Initiation à (f)lex

Attention : quelques remarques avant de commencer.

- **Respecter la mise en page en écrivant ou recopiant des spécification lex !**

  Si quelque chose commence en première colonne, c'est qu'il doit être en première colonne. Si il y a un espace entre deux mots, c'est qu'il doit y être.

- Rajouter toujours un retour chariot à la fin de votre spécification.

  Certaines version sont très sensibles à cela, et mettent des messages d'erreur tels quels.

- L'obtention de l'executable final se fait au bout de plusieurs manipulations. Si vous en oubliez une, vous ne verrez pas l'effet de la modification de votre spécification lex.

- le programme `lex.yy.c` contient du code C, recopié tel quel, provenant de la spécification lex.

  Si vous faites une erreur de programmation C (oubli de ';' par exemple) dans la spécification lex, cette erreur ne sera détectée que dans la dernière étape.

Exercice 1

1. Écrire et compiler le fichier de spécifications lex composé des 2 lignes suivantes

   ```
   %%
   (0|1)+      {printf("\n Un nombre binaire[%s]\n",yytext);}
   ```

   Le tester avec les entrées suivantes. Essayer de prévoir ce que fera chaque entrée.

   ```
   001110
   101    0111
   012
   01201201 0000 1es CX 2500 et les DS 19 c'est gén1al
   0+0= la tete a toto.
   ```

2. Reprendre la question précédente en rajoutant la troisième ligne (le '.' de début de ligne n'est pas une erreur).

   ```
   .  // taper au moins un espace après le point.
   ```

   Quelle est la différence avec la spécification précédente ?

Exercice 2

Compiler le fichier de spécifications lex :

```
%{
int nb ;
%}
pairpair (aa|bb)*((ab|ba)(aa|bb)*(ab|ba)(aa|bb)*)*
%%
{pairpair}  {printf(" [%s] Un nombre pair de a et de b\n",yytext);}
a*b*        {printf(" [%s] D'abord des a, puis des b \n",yytext);}
[\n]        {printf("retour chariot, on ne fait rien \n ");}
.           {nb++ ; printf(" caractere '%c' ignore \n",yytext[0]);}
%%
int main(){
   nb = 0 ; yylex() ; printf ("\n On a ignore %d caracteres \n",nb) ;
}
```

• Le tester sur les entrées `babbaaab abbb aabb baabbbb bbaabbbba babbbbbab aaabbbba`. Essayer de deviner ce que va répondre le programme avant qu'il ne le fasse.
• Reprendre ce programme en permuttant les deux lignes commençant par `{pairpair}` et `a*b*`.
Sur les entrées précédentes, lesquelles sont traitées différemment, et pourquoi ?

Exercice 3 Le petit robot (1)

Nous souhaitons écrire un petit interpréteur permettant de commander un robot. Vous trouverez sur Updago le fichier `Exemple.chariot.txt`, il contient un exemple utilisant toutes les –simples – possibilités du langage de programmation de ce robot.

Le lexique utilisé par ce programme contient :

- des entiers signés ou non, des flottants à la Ada (signés ou non, avec ou sans exposant, l'exposant étant non signé ou non), des chaînes de caractères (à la C ou à la Ada), des commentaires (à la Ada) ;
- des mots clés `nom`, `debut`, `fin`, `avance`, `tourner`, `fois` et `affiche`, ces mots pouvant être écrits en utilisant indifféremment majuscules ou minuscules comme dans l'exemple ;
- le terminateur d'instruction est le caractère `';'`.

Les espaces, tabulations et sauts de ligne sont ignorés, ne servent qu'à séparer les différents mots du programme.

1. Écrire une spécification lex qui reconnaisse et affiche ces différents éléments. Sur le programme donné en exemple, il faudra afficher :
   `Commentaire Nom Chaine Pv Commentaire Avance Flottant Pv Tourner Entier Pv Fois Entier Debut ...`

   Les blancs, tabulations et espaces ne doivent pas apparaître en sortie.

   Je vous conseille de traiter et tester les lexèmes les uns après les autres, en complétant au fur et à mesure votre spécification.

2. Reprendre et modifier la spécification précédente pour qu'elle
   - supprime les commentaires,
   - pour chaque mot ou lettre n'apparaissant pas dans le lexique, il faudra afficher une erreur. Par exemple si le programme contient un `','` ou bien un mot clé mal orthografié.
   - rajoute un saut de ligne après chaque fin d'instruction,
   - remplace toute suite de blanc par un seul blanc,
   - à la fin de la lecture du fichier, il faudra afficher le nombre de constantes entières et flottantes utilisées, et confirmer qu'on a bien mis autant de `debut` que `fin` dans le fichier de commande

Exercice 4

Écrire une spécification lex qui mette un texte français au pluriel. Bien sûr c'est impossible de faire quelque chose de correct, mais l'analyseur devra au minimum transformer une phrase du genre *"dans le festival, le petit soldat et le petit cheval mangent au comptoir un pain avec un chou avec du pastis"* en *"dans les festivals, les petits soldats et les petits chevaux mangent aux comptoirs des pains avec des choux avec des pastis"*.