TP noté 1: Un vérificateur d'orthographe

Durée: 2h

L'objectif de ce TP est d'écrire un programme qui détecte les fautes d'orthographe dans plusieurs fichiers texte. Pour ce faire, nous allons procéder par étapes. (Les fichiers de test sont tous dans le répertoire `/users/prof/ahabibi/Public`.)

1. Écrire une fonction `creerNTubes` de profil

   ```
   int** creerNTubes(int N);
   ```

   L'objectif de cette fonction est de créer N tubes. Chaque tube est désigné par un tableau de 2 entiers. Donc pour N tubes, cette fonction devra allouer un tableau de N tableaux de 2 entiers, créer les tubes correspondants au moyen de la primitive `pipe` et retourner un pointeur vers ce tableau.

   Pour allouer un tableau où le nombre d'éléments n'est pas connu à l'avance, on peut utiliser la fonction `calloc`. (Faire `man` de cette fonction si nécessaire.)

2. Écrire en C deux fonctions de prototype :

   ```
   int nbMots(int filein);
   void litListeMots(int filein, char **mots);
   ```

   `nbMots` lit un fichier texte (de descripteur `filein`) et compte le nombre de mots qu'il contient. Les mots sont séparés par des espaces ou des symboles de ponctuation.

   `litListeMots` a comme paramètre un fichier texte (`filein`) et un tableau (`mots`) de chaînes de caractères déjà alloué. Cette fonction lit le fichier et place les mots dans le tableau en transformant les majuscules en minuscules.

   - On supposera qu'aucun mot n'a plus de 30 caractères.
   - On utilisera les fonctions `isspace`, `ispunct` et `tolower`. (Faire `man` de ces fonctions.)

   Écrire un programme `question2.c` qui affiche sur la sortie standard (à raison d'un mot par ligne) la liste des mots lus sur l'entrée standard. Tester ce programme sur le fichier `grillon.txt`.

3. Écrire en C une fonction `estCorrecte` de profil :

   ```
   int estCorrecte(char *mot, int nb_mots, char **dictionnaire);
   ```

   qui, a comme paramètre une chaîne de caractères `mot` (qui peut comporter des majuscules) et un tableau de `nb_mots` chaînes de caractères `dictionnaire` ne contenant que des minuscules. La fonction `estCorrecte` retourne 1 si `mot` appartient au dictionnaire et 0 sinon.

   - On utilisera les fonctions `strlen` et `strcmp`. (Faire `man` de ces fonctions.)

   Écrire un programme `question3.c` qui prend en argument le nom d'un fichier dictionnaire et un mot, et qui indique si le mot appartient ou non au dictionnaire. Tester votre programme avec le fichier `dictionnaire.txt`. (Il ne s'agit pas vraiment d'un dictionnaire. Il ne contient que les mots du fichier `grillon.txt`.)

4. Écrire un programme `question4.c` qui prend en argument un fichier dictionnaire et une chaîne de caractères `mot`. Ce programme crée un processus fils et deux tubes (un pour chaque sens de communication). Le fils envoie au père dans le tube, l'entier `L+1` où `L` représente le nombre de caractères de `mot`, puis le mot `mot` lui-même. Le père vérifie que `mot` appartient bien au dictionnaire. Si oui il envoie au fils l'entier 1, sinon l'entier 0. Le fils affiche un message à l'écran si `mot` est incorrect (rien sinon) et termine. Le père termine lorsque son fils a terminé.

5. Même chose que précédemment, mais avec N chaînes de caractères et N processus fils.

   Écrire un programme `question5.c` qui prend en argument le nom d'un fichier dictionnaire et N chaînes de caractères. Ce programme crée $2N$ tubes (un pour chaque fils et pour chaque sens de communication). Ensuite il crée N fils. Le n-ième fils envoie au père la n-ième chaîne de caractères. Le fils termine après avoir reçu la réponse du père (1 ou 0 selon que le mot est correct ou non) et affiché un message si le mot est incorrect. Le père lit une fois le tube de tous ses fils, y répond et attend la fin de tous ces fils avant de terminer.

6. Même chose que précédemment, mais où chaque fils envoie au père non plus un seul mot mais tous les mots d'un fichier.

   Écrire un programme `question6.c` qui prend en argument un fichier dictionnaire et N fichiers texte. Le père vérifie que les mots sont corrects et répond par 0 ou 1 selon le cas. Lorsqu'un fils a vérifié tous les mots de son fichier, il envoie au père l'entier 0 pour lui signifier qu'il a fini et termine.

   Le père tient à jour un tableau qui indique quels sont les fils qui ont fini et quels sont ceux qui n'ont pas fini. Tour à tour il lit les tubes de tous les fils qui n'ont pas fini et y répond. Le père termine lorsque tous ses fils ont fini. Tester ce programme avec les fichiers `grillon1.txt`, `grillon2.txt`, `grillon3.txt` et `grillon4.txt` dans lesquels il n'y a qu'une faute d'orthographe.