Ministère de l'Enseignement Supérieur, de la Recherche Scientifique et de l'Innovation (M.E.S.R.S.I)

Secrétariat Général

UniversitéNazi BONI (U.N.B.)

École Supérieure d'Informatique (E.S.I)

Master Informatique (M.I.)

Option: Systèmes d'Informations/Systèmes d'Aide à la Décision (S.I/SAD)

TRAVAUX PRATIQUES

Auteurs : **NABALOUM** Emile

Enseignant : Dr T. **BISSYANDE**

Année académique 2021-2022

Exercice 5_1

Ecrivez un programme en flex qui compte dans un texte le nombre de voyelles, consonnes, ponctuations, caractères et affiche les résultats.

Solution :

```
/* voyelle.jflex */
%%
/* ----------------- Section Options et déclarations ----------------- */
%class compte
%unicode
%line
%char
%column
%standalone
%{ /*declaration des variables*/
        public int n=0, voy=0,con=0,pon=0,car=0;

        public String input;

%}

%{eof

System.out. println ("le nombre de Voyelles est: "+(voy));

System.out. println ("le nombre de Consonnes est: "+(con));

System.out. println ("le nombre de ponctuations est: "+(pon));

System.out. println ("le nombre de caracteres est: "+(car+voy+con+pon));

%eof}/*------------------------------Déclarations de macros----------------------*/

        caracteres =[0-9]+ /*declaration de la variable caracteres*/

        voyelle=[aeiouy]+

        poncte=[!,.?:]+

        consone=[^voyelle]

        caracteres=[^ ]

        finligne = \r|\n|\r\n

        WhiteSpace = {finligne} | [ \t\f]

%%
```

```
/* ------------------------ Section des règles lexicales ---------------------- - */

/*--procurer toutes les voyelles trouver dans le fichier et incrémenter la variable voy---*

{ Voyelles } {  input=yytext(); /*----permet de récupérer tous les voyelles dans une
variables String déclarer ci haut--*/

 for(n=1; n<=input.length(); n++){ voy++;}

}


{caracteres} {  input=yytext();

for(n=1; n<=input.length(); n++){

car++;

}

}

{ ponctuations } {  input=yytext();

for(n=1; n<=input.length(); n++){pon++;}

 }

{ Consonnes } {  input=yytext();

 for(n=1; n<=input.length(); n++){con++;}

  }

{WhiteSpace} { /* Ignorer simplement ce qui a été trouvé, ne rien faire */ }
```

Cas pratique sur invite de commande :

Exercice 6_2

On donne un alphabet {a,b,c}. On veut coder a en 0, b en 01 et c en 11. On souhaite ensuite décoder une suite de 0 et de 1 provenant de codage de a en 0, b en 01 et c en 11.

Solution Codage : cde.flex

```
/*codage.jflex*/
%%
/* ----------------- Section Options et déclarations ----------------- */

%class codage
%unicode
%line
%char
%column
%standalone
%{
        public int n=0;
        public String input, put;
%}



%{eof
/*affichons le code apres lecture final de notre  fichier*/
        System.out. println ("Le codage de l'alphabet est: "+(put));
%eof}
/*-----------------------------Déclarations de macros----------------------*/

alphabet=[abc]+

%%
/* ------------------------ Section des règles lexicales ---------------------- - */

/* nous avons déclaré une variable String qui récupéré nos alphabet sous forme de
chaine, en suite nous utilisons une boucle for et une fonction replace de la variable
input pour code les alphabet présent dans la chaine input */
```

```
{alphabet} {  input=String.valueOf(yytext());
                for(n=0; n<=input.length(); n++){
                put=input.replace("a","0")
                      .replace("b","01")
                      .replace("c","11"); }
              }
```

Cas pratique sur invite de commande :

```
C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_6>Jflex codage.jflex
Reading "codage.jflex"
Constructing NFA : 8 states in NFA
Converting NFA to DFA :
...
5 states before minimization, 3 states in minimized DFA
Writing code to "decode.java"

C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_6>javac decode.java

C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_6>java decode teste.txt
Le codage du mots: abbcccccbbcbaab est:  00101111111111110101111010001
```

Solution Codage : dcde.flex

```
/* decodage.jflex */
%%
/* ----------------- Section Options et déclarations ----------------- */

%class decodage
%unicode
%line
%char
%column
%standalone

%{
public int i;
public String output,input="";

%}

%{eof
                /*affichons le code apres lecture final de notre  fichier*/

System.out.println ("Le decodage du mots: " + output + " est: "+(input));
```

```
%eof}
/*------------------------------Déclarations de macros----------------------*/

alphabet=[0-9]+

%%
/* ------------------------ Section des règles lexicales ---------------------- - */

/* nous avons déclaré une variable String qui récupéré nos alphabets sous forme de
chaine(output),*/

{alphabet} {  output =String.valueOf(yytext());

/*ensuite nous utilisons une boucle for pour parcourir la chaine stoker dans ouput*/
for(i=0;i< output.length();i++)
{
/*nous disons de faire tant avec la boucle do while, si le premier élément du fichier est 0,ensuite on
verifie si l'element suivant egale a 0, si ou stoker « a » dans input, si non si c'est 1 stoker «  b »  si non
stoker c*/
do{ if(output.charAt(i)=='0')
        { if(i!= output.length()-1)
           {   if(output.charAt(i+1)=='0')
                 {input+="a"; i+=1;} else
               if(output.charAt(i+1)=='1')
                 {input+="b";i+=2;}
         }else {input+="c";i++;}
}else
  /*Si non le premier élément du fichier est 1, ensuite on vérifie si l'élément suivant égale a 0, si ou
uoi ignore et mettre « 1 » dans output, si non si c'est 1 stoker « c » dans input*/
   if(output.charAt(i)=='1')
        {  if(i!= output.length()-1)

                {   if(output.charAt(i+1)=='0')
                      { input +="c"; i+=1;} else
                   if(output.charAt(i+1)=='1')
                      { input +="c";i+=2;}

                  }else { input +="1";i++;}
        }

}  while(i< output.length());

}

}
```

Cas pratique sur invite de commande :

```
C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_6>Jflex decodage.jflex
Reading "decodage.jflex"
Constructing NFA : 8 states in NFA
Converting NFA to DFA :
...
5 states before minimization, 3 states in minimized DFA
Writing code to "decodage.java"

C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_6>javac decodage.java

C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_6>java decodage test.txt
Le decodage du mots: 00101111111111101011101 0001 est: abbcccccbbcbaab
```

Exercice 7_3

Ecrivez un programme en flex qui vérifie le bon parenthésage d'un flot de données :

Solution :

```
/*parentheses*/
%%
%class parente
%unicode
%line
%char
%column
%standalone
%{
/*déclaration des variables à utiliser en java*/
public int n=0,np=0, np1=0,np2=0;
public String input, put;
%}
%{eof
/*ici nous vérifions si le nombre de parenthèse ouverte égale au nombre de parenthèse fermer*/
if(np1==np2){
```

```
System.out.println ("Le nombre de parenthèses est: " +(np1)+ " et sont bien former.");
}
else {System.out.println ("Les parenthèse sont mal forme");}
%eof}

/* modeles */
ouvert=[(]+
ferme=[)]+

%%
/* régles */
/*pour tout parenthèses ouvertes, les stokers dans une variable input et le parcourir et incrémenter
notre variable np1 qui au départ est égale a 0*/
{ouvert} {  input=yytext();
                        for(n=1; n<=input.length(); n++){
                        np1++;
}}

{ferme} {  input=yytext();
                        for(n=1; n<=input.length(); n++){
                        np2++;
}}
```

Cas pratique sur invite de commande :

```
C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_7>JFlex  parente.jflex
Reading "parente.jflex"
Constructing NFA : 12 states in NFA
Converting NFA to DFA :
.....
7 states before minimization, 4 states in minimized DFA
Writing code to "parente.java"

C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_7>javac parente.java

C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_7>java parente test.txt
  Le nombre de parentheses est: 2 et sont bien former.
```

Exercice 8_4

Le principe du programme est de compter le nombre de parenthéses (variable p1), crochets (variable p2) ou accolades (variables p3) dejà ouvertes.

Solution :

```
/* parentheseses.jflex*/

%%
%class parente
%unicode
%line
%char
%column
%standalone
%{
public int n=0,par,par_1,cro,cro_1,acc,acc_1;
public String input, put;
%}

%{eof
/*pour tout parenthèses ouvertes, les stokers dans une variable input et le parcourir et incrémenter notre variable np1 qui au départ est égale a 0*/

if(par==par_1){
      System.out.println("Le nombre de parentheses est: " +(par)+ " et sont bien former.");
}else {System.out.println("Le nombre de parentheses est: " +(par)+ " et sont mal former.");
}



if(cro==cro_1){
      System.out.println("Le nombre de parentheses est: " +(cro)+ " et sont bien former.");}
else {System.out.println("Le nombre de parentheses est: " +(cro)+ " et sont mal former.");}

if(acc==acc_1){
      System.out.println("Le nombre de parentheses est: " +(acc)+ " et sont bien former.");}
 else {System.out.println("Le nombre de parentheses est: " +(acc)+ " et sont mal former.");}
%eof}


/* modeles yychar */
parenth_o=[(]+
parenth_f=[)]+
crochets_o=\[+
```

```
crochets_f=\]+
accolade_O=[{]+
accolade_f=[}]+
%%
/* régles */
{parenth_o} {  input=yytext();
                    for(n=1; n<=input.length(); n++)
                    {par++;}
                }
{parenth_f} {  input=yytext();
                    for(n=1; n<=input.length(); n++)
                    {par_1++;}
                }
{crochets_o} {  input=yytext();
                    for(n=1; n<=input.length(); n++)
                    {cro++;}
                    }


{crochets_f} {  input=yytext();
                    for(n=1; n<=input.length(); n++)
                    {cro_1++;}
                    }
{accolade_O} {  input=yytext();
                    for(n=1; n<=input.length(); n++)
                    {acc++;}
                    }
{accolade_f} {  input=yytext();
                    for(n=1; n<=input.length(); n++)
                    {acc_1++;}
                    }
```

Cas pratique sur invite de commande :

```
C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_8>JFlex parenthese.jflex

Reading "parenthese.jflex"
Constructing NFA : 28 states in NFA
Converting NFA to DFA :
.............
15 states before minimization, 8 states in minimized DFA
Writing code to "parente.java"

C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_8>javac parente.java

C:\Users\HP\Desktop\TP JFlex\TP JFlex Complet\Exercice_8>java parente test.txt
Le nombre de parentheses est: 2 et sont bien former.
Le nombre de crochets est: 2 et sont bien former.
Le nombre de accolades est: 3 et sont mal former.
```

Exercice 9_5

Solution

```
import java_cup.runtime.*;

import sym;

%%

%class Lexer

%line

%column
```

```
%cup
%{
private Symbol symbol(int type) {
 return new Symbol(type, yyline, yycolumn);
}
private Symbol symbol(int type, Object value) {
 return new Symbol(type, yyline, yycolumn, value);
}
%}
/* L'espace blanc est une fin de ligne, un espace, une tabulation ou un saut de ligne. *
finligne = \r|\n|\r\n
Spacement = { finligne r} | [ \t\f]
nombre = 0 | [1-9][0-9]*

charactere = [A-Za-z_][A-Za-z_0-9]*
%%

<YYINITIAL> {
";" { return symbol(sym.SEMI); }
/* Affiche le jeton trouvé qui a été déclaré dans la classe sym puis le renvoie.*/
"+" { System.out.print(" + "); return symbol(sym.PLUS); }
"-" { System.out.print(" - "); return symbol(sym.MINUS); }
"*" { System.out.print(" * "); return symbol(sym.TIMES); }
"/" { System.out.print(" / "); return symbol(sym.DIVIDE); }
"(" { System.out.print(" ( "); return symbol(sym.LPAREN); }
")" { System.out.print(" ) "); return symbol(sym.RPAREN); }
/* Si un entier est trouvé, imprimez-le, renvoyez le jeton NUMBER qui représente un entier et la
valeur de l'entier contenu dans la chaîne yytext qui sera transformée en entier avant de revenir */
{ nombre } { System.out.print(yytext());
 return symbol(sym.NUMBER, new Integer(yytext())); }
```

```
{ charactere } { System.out.print(yytext());
 return symbol(sym.ID, new Integer(1));}

{Spacement} { }
}

. { throw new Error("caractere ilegale <"+yytext()+">");} /*gestion des exception*/
```

Exercice 10_6

```
%%
%class html
%unicode
%line
%char
%column
%standalone

%{ /*declaration de variable*/
public int n=0,par,par_1,cro,cro_1,acc,acc_1;
public String input, put;

%}

%{eof
/*afficher fin a la fin su parcours du fichier*/
System.out.println("\n fin");


%eof}

/* modeles yychar */
hypertext=http://.jpg{1}
```

```
Script=http://.js{1}
css=http://.css{1}

%%
/* régles */
{hypertext} {
/* si il Ya exactement un HyperText contenant .jpg c'est une image*/
                    System.out.printf("les images sont : %s sont:",yytext());
}




{Script} {
/* si il y a exactement un HyperText contenant .js c'est une c'est un script*/

                    System.out.printf("les scripts %s sont:",yytext());
}
{css} {

/* si il y a exactement un HyperText contenant .css c'est une c'est un lien css ou boostrap*/

                    System.out.printf("les lien %s sont:",yytext());
}
```