NOI Winter Camp '17 P2: Challenge

Time limit: 4.0s **Memory limit:** 1G

In this problem, there are 3 subtasks.

Subtask 1

Given n 32-bit unsigned integers, sort them in nondecreasing order.

Subtask 2

There are $2n$ people playing "Rock,Paper,Scissors". They stand in two rows, and there are n people in each row. A player will use a fixed strategy for every game: for the j-th ($0 \leq j < n$) player on the i-th ($i \in \{1, 2\}$) row, if we use an integer a_{ij} to describe his strategy, then 0 means the player will only use rock, 1 means the player will only use scissors, and 2 means the player will only use paper.

Now there are q queries. Each query specifies three integers x, y, l ($0 \leq x, y < n, 1 \leq l \leq n - \max(x, y)$). You need to answer how many people in the first row will win if the $x \sim x + l - 1$-th person in the first row plays the game with the $y \sim y + l - 1$-th person on the second row.

More formally, "plays the game" here means for all i satisfying $0 \leq i < l$, the $x + i$-th person on the first row plays "Rock,Paper,Scissors" with the $y + i$-th person on the second row.

Subtask 3

We say a parenthesis sequence is valid if it is a sequence that is (1) formed entirely by `(` and `)` (2) the numbers of `(` and `)` are equal (3) for any prefix, the number of `(` is no less than the number of `)`. Now given a string formed by `(`, `)`, and `?`, compute the number of ways to replace each `?` with `(` or `)` such that the parenthesis sequence is valid. We say two solutions are different if and only if there is at least one `?` replaced with different parenthesis.

Input Format

This problem has a template. The first line of the input has an integer $task_{id}$ ($1 \leq task_{id} \leq 3$) denoting the subtask. Next is the specific input to a subtask. Two adjacent integers in the same line are separated by a space.

- Subtask 1: There is a line with two integers n, s. Let $a_0 = \text{next\_integer}(s)$, $a_i = \text{next\_integer}(a_{i-1})$, $1 \leq i < n$. Then $a_0, a_1, \ldots, a_{n-1}$ is the n integers that shall be sorted.
- Subtask 2: The first line contains two integers n, q. In the second line, there is a string of length n consisting of 0,1,2. The i-th letter of the string denotes the strategy of the i-th person in the first row (i.e. a_{1i}). The third line has the same format as the second line. The third line denotes the strategies of the people on the second row.
- Subtask 3: The first line contains an integer n denoting the length of the string. The second line is the string.

Output Format

- Subtask 1: Let b be the sorted array. Call $\text{output\_arr}(b, 4q)$.
- Subtask 2: Store the answers to the queries in an array of 32-bit unsigned integers b (i.e. store into $b_0, b_1, \ldots, b_{q-1}$), and call $\text{output\_arr}(b, 4q)$.
- Subtask 3: Output an integer denoting the number of possibilities modulo 2^{32}.

Sample Input 1

```
1
100000 2017012501
```

Sample Output 1

```
4275990336
```

Sample Input 2

```
2
6 6
200100
200211
5 3 1
2 0 1
2 0 3
2 0 2
2 3 3
0 1 3
```

Sample Output 2

```
3349208141
```

Sample Input 3

```
3
4
(???
```

Sample Output 3

```
2
```

Sample Input 4

```
3
4
)???
```

Sample Output 4

```
0
```

Constraints

In the original problem, the memory limit is 2 GB. Due to limitations of DMOJ, the memory limit has to be 1 GB and thus it is likely the 3rd test case is not solvable on DMOJ.

Subtask	Score	Test Case	Constraints
1	5	1	$n = 10^5$
	19	2	$n = 10^8$
	11	3	$n = 2 \times 10^8$
2	7	4	$n = q = 10^3$
	23	5	$n = q = 3 \times 10^5$
3	9	6	$n = 10^3$
	5	7	$n = 120\,000$
	7	8	$n = 225\,000$
	14	9	$n = 266\,666$

Test Case 3

```
1
200000000 2017012503
```

Template

```cpp
#include <stdio.h>
#include <string.h>
#include <algorithm>

typedef unsigned int u32;
typedef unsigned long long u64;

inline u32 next_integer(u32 x) {
    x ^= x << 13;
    x ^= x >> 17;
    x ^= x << 5;
    return x;
}

bool output_arr(void *a, u32 size) {
    if (size % 4) {
        return puts("-1"), 0;
    }

    u32 blocks = size / 4;
    u32 *A = (u32 *)a;
    u32 ret = size;
    u32 x = 23333333;
    for (u32 i = 0; i < blocks; i++) {
        ret = ret ^ (A[i] + x);
        x ^= x << 13;
        x ^= x >> 17;
        x ^= x << 5;
    }

    return printf("%u\n", ret), 1;
}

// ===== header ======

namespace Sorting {
void init_data(u32 *a, int n, u32 seed) {
    for (int i = 0; i < n; i++) {
        seed = next_integer(seed);
        a[i] = seed;
    }
}

void main() {
    int n;
    u32 seed;
    scanf("%d%u", &n, &seed);
```

```
    u32 *a = new u32[n];
    init_data(a, n, seed);

    // sort(a, n);

    output_arr(a, n * sizeof(u32));
}
}

namespace Game {
void main() {
    int n, q;
    scanf("%d%d", &n, &q);

    char *s1 = new char[n + 1];
    char *s2 = new char[n + 1];
    scanf("%s%s", s1, s2);

    u32 *anss = new u32[q];
    int *q_x = new int[q];
    int *q_y = new int[q];
    int *q_len = new int[q];

    for (int i = 0; i < q; i++) {
        scanf("%d%d%d", q_x + i, q_y + i, q_len + i);
    }

    // solve(n, q, s1, s2, q_x, q_y, q_len, anss);

    output_arr(anss, q * sizeof(u32));
}
}

namespace Parentheses {
void main() {
    int n;
    scanf("%d", &n);

    char *s = new char[n + 1];
    scanf("%s", s);

    u32 ans;
    // ans = solve(n, s);

    printf("%u\n", ans);
}
}
```

```
int main() {
    int task_id;
    scanf("%d", &task_id);

    switch (task_id) {
        case 1:
            Sorting::main();
            break;
        case 2:
            Game::main();
            break;
        case 3:
            Parentheses::main();
            break;
    }

    return 0;
}
```