Escrivão camarão

Algumas pessoas dizem que Leonardo era um grande admirador de Johannes Gutenberg, o ferreiro alemão que inventou impressão por tipos móveis, e que ele prestou-lhe homenagem através da concepção de uma máquina chamada de escrivão camarão - *Il Gambero Scrivano* - um dispositivo muito simples de digitação. É de certa forma semelhante a uma máquina de escrever moderna simples e aceita apenas dois comandos: um para digitar o próximo caractere e um para desfazer os comandos mais recentes. A característica notável do escrivão camarão é que o comando desfazer é extremamente poderoso: um desfazer também é considerado um comando por si mesmo, e pode ser desfeito.

Enunciado

Sua tarefa é escrever uma versão do software do escrivão de camarão: começa com um texto em branco e aceita uma sequênca de commandos fornecidos pelo usuário e consultas para posições específicas da versão corrente do texto, como segue.

- `Init`() — chamada apenas uma vez no início da execução, sem argumentos. Pode ser usada para inicializar estruturas de dados. Nunca precisará ser desfeita.
- `TypeLetter`(L) — anexa no finial do texto uma letra minúscula L escolhida entre `a`, ..., `z`.
- `UndoCommands`(U) — desfaz os últimos U comandos, onde U é um inteiro positivo.
- `GetLetter`(P) — devolve a letra na posição P do texto atual, onde P é um índice não negativo. A primeira letra do texto possui índice 0. (Esta consulta não é um comando e deve ser ignorada pelo comando desfazer.)

Depois da chamada inicial para `Init`(), as outras rotinas podem ser chamadas zero ou mais vezes em qualquer ordem. É garantido que U não excederá o número de comandos recebidos previamente, e que P será menor que o tamanho do texto atual (o número de letras no texto atual).

`UndoCommands`(U), desfaz os U comandos anteriores na ordem "inversa": se o comando que deve ser desfeito é `TypeLetter`(L), então este remove L do final do texto atual; se o comando que deve ser desfeito é `UndoCommands`(X) para um valor X, este re-faz os X comandos anteriores na ordem "original".

Exemplo

Nos mostraremos uma possível sequência de chamadas, juntas com o estado do texto depois de cada chamada.

Chamada	Devolução	Texto atual
`Init()`		
`TypeLetter(a)`		`a`
`TypeLetter(b)`		`ab`
`GetLetter(1)`	`b`	`ab`
`TypeLetter(d)`		`abd`
`UndoCommands(2)`		`a`
`UndoCommands(1)`		`abd`
`GetLetter(2)`	`d`	`abd`
`TypeLetter(e)`		`abde`
`UndoCommands(1)`		`abd`
`UndoCommands(5)`		`ab`
`TypeLetter(c)`		`abc`
`GetLetter(2)`	`c`	`abc`
`UndoCommands(2)`		`abd`
`GetLetter(2)`	`d`	`abd`

Subtarefa 1 [5 pontos]

- O número de comandos e consultas é entre 1 e 100 (inclusive) e não existirão chamadas para `UndoCommands`.

Subtarefa 2 [7 pontos]

- O número total de comandos e consultas é entre 1 e 100 (inclusive) e nenhum `UndoCommands` será desfeito.

Subtarefa 3 [22 pontos]

- O número total de comandos e consultas é entre 1 e 5 000 (inclusive).

Subtarefa 4 [26 pontos]

- O número total de comandos e consultas é entre 1 e 1 000 000 (inclusive). Todas as chamadas de `GetLetter` acontecerão depois das chamadas de `TypeLetter` e `UndoCommands`.

Subtarefa 5 [40 pontos]

- O número total de comandos e consultas é entre 1 e 1 000 000 (inclusive).

Detalhes de implementação

Você deve submeter exatamente um arquivo, chamado `scrivener.c`, `scrivener.cpp` ou `scrivener.pas`. O arquivo precisa implementar os subprogramas descritos acima usando as seguintes assinaturas.

Programas C/C++

```
void Init();
void TypeLetter(char L);
void UndoCommands(int U);
char GetLetter(int P);
```

Programas Pascal

```
procedure Init;
procedure TypeLetter(L : Char);
procedure UndoCommands(U : LongInt);
function GetLetter(P : LongInt) : Char;
```

Estes subprogramas devem comportar-se como descrito acima. Claro que é livre de implementar outros subprogramas para uso interno. As suas submissoes não devem interagir de qualquer forma com o standard input/output, nem com qualquer outro ficheiro.

Avaliador fornecido

O avaliador fornecido lê o input no seguinte formato:

- linha 1: o número total de comandos e consultas na entrada;
- em cada uma das linhas seguintes:
  - `T` seguido por um espaço e uma letra minúscula para um comando `TypeLetter`;
  - `U` seguido por um espaço e um inteiro para `UndoCommands`;
  - `P` seguido por um espaço e um inteiro para `GetLetter`.

O avaliador fornecido imprimirá os caracteres devolvidos por `GetLetter`, cada um em uma linha separada.