I61- Compilation et théorie des langages

Licence 3 - 2016/2017

Analyse lexicale avec `flex`

Le programme `flex` est un générateur automatique d'analyseur lexical. Un programme `flex` se décompose en 4 parties: prologue, définitions, règles et épilogue. Les zones du code correspondantes sont séparées dans le code source par les balises `%{`, `%}`, `%%`, `%%`.

```
%{
Prologue : declarations pour le compilateur C
%}
Definitions : definition de motifs par des exreg
%%
Regles : actions a realiser lorsqu un motif est reconnu
%%
Epilogue : corps de la fonction main()
```

1 Premier programme `flex`

1. Recopier le programme suivant qui retourne le mot le plus long d'une flot de caractères:

```
%{
  /*Prologue*/
  #include <stdio.h>
  #include <string.h>
  int longmax = 0;
  char motlepluslong[256];
%}
%option nounput
%option noinput
/*Definitions*/
BLANC    [ \t\n]
LETTRE   [a-zA-Z]
MOT      {LETTRE}+

%%

{MOT} { if (yyleng > longmax){
    longmax = yyleng;
    strcpy(motlepluslong, yytext);
    printf("\n%s",yytext);
  }
}
{BLANC}
.
%%
int main(void) {
  yylex();
  printf("\nMot le plus long: %s, de longueur: %d\n", motlepluslong, longmax);
  return 0;
```

```
}

%}
```

Pour compiler un programme flex on utilise la commande `flex prog.lex` pour produire un fichier `C lex.yy.c` lui-même compilé avec l'option `-lfl`: `gcc lex.yy.c -lfl`. Utiliser un *makefile* pour compiler le programme et l'exécuter par la commande:

`./analex < lex.yy.c`.

2. Modifier le programme pour que celui-ci retourne la ligne et la colonne où se trouve le mot le plus long.
3. Modifier le programme pour que celui-ci retourne la somme des entiers présents dans le fichiers.
4. Utiliser la variable interne gérant le flux d'entrée de `flex`

```
FILE *yyin
```

pour pouvoir passer le fichier à analyser en paramètre de la commande.

2 Gestion de symboles

Dans cette partie, il s'agit de construire un analyseur lexical pour déterminer les mots les plus fréquents dans un texte.

1. Implanter tout d'abord une structure de liste chaînées élémentaire:

```
typedef struct list_symb {
  char *symb;
  int count;
  struct list_symb * suiv;
} list_symb;

int inserer(char *nom, list_symb *ptr);
void print_list(list_symb *ptr);
```

La fonction `inserer` permet d'insérer un nouvel élément dans la liste s'il n'est pas déjà présent. Elle retourne 0 si la chaîne `nom` est déjà présente dans la liste et 1 sinon.

2. Écrire un programme `flex` pour comper la fréquence des mots dans un fichier texte.

3 Analyse de programme Python

Le but de cet partie est de programmer un outils `analysePython` dont le but est de s'assurer que les étudiants de L1 de programme pas comme des cochons. Écrire un programme `flex` qui réalise les opérations suivantes:

1. affichage du nombre de maximal de boucles imbriquées,
2. utilisation d'une fonction (ou méthode) dont le nom est passée en paramètre.

On considèrera que les programmes Python sont des scrips simples, sans définition de fonctions, classes etc. Il ne contiennent pas de caratères tabulation (''); seul le nombre d'espaces permettra de mesurer le niveau d'indentation.

Les noms de fonction respecteront la convention des noms d'identificateur Python. Un appel de fonction en python consiste en un nom d'identificateur suivi éventuellement d'espaces, suivi d'un parenthèse ouvrante.

On ne cherchera évidemment pas à vérifier si les programme sont syntaxiquement valide.

Le fichier `flex.tar` contient des exemples de fichiers types à analyser.