Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Carthage
Institut Supérieur des Technologies de l'Information et de la Communication

Licence Fondamentale Sciences de l'informatique	Classes : LGLSI-2
TP n°2 Compilation	
Objectifs	Analyse lexicale avec Flex

TP 2 Flex

Exercice 1

1. Que fait le programme flex suivant ?

```
"/*"     printf("<SE>");
"*/"     printf("<ES>");
"\\n"    printf("<ASN>");
"("|"["|"{"       printf("<PO>");
")"|"]"|"}"       printf("<PF>");
"?"+              printf("<?;%d>",yyleng);
. ;
%%
int main() {
yylex();
return 0;
}
```

2. Explorer le contenu du fichier lex.yy.c engendré par la commande flex.

Exercice 2

Écrire un programme Flex reconnaissant les ensembles de lexèmes suivants:

1. les identificateurs du C : commencent par une lettre ou un _, puis une suite de chiffres, lettres ou _, le tout avec au moins une lettre;
2. les chaînes de caractères du Pascal (‘ ... ‘ avec ” pour faire une apostrophe) ;
3. les chaînes de caractères du C;
4. les commentaires du C++ (// ...) ;
5. les entiers (décimal, hexadécimal 0x... ou 0X..., et octal 0...) ;
6. les réels avec exposant.

À chaque correspondance trouvée, votre analyseur lexical affichera en sortie le lexème reconnu et son unité lexicale. Tout caractère non reconnu sera recopié sur la sortie standard.

Exercice 3

1. Écrire une expression rationnelle lex qui définit les commentaires du C (ouverts par /* et fermés par */, potentiellement sur plusieurs lignes).

Rappel : la norme précise qu'on ne peut pas imbriquer les commentaires C les uns dans les autres.

Licence Fondamentale Sciences de l'informatique	Classes : LGLSI-2
TP n°2 Compilation	
Objectifs	Analyse lexicale avec Flex

2. Améliorer la lisibilité de la reconnaissance des commentaires en utilisant des conditions de démarrage (voir l'extrait du manuel de flex fourni).
3. Reprendre les unités lexicales chaîne Pascal et chaîne C de l'exercice précédent en utilisant les conditions de démarrage.

Exercice 4. Mots de moins de 10 lettres
Ecrire un programme flex qui extrait d'un texte tous les mots de moins de dix lettres, sans utiliser la fonction strlen ou la variable yyleng.

Exercice 5. Colonne de texte
Ecrire un programme flex qui prend en entrée un texte et qui affiche en sortie toutes les lettres de ce texte sur une colonne de 5 caractères de large. Voici ce qui doit s'afficher pour le texte "Je crois que c'est bien, oui." :
Jecro
isque
cestb
ienou

Exercice 6. Codage circulaire
Ecrire un programme flex qui remplace dans un texte chaque lettre par sa suivante en conservant la casse (a par b, B par C, z par a).
Exemple pour "Je me sens VRAIMENT bien !" :
Kf nf tfot WSBJNFOU cjfo!

Exercice 7. Repérage des noms de fonctions en C
Ecrire un programme flex qui prend un programme C et qui affiche tous les noms de fonctions utilisés dans ce programme. On fera attention à ne pas repérer abusivement les noms qui apparaissent dans les chaînes et les commentaires. Si l'on prend en entrée le programme suivant :

```
/* la fonction plus(int,int) renvoie
la somme de ses paramètres */
int plus(int a,int b) {
return a+b;
}
int main (void) {
    printf ("plus(4,7)=%d\n",plus(4,7));
    getchar(); // getchar() attend un retour chariot
return 0;
}
```

On devra obtenir la liste suivante : plus main printf plus getchar

Licence Fondamentale Sciences de l'informatique	Classes : LGLSI-2
TP n°2 Compilation	
Objectifs	Analyse lexicale avec Flex

Exercice 8. Coupures de mots

Les traitements de texte coupent parfois les mots en fin de ligne, en insérant un tiret juste avant le retour à la ligne. Ecrire un programme flex qui supprime les coupures d'un texte, en rétablissant les mots en fin de ligne. On fera attention au caractère qui suit la fin du mot, de façon à ne pas laisser d'espace en début de ligne et à ne pas envoyer une ponctuation à la ligne. Si l'on prend le texte suivant :

```
Ma grand-mère me dit sou-
vent de jouer - sagement -
à faire de la mu-
sique
pour me dis-
traire.
```

il ne doit pas devenir ceci :

```
Ma grand-mère me dit souvent
  de jouer - sagement -à faire de la musique

pour me distraire
.
```

mais ceci :

```
Ma grand-mère me dit souvent
de jouer - sagement -
à faire de la musique
pour me distraire.
```

Une condition de démarrage

Une condition de démarrage permet de créer des « mini-analyseurs » au sein de l'analyseur principal. Pour déclarer une condition de démarrage exclusive, on ajoute l'en-tête %x NOM, où NOM est le nom de la condition. Les règles dont les motifs sont préfixés par <NOM> ne seront pris en compte que si l'on a préalablement exécuté l'action BEGIN(NOM);, et tous les autres motifs sont ignorés. Pour revenir à la condition initiale, on exécute BEGIN(INITIAL);. Une condition de démarrage inclusive fonctionne de manière similaire, à la différence qu'elle doit être déclarée avec la directive %s NOM, et que les règles sans condition restent actives lorsqu'on exécute BEGIN(NOM);