Le scribe écrevisse (Crayfish scrivener)

Certains personnes disent que Léonard était un grand admirateur de Johannes Gutenberg, le forgeron allemand qui a inventé l'imprimerie à caractères mobiles, et qu'il lui a rendu hommage en concevant une machine appelée le scribe écrevisse — *il gambero scrivano* — une machine à écrire très simple. Celle-ci ressemble a une machine à écrire moderne mais n'accepte que deux commandes : une pour écrire le prochain caractère et une pour annuler les commandes les plus récentes. La fonctionnalité la plus remarquable du scribe écrevisse est que la commande annuler est extrêmement puissante : annuler est également considéré comme une commande et peut donc également être annulé.

Problème

Votre tâche est d'écrire une version informatique du scribe écrevisse : il commence avec un texte vide et accepte une suite de commandes entrées par l'utilisateur et de requêtes permettant de récupérer le contenu à certaines positions du texte courant. Les voici :

- `Init`() — appelé une fois en début d'exécution, sans argument. Elle peut être utilisée afin d'initialiser des structures de données. Elle ne peut être annulée.
- `TypeLetter`(L) — ajoute à la fin du texte un lettre L en minuscule parmi `a`, …, `z`.
- `UndoCommands`(U) — annule les U dernières commandes, U étant un entier strictement positif.
- `GetLetter`(P) — retourne la lettre à la position P du texte courant, pour un indice P positif ou nul. La première lettre du texte a l'indice 0. (Cette requête n'est pas une commande et donc doit être ignorée par la commande annuler.)

Après l'appel initial à `Init`(), les autres fonctions ci-dessus peuvent être appelées zéro, une ou plusieurs fois, dans n'importe quel ordre. On vous garantit que U ne dépassera pas le nombre de commandes précédentes, et que P est strctement plus petit que la longueur du texte actuel (le nombre de lettres du texte courant).

Pour ce qui est de la commande `UndoCommands`(U), elle annule les U dernières commandes dans l'ordre inverse de leur exécution : si la commande à annuler est `TypeLetter`(L), alors elle efface L de la fin du texte courant ; si la commande à annuler est `UndoCommands`(X) pour une valeur X, elle ré-exécute les X commandes précédentes dans leur ordre initial.

Exemple

Voici une séquence de commandes possibles et l'état du texte après chaque appel.

Commande	Retourne	Texte courant
`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

Sous-tâche 1 [5 points]

- Le nombre total de commandes et de requêtes est compris entre 1 et 100 (inclus) et il n'y aura aucun appel à `UndoCommands`.

Sous-tâche 2 [7 points]

- Le nombre total de commandes et de requêtes est compris entre 1 et 100 (inclus) et aucune commande `UndoCommands` ne sera annulée.

Sous-tâche 3 [22 points]

- Le nombre total de commandes et de requêtes est compris entre 1 et 5 000 (inclus).

Sous-tâche 4 [26 points]

- Le nombre total de commandes et de requêtes est compris entre 1 et 1 000 000 (inclus). Tous les appels à `GetLetter` auront lieu après tous les appels à `TypeLetter` et `UndoCommands`.

Sous-tâche 5 [40 points]

- Le nombre total de commandes et requêtes est compris entre 1 et 1 000 000 (inclus).

Détails d'implémentation

Vous devez soumettre exactement un fichier, appelé `scrivener.c`, `scrivener.cpp` ou `scrivener.pas`. Ce fichier doit implémenter les fonctions décrites ci-dessus en utilisant les signatures suivantes.

Programmes en C/C++

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

Progammes Pascal

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

Ces fonctions doivent se comporter comme décrit ci-dessus. Bien entendu, vous êtes libre d'implémenter d'autres fonctions pour un usage interne. Vos soumissions ne peuvent pas interagir de quelque façon que ce soit avec l'entrée/la sortie standard, ni aucun autre fichier.

Exemple d'évaluateur

L'évaluateur d'exemple lit l'entrée au format suivant:

- ligne 1 : le nombre total de commandes et requêtes fournies en entrée ;
- sur chacune des lignes suivantes :
  - `T` suivi par un espace et une lettre minuscule pour une commande `TypeLetter` ;
  - `U` suivi par un espace et un entier pour `UndoCommands` ;
  - `P` suivi par un espace et un entier pour `GetLetter`.

L'évaluateur d'exemple affichera les caractères retournés par `GetLetter`, chacun sur une ligne séparée.