Procesando un texto

Contribución de Agustín Santiago Gutiérrez

Descripción del problema

Ricardo quiere crear un procesador de texto que sea mucho mejor que los existentes.

Un procesador de texto es, para Ricardo, una función que toma un texto y una lista de comandos, y va aplicando sucesivamente los comandos indicados en orden, para obtener un texto final como resultado.

Más precisamente, un texto es una cadena de inicialmente **N** caracteres formada solo por letras minúsculas y/o mayúsculas del alfabeto inglés. Los caracteres en mayúsculas y minúsculas se consideran diferentes, es decir por ejemplo `c` no es equivalente a `C`.

Los **T** comandos que hay que aplicar al texto inicial se describen cada uno mediante una cadena de caracteres. Las cadenas posibles para describir comandos son:

- `INTERCAMBIA`: Las mayúsculas del texto se vuelven minúsculas, y viceversa.
- `BORRAULTI`: Borra el ultimo caracter. Si el texto era vacío, no hace nada.
- `BORRAPRI`: Borra el primer caracter. Si el texto era vacío, no hace nada.
- `DUP`: Duplica la cadena. Por ejemplo si era `coPIA`, se transforma en `coPIAcoPIA`.
- `ROTA`: Rota la cadena, llevando el primer caracter al final. Por ejemplo si era `aBCD`, se transforma en `BCDa`. Si el texto era vacío, no hace nada.
- `INVERTIR`: Invierte toda la cadena. Por ejemplo `abcdef` cambia a `fedcba`.
- `CHAUAGUS`: Borra **la primera** aparicion del texto `AGUS` en la cadena, **ya sea que aparezca en mayúscula o minúscula o mezclado**. Si no aparece no hace nada. Por ejemplo `aguaAgUSagusyAGUS` se transforma en `aguaagusyAGUS`.
- `AGREGA-`α: Agrega la cadena α dada en el comando, al final del texto. Por ejemplo el comando `AGREGA-xYzz` transformaría `abc` en `abcxYzz`
- `DUP-`**i**`-`**j**: Duplica la subcadena que empieza en el caracter **i** y termina en el **j**, ahi mismo donde aparece. Por ejemplo el comando `DUP-3-7` transformaría `xxabcefyy` en `xxabcefabcefyy`.
- `INVERTIR-`**i**`-`**j**: Invierte la subcadena que empieza en el caracter **i** y termina en el **j**, ahi mismo donde aparece. Por ejemplo el comando `INVERTIR-3-7` transformaría `xxabcefyy` en `xxfecbayy`.
- `BORRA-`**i**: Borra el **i**-ésimo caracter.
- `CHAU-`α: Borra **la primera** aparicion del texto α en la cadena, **que tiene que coincidir exacto en mayúsculas y minúsculas**. Si no aparece no hace nada. Por ejemplo el comando `CHAU-xY` transformaría `xyaxYbxY` en `xyabxY`.

El resultado que se devuelve es el texto final luego de aplicar los **T** comandos indicados.

En esta primera versión, Ricardo quiere que el procesador de texto tenga una memoria de hasta **1000** caracteres. Si en cualquier momento (ya sea en el resultado final, o en el texto que resulta de aplicar los primeros **k** comandos, para algún $\mathbf{1} \leq \mathbf{k} \leq \mathbf{T}$) se forma un texto de estrictamente más de **1000** caracteres, se debe retornar la cadena "`MemoryLimitExceeded`" (sin las comillas).

Descripción de la función

Debes implementar la función `procesatexto(texto, comandos)`

Sus parámetros son:

- `texto`: Una cadena con el texto inicial que será procesado.
- `comandos`: Un arreglo de **T** cadenas, que indican en orden los comandos que se deben aplicar al texto.

La función debe retornar una cadena, con el texto resultante luego de aplicar los **T** comandos.

Evaluador local

El evaluador lee de la entrada estándar la cadena `texto` y el entero **T**. Luego lee **T** cadenas de texto, los elementos del arreglo `comandos`.

Escribe a la salida estándar la cadena retornada por la llamada `procesatexto(texto, comandos)`.

Restricciones

- **1** $\leq$ **N** $\leq$ **100**
- **1** $\leq$ **T** $\leq$ **100**
- Cada comando es una cadena de hasta **50** caracteres
- Se garantiza que en los comandos `DUP-`**i**`-`**j** e `INVERTIR-`**i**`-`**j**, se tiene **i** $\leq$ **j** y tanto **i** como **j** son índices válidos de un caracter en el texto al momento de ejecutar el comando
- Se garantiza que en los comandos `BORRA-`**i**, **i** es un índice válido de un caracter en el texto al momento de ejecutar el comando
- Tanto el texto inicial como las cadenas α en los comandos explicados contienen únicamente letras mayúsculas y minúsculas del alfabeto inglés

Ejemplo

Si se invoca al evaluador con la siguiente entrada:

```
CACEROLAcasaLimonXYZagusiagus 10
INVERTIR
CHAUAGUS
INVERTIR
CHAUAGUS
BORRA-24
BORRAULTI
INVERTIR
BORRAULTI
INTERCAMBIA
INVERTIR
```

Para un programa correcto, la salida será:

```
acerolaCASAlIMONxyzIAG
```

Subtareas

1. **T** = **1**, comando `INTERCAMBIA` (5 puntos)
2. **T** = **1**, comando `BORRAULTI` (5 puntos)
3. **T** = **1**, comando `BORRAPRI` (5 puntos)
4. **T** = **1**, comando `DUP` (5 puntos)
5. **T** = **1**, comando `ROTA` (5 puntos)
6. **T** = **1**, comando `INVERTIR` (5 puntos)
7. **T** = **1**, comando `CHAUAGUS` (5 puntos)
8. **T** = **1**, comando `AGREGA-`α (5 puntos)
9. **T** = **1**, comando `DUP-`**i**`-`**j** (5 puntos)
10. **T** = **1**, comando `INVERTIR-`**i**`-`**j** (5 puntos)
11. **T** = **1**, comando `BORRA-`**i** (5 puntos)
12. **T** = **1**, comando `CHAU-`α (5 puntos)
13. **T** = **1** (20 puntos)
14. Sin más restricción (20 puntos)