5 Rope

Problem Introduction

In this problem you will implement Rope — data structure that can store a string and efficiently cut a part (a substring) of this string and insert it in a different position. This data structure can be enhanced to become persistent — that is, to allow access to the previous versions of the string. These properties make it a suitable choice for storing the text in text editors.

This is a very advanced problem, harder than all the previous advanced problems in this course. Don't be upset if it doesn't crack. Congratulations to all the learners who are able to successfully pass this problem!

Problem Description

Task. You are given a string S and you have to process n queries. Each query is described by three integers i, j, k and means to cut substring $S[i..j]$ (i and j are 0-based) from the string and then insert it after the k-th symbol of the remaining string (if the symbols are numbered from 1). If $k = 0$, $S[i..j]$ is inserted in the beginning. See the examples for further clarification.

Input Format. The first line contains the initial string S.
The second line contains the number of queries q.
Next q lines contain triples of integers i, j, k.

Constraints. S contains only lowercase english letters. $1 \le |S| \le 300\,000$; $1 \le q \le 100\,000$; $0 \le i \le j \le n - 1$; $0 \le k \le n - (j - i + 1)$.

Output Format. Output the string after all q queries.

Time Limits.

language	C	C++	Java	Python	C#	Haskell	JavaScript	Ruby	Scala
time (sec)	3	3	6	120	4.5	6	120	120	12

Memory Limit. 512MB.

Sample 1.

Input:

```
hlelowrold
2
1 1 2
6 6 7
```

Output:

```
helloworld
```

Explanation:
hlelowrold → *hellowrold* → *helloworld*
When $i = j = 1$, $S[i..j] = l$, and it is inserted after the 2-nd symbol of the remaining string *helowrold*, which gives *hellowrold*. Then $i = j = 6$, so $S[i..j] = r$, and it is inserted after the 7-th symbol of the remaining string *hellowold*, which gives *helloworld*.

Sample 2.

Input:

```
abcdef
2
0 1 1
4 5 0
```

Output:

```
efcabd
```

$abcdef \to cabdef \to efcabd$

Starter Files

The starter solutions for C++ and Java in this problem read the input, implement a naive algorithm to cut and paste substrings and write the output. The starter solution for Python3 just reads the input and writes the output. You need to implement a data structure to make the operations with string very fast. If you use other languages, you need to implement the solution from scratch.

What to Do

Use splay tree to store the string. Use the split and merge methods of the splay tree to cut and paste substrings. Think what should be stored as the key in the splay tree. Try to find analogies with the ideas from this lecture.

Need Help?

Ask a question or see the questions asked by other learners at this forum thread.

6 Appendix

6.1 Compiler Flags

C (`gcc 7.4.0`). File extensions: `.c`. Flags:

```
gcc -pipe -O2 -std=c11 <filename> -lm
```

C++ (`g++ 7.4.0`). File extensions: `.cc`, `.cpp`. Flags:

```
g++ -pipe -O2 -std=c++14 <filename> -lm
```

If your `C/C++` compiler does not recognize `-std=c++14` flag, try replacing it with `-std=c++0x` flag or compiling without this flag at all (all starter solutions can be compiled without it). On Linux and MacOS, you most probably have the required compiler. On Windows, you may use your favorite compiler or install, e.g., `cygwin`.

C# (`mono 4.6.2`). File extensions: `.cs`. Flags:

```
mcs
```

Go (`golang 1.13.4`). File extensions: `.go`. Flags

```
go
```

Haskell (`ghc 8.0.2`). File extensions: `.hs`. Flags: