Rapport du projet « automate LR(1) »

Fait part KHIDOUR Alae

Sommaire :

I- Les besoins du programme :

1) Bibliothèques et structures :

Tout d'abord j'ai eu besoin de deux bibliothèques principalement : stdio.h et stdlib.h. pour les fonctions suivantes :fopen, fclose, fgetc, fgets, fprint, fscancf et fseek.

Après avoir lu le sujet, la structure la plus pertinente à implémenter était la structure de pile : pour cela j'ai décider d'implémenter la structure de pile avec des listes chaînées en définissant deux structure :
-Tout d'abord la structure « Élément » et donc pour chaque élément on a sa valeur et un pointeur vers l'élément suivant de la pile .
-Ensuite une deuxième structure « Pile » qui est tout simplement définie par le premier élément de la pile qui donc pointe vers le second et ainsi de suite jusqu'à la fin de la pile.

Pour exploiter cette structure de Pile , j'ai donc dû codé la fonction « initialiser » qui renvoie une pile vide, ainsi que les fonctions « empiler » et « dépiler », et enfin pour la partie débogage la fonction « afficherPile » (qui d'ailleurs n'est pas utilisée dans le code final et que j'ai donc enlevé).

Précision sur les valeurs des éléments de la pile : sachant que la pile stockera des états et que les états sont codé sur un octet et ayant une valeur comprise entre 0 et 255, tout naturellement les valeurs des éléments de la pile sont du type « unsigned char ».

2) Utilisation du projet :

Le projet est divisé en 3 parties : (Pile.c/Pile.h), (fonctionsAutomate.c/fonctionsAutomate.h) et le main.c. Pour faire appel au programme on compile d'abord avec
« gcc -Wall "main.c" "Pile.c" "fonctionAutomate.c" -o "main" » et on lance le programme avec
« ./main.exe *nomDeL'automaton.aut* ».

II- L'algorithme du programme :

En lisant le sujet on comprend qu'il va falloir coder les fonctions action, réduire, décale et branchement. Je commence donc par les coder pour pouvoir les utiliser dans le main et ainsi exécuter l'algorithme de l'automate. Vu que le sujet décrivait bien comment est écrit le fichier .aut j'ai décider de ne pas le stocker dans des tableaux ou des matrices mais plutôt aller chercher directement l'information dont j'ai besoin

avec des fseek (méthode que j'ai trouvé plus simple que devoir stocker toutes les informations de l'automate).

Pour cela j'ai codé une fonction « nombre_etat » qui lit donc la première ligne de l'automate « a n » et qui donc renvoie « n » le nombre d'états de l'automate que j'utilise dans les fonctions réduire, décale et branchement. Pour me placer exactement sur l'octet qui m'intéresse vu que je savait que la première ligne contenait 4 octets « a n\n », la deuxième ligne contenait 128*n+1 octets, la troisième et la quatrième ligne contenaient n+1 octets. Et donc avec ces informations je pouvait me placer où je voulais de la première ligne à la cinquième ligne. Une fois toutes les fonctions prêtes je n'ai que suivi la méthode décrite dans le sujet dans mon main en utilisant les fonctions codées avant pour appliquer mon automate sur l'entrée texte.

III- Problèmes rencontrés :

En théorie j'aurais pu me placer directement sur le début de la ligne qui m'intéressait avec des fgets et donc ne pas coder la fonction nombre_etat mais cela m'avait posé un problème que je n'avait pas su résoudre car je ne le comprenait tout simplement pas : en utilisant le fgets plus d'une fois le curseur ne se plaçait pas toujours au bon endroit (c'est à dire au début de la ligne suivante) et je n'arrivais pas à comprendre pourquoi, cela est sûrement dû à une erreur de ma part mais au lieu de perdre du temps sur cela j'ai préféré opter pour une solution plus sûre et profiter du fait que je connais la manière dont est écrit le fichier et donc placer manuellement le curseur là où je voulais.