

Programmation séquentielle

Série 12 - Liste doublement chaînées pour un navigateur

21.02.2023

Les listes doublement chaînées

Buts

- Manipulation de pointeurs et chaînes de caractères.
- Implémentation d'une liste doublement chaînée.
- Structuration d'un programme en librairie et programme utilisateur.
- Implémentation d'une interface CLI simple.
- Gérer la mémoire dynamique.

Librairie de liste doublement chaînée de chaînes de caractères

La première étape est d'écrire une librairie permettant de manipuler des listes doublement chaînées contenant des chaînes de caractères. Pour cela, on utilisera la structure de données suivante :

```
typedef struct element_{  
    char* data;  
    struct element_* prev;  
    struct element_* next;  
}element;
```

```
typedef struct dll_{  
    element* head;  
    element* cur;  
}dll;
```

Vous implémenterez les fonctions suivantes pour manipuler ce type de liste :

```
// === Création et consultation ===  
// crée la liste doublement chaînée  
dll dll_create();  
// retourne la valeur à la position actuelle dans la liste  
char* dll_value(dll list);  
// la liste est-elle vide?  
bool dll_is_empty(dll list);  
// Est-ce que cur est le 1er élément?
```

```

bool dll_is_head(dll list);
// Est-ce que cur est le dernier élément?
bool dll_is_tail(dll list);
// data est-elle dans la liste?
bool dll_is_present(dll list, char* data);
// affiche la liste
void dll_print(dll list);

// === Manipulation ===
// déplace cur au début de la liste
dll dll_move_to_head(dll list);
// déplace cur à la position suivante dans la liste
dll dll_next(dll list);
// déplace cur à la position précédente dans la liste
dll dll_prev(dll list);

// === Insertion ===
// insertion de data dans l'élément après cur
dll dll_insert_after(dll list, char* data);
// insertion de data en tête de liste
dll dll_push(dll list, char* data);

// === Extraction ===
// extraction de la valeur se trouvant dans l'élément cur
// l'élément cur est libéré (mais pas la chaîne de caractères)
char* dll_extract(dll *list);
// extrait la donnée en tête de liste
// l'élément en tête est libéré (mais pas la chaîne de caractères)
char* dll_pop(dll *list);
// vide la liste (les chaînes de caractères sont libérées)
void dll_destroy(dll *list)

```

Lorsqu’une chaîne de caractère est insérée dans la liste (via `push` ou `insert_after`), c’est la responsabilité de la librairie de liste de gérer la mémoire de la chaîne (par exemple lors d’un `destroy`). Lorsqu’une chaîne de caractères est extraite de la liste (via `extract` ou `pop`), c’est la responsabilité de l’utilisateur de gérer la mémoire de la chaîne.

Vous validerez la bonne gestion de la mémoire par votre librairie à l’aide des sanitizers.

Un pseudo navigateur web

A l’aide de la librairie implémentée précédemment, vous devez implémenter un pseudo navigateur en CLI. Le navigateur attend une “adresse” web en entrée (une chaîne de caractères) et la “visite” (affiche la chaîne de caractères). Les sites visités sont ajoutés à une liste doublement chaînée. Lorsque l’utilisateur entre la commande “b”, le navigateur consulte le site précédent dans la liste, et lorsque l’utilisateur entre la commande “n”, le navigateur visite le site suivant dans la liste.

Si un utilisateur entre un nouveau site alors que le site courant n'est pas le dernier, tous les éléments après le site courant sont supprimés de la liste avant d'ajouter le nouveau site à la liste. Si l'utilisateur choisit la commande "b" alors qu'il est déjà au début de la liste, le premier site de la liste est réaffiché. De même si l'utilisateur choisit "n" alors qu'il est déjà en fin de liste.

Un exemple d'affichage de ce programme :

```
> truc.com
visiting truc.com
> bidule.org
visiting bidule.org
> machin.net
visiting machin.net
> n
visiting machin.net
> b
visiting bidule.org
> b
visiting truc.com
> chose.ch
visiting chose.ch
> b
visiting truc.com
> n
visiting chose.ch
> n
visiting chose.ch
> site.eu
visiting site.eu
> b
visiting chose.ch
> b
visiting truc.com
```