Projet Algorithme 2

Modules

- **str_buf** : Le module str_buf est une implémentation d'un tableau de caractère, d'une capacité, d'une longueur, d'un filtre et d'une énumération. Lorsque la longueur de la chaîne de caractère dépasse la capacité du tableau, une ré-allocation est effectuée et sa capacité est donc doublée. Le facteur de ré-allocation a été arbitrairement choisi à 2. Grâce à une fonction d'ajout d'un caractère nous pouvons filtrer ses ajouts avec une suite de caractère :

  - a : filtre avec la fonction isalpha
  - c : filtre avec la fonction iscrntl
  - d : filtre avec la fonction isdigit
  - n : filtre avec la fonction isalnum
  - p : filtre avec la fonction ispunct
  - s : filtre avec la fonction isspace

  En plus de cela l'énumération a pour valeur UPPER, LOWER ou NOTHING ; le comportement du str_buf en découle, en effet UPPER permet de mettre en majuscule, LOWER en minuscule et NOTHING n'effectue aucun changement sur les caractères ajoutés.

  Exemple :

  ```
  str_buf *s = str_buf_empty("ad", UPPER);
  ```

  Cet exemple va créer un str_buf. Lors d'un ajout de caractère il ne serra ajouté seulement s'il correspond à une lettre (is alpha) ou à un chiffre (isdigit), de plus celui-ci serra automatiquement mis en majuscule. Ce filtre correspond en réalité au filtre n seul. Ceci permettra de gérer les différentes options de lors de l'exécution du programme (-f et -c). Et les autres spécificités sont expliquées dans le .h du module str_buf.

- **line** : Le module line est une implémentation d'un tableau à deux dimensions dynamique sur une structure occrs, et d'une taille représentant le nombre de fichier actuel. Lors de son allocation une place mémoire pour une seule occrs est faite. Cette structure est *static* à notre définition du module et a pour but d'enregistrer tous les numéros de ligne où de cette même ligne a été rencontrée. A chaque ajout d'un fichier le

tableau de notre structure line est ré-alloué pour obtenir une nouvelle place pour une nouvelle occrs qui correspond à ce nouveau fichier. Aussi le tableau définit par occrs est lui aussi ré-alloué pour permettre d'enregistrer des numéros de lignes sans limites (mise à part *LONG_MAX*).

```
struct line {
    occrs **list_file;
    size_t n;
};
```

```
struct occrs {
    long int *array;
    size_t capacity;
    size_t length;
};
```

- **helpstruct** : Ce module est simplement un module dédié à la présentation du menu d'aide et permet de stocker dans une structure les différentes options disponibles et leur argument. Ainsi toutes ces options sont modulables.

- **hashtable** : Ce module est une implémentation d'une simple table de hachage et permet ainsi de réduire le temps de comparaison des différentes chaînes de caractère lue(s) dans le(s) fichier(s). Ainsi nous associons un pointeur vers une chaîne de caractère (donnée directement par le module str_buf, une nouvelle place est alloué pour cette chaîne) à un pointeur vers une structure line.

- **slist** : Ce module est une implémentation d'une liste chaînée. Elle nous permet entre autre d'enregistrer dans une liste toutes les clés de la table de hachage (ainsi nous pouvons la parcourir en entier), et aussi d'enregistrer toutes les structures line, ce qui permet lorsqu'un nouveau fichier est ouvert de signifier à toutes les structures qu'un nouveau fichier est présent.

Fonction principale

Le fichier *main.c* est le corps du programme. Celui-ci gère les différentes options que nous avons implémentées. Tout d'abord l'option *-c TYPE* ou *--case=TYPE* à été ajoutée et est fonctionnelle. L'option *-f MOTIF* ou *--filter=MOTIF* à aussi été ajoutée et les filtres sont disponibles parmi ceux proposés. Dans les exemples ci-dessous nous pouvons voir que le filtre *ad* est utilisée et que le comportement est donc celui de *n*. L'option *-s* ou *-sort* quant à elle n'a pas été implémentée.

```
tom-pc@tom-pc--virtual:~/Documents/L2 - Semestre 1/Algo_2/Projet/main$ ./lntracker -c upper hashtbl.c
hashtbl.c
12,80,102,106,140          }
17,48,91          SIZE_T HKEY;
19,26   };
31,38,89,118        RETURN NULL;
32,39,42,57,60,68,71,81,90,112,119,144    }
46,62,86,114,126,130,145        }
94,108      (*PP) -> VALUE = VALUE;
100,139       }
113,125   RETURN VALUE;
116,128   CLIST **PP = HASHTABLE_SEARCH(HT, KEY, NULL);
```

```
tom-pc@tom-pc--virtual:~/Documents/L2 - Semestre 1/Algo_2/Projet/main$ ./lntracker -f ad hashtbl.c hashtbl.h
hashtbl.c       hashtbl.h
1       1       constvoidvalue
2       1       intcomparconstvoidconstvoid
```

Grâce à cette exemple nous pouvons aussi remarquer que la comparaison avec plusieurs fichiers a aussi été implémentée. Ainsi lorsque plusieurs fichiers sont données en argument, au lieu de nous donner les numéros de lignes identiques celui-ci nous donne le nombre d'occurrence des lignes présentes dans tous les fichiers. Grâce aux macros fonctions ON_VALUE_GOTO et NOT_ON_VALUE_GOTO les erreurs de toutes les fonctions sont traitées et la libération des ressources allouées s'effectue bien.

Difficultés et limites

L'ordre des options et des fichiers est très important. Comme spécifié dans l'utilisation nous voulons d'abord les options suivis des noms de fichier. Ce qui veux dire que si une options est passée après un nom de fichier celle-ci serra ignorée complètement. Aussi si une mauvaise option est passé au programme celui-ci la considéra comme un nom de fichier et tentera donc de l'ouvrir et provoquera (sûrement si aucun) une erreur d'ouverture. Si plusieurs fichiers sont données mais que l'un d'eux pose problème le programme s'arrête et provoque une erreur d'ouverture et mise a part les noms de fichiers déjà ouvert rien n'est écrit. Comme nous n'avons pas réussi à implémenter le tri d'une liste chaînée nous avons préférés ne pas ajouter l'option de tri *-s* ou *--sort*. Pour l'option de filtre si l'utilisateur rentre un filtre inexistant le programme filtre donc les caractères avec un un filtre qui ne correspond pas aux filtres disponibles, ainsi tout les caractères sont filtrés et aucun n'est écrit. Pour l'option -c le comportement est différent. Si l'utilisateur rentre un modification inexistante le problème l'ignore simplement et alors aucune modification ne serra faîte.

Benoit SAINT-ETIENNE
Tom CHAMBARETAUD

Aussi les options doivent être strictement égales, ce qui veut dire que pour *--case* il faut bien écrire *--case*, *--cas*, *--ca et --c* ne fonctionneront pas.

Benoit SAINT-ETIENNE
Tom CHAMBARETAUD