 0, 1].

- The pointer starts at 0. operation is M. Therefore replace (0,1) with 0. Ring is now [-2, -1, 0]
- The pointer is at 0. operation is A. Therefore replace (0,-2) with -2. Ring is now [-1, -2]. (Note the cyclic nature of the ring)
- The pointer is at -2. operation is S. Therefore replace (-2, -1) with -1. Ring is now [-1].
- Output is -1 mod 1e9 + 7 = 1e9 +6

Explanation 2

There are 4 arithmetic operations, and 2 shift operations. Therefore N = 5. The ring is [-2, -1, 0, 1, 2].

- The pointer starts at 0. operation is A. Therefore replace (0,1) with 1. Ring is now [-2, -1, 1, 2]
- The pointer is at 1. operation is L 1. Therefore move pointer to -1. Ring is unchanged.
- The pointer is at -1. operation is M. Therefore replace (-1,1) with -1. Ring is now [-2, -1, 2]
- The pointer is at -1. operation is R 1. Therefore move pointer to 2. Ring is unchanged.
- The pointer is at 2. operation is A. Therefore replace (2,-2) with 0. Ring is now [-1, 0]
- The pointer is at 0. operation is S. Therefore replace (0,-1) with 1. Ring is now [1]
- Output is 1

There are 2 images attached in the handout (in the p3 subfolder) that illustrate the first 2 operations performed in example 2.

Image 1
Image 2

Note

- The handout recommends implementing the ring as a circular doubly linked list. This merely a recommendation, any approach that solves the problem is allowed (incl. arbitrary modifications to the code provided in the handout)
- You can implement the case of M = 0 (ie. no shift operations) to obtain 30% marks. Do consider all edge cases, especially small values of N.

Submission Guidelines

Do not rename any files given in the handout. Only write the code in the specified C files in the respective directories.