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°1 Compilation	
Objectifs	Analyse lexicale avec Flex

TP 1 Flex

Dans un compilateur, **l'analyseur lexical** décompose le flux d'entrée de caractères (provenant du fichier qui contient le programme source) en une suite d'unités lexicales appelées symboles ou lexèmes (tokens); Il vérifie que chaque lexème trouvé appartient bien au lexique (dictionnaire) et retourne (éventuellement) des messages d'erreur.

L'implémentation directe d'un analyseur lexical est possible, mais, il est plus simple d'utiliser un **générateur d'analyseur lexical**. Un exemple de générateur d'analyseur lexical est **Flex** (FastLex). **Lex** est un générateur d'analyseur lexical en C (années 1970, laboratoires Bell). Il prend en entrée un ensemble d'expressions régulières et produit en sortie le texte source d'un programme C qui, une fois compilé, est l'analyseur lexical correspondant au langage défini par les expressions régulières en question.

But du TP : Le but du TP est de réaliser, avec l'aide de Flex, un analyseur lexical pour un sous-ensemble du langage C.

1. Installation de Flex sous Ubuntu

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install flex
```

2. La spécification Flex suivante reconnaît des mots commençant par une lettre, puis pouvant comporter plusieurs lettres ou chiffres. Copier cette spécification Flex dans un fichier nommé **first.flex**.

```
%%
[a-zA-Z][a-zA-Z0-9]*   printf("[ %s ]", yytext );
%%
```

3. Pour tester, on utilisera l'exemple de texte source suivant. Copier ce texte source dans un fichier nommé **first.data**.

```
for( i=0; i < 10; i++ ) {
            truc[i] = 0;
            machin  = 12.3;
            bidule  = .0;
          }
```

Licence Fondamentale Sciences de l'informatique	Classes : LGLSI-2
TP n°1 Compilation	
Objectifs	Analyse lexicale avec Flex

4. Construisez et testez cet analyseur lexical en tapant les lignes suivantes dans le terminal.

```
> flex first.flex
> gcc lex.yy.c -ll
> ./a.out < first.data
```

5. Expliquez le résultat.

Variante : Si la librairie libl.a (-ll) n'existe pas, modifier first.flex sous la forme :

```
%%
[a-zA-Z][a-zA-Z0-9]*   printf("[ %s ]", yytext );
%%
int yywrap(void) { return 1; }
int main(int argc, char *argv[]) { while (yylex()!=0) ; return 0; }
```

6. Modifiez et complétez la spécification pour que l'analyseur reconnaisse les unités lexicales suivantes :

   - Mot-clé du langage qui sont : **for**, **while**, **if**
   - Opérateurs arithmétiques : **+ * / -**
   - Opérateurs de comparaison : **< >**
   - Séparateurs : ( ) { } ; , [ ]

L'affichage distinguera les différentes unités lexicales. Pour cela, définissez une fonction **echo()** dans la spécification flex qui affichera pour chaque lexème reconnu, son unité lexicale et le lexème, c.a.d "[KeyWord:if][Sep:(]..."

7. Complétez l'analyseur précédent :

   - Ajoutez la reconnaissance de valeurs numériques entières
   - Ajoutez la reconnaissance des valeurs numériques flottantes
   - Ignorer les caractères 'espace', 'tabulation' et 'retour à la ligne'.
   - Quels sont, dans le fichier source, les caractères non reconnus ? Afficher un message d'erreur pour les caractères non reconnus.
   - Modifier votre spécification Flex afin d'éliminer les caractères non reconnus.