Devoir Maison Numero 1

Matricule	Nom	Prenom
222231485010	Chabane Chaouche	Rabah
222231501708	Foul	Rami Abdeldjalil

1 Langage

1.1 Entités Lexicales Du Langage

Entités Lexicales

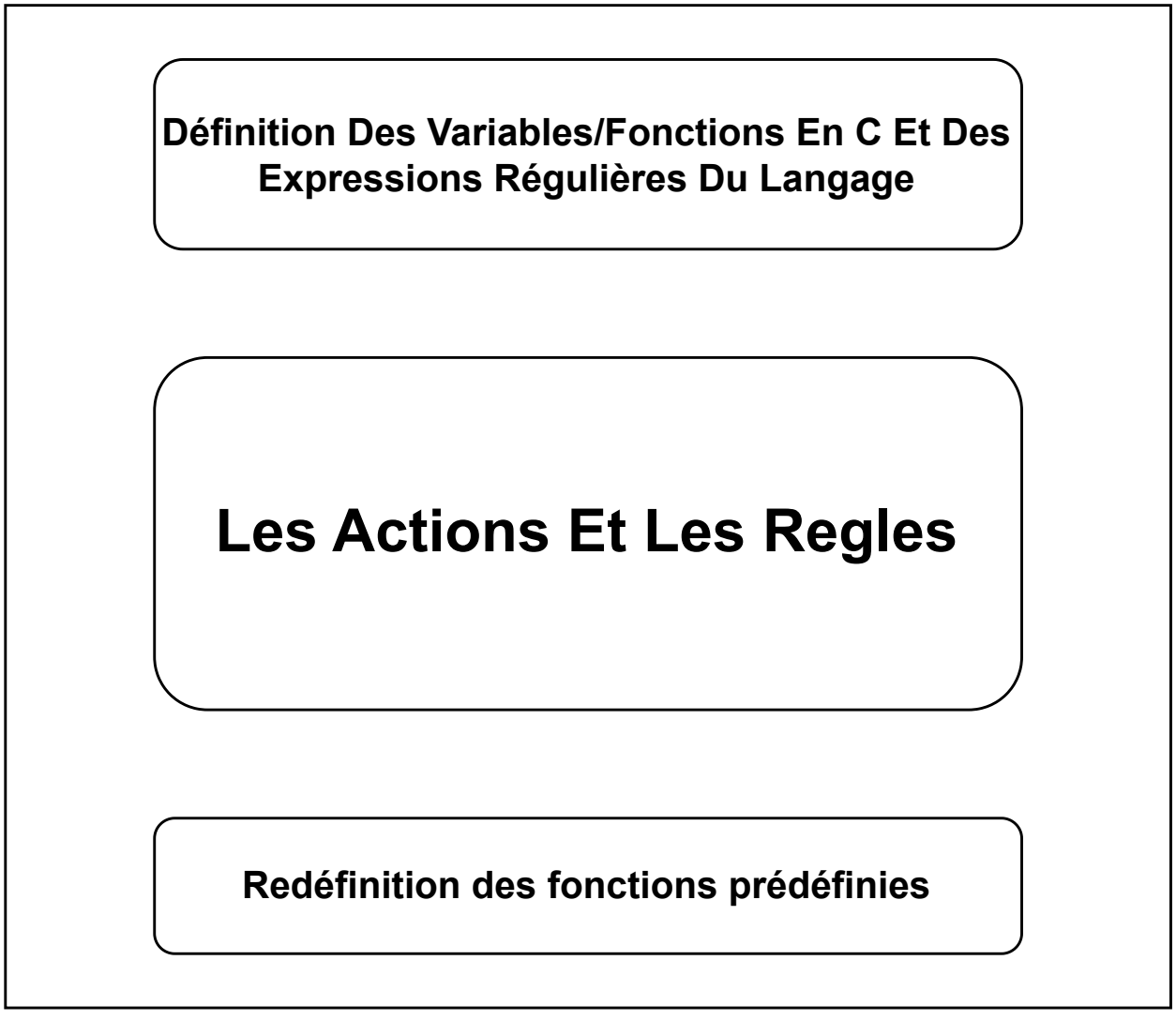
- **Identifiants** : les noms des variables , et fonctions.
- **Mots Clés** :
  - **Type Prédéfinis** : 'Entier', 'Reel'.
  - **Délimiteurs De Bloc** : 'Begin', 'end', 'Body', 'Declaration', 'MainProgram'.
- **Constantes** :
  - **Constante Entière**
  - **Constante Chaine De Caractères**
- **Séparateurs** :
  - **Opération Arithmétique** : '/', '+', '-', '*'.
  - **Parenthèses** : '(', ')'.
  - **Accolades** : '{' , '}'.
  - **Affectation** : ':='.
  - **Deux Points** : ':'.
  - **Virgule et Point-Virgule** : ',', ';'.

1.2 Commentaires

Commentaires

- **Commentaire Simple** : commentaire écrit en une ligne commençant par '##'.
- **Commentaire Multi-ligne** : commentaire qui peut être écrit sur plusieurs lignes, commence par '{ --' et finit par '-- }'.

2 Implementation En Flex



2.1 Définition Des Variables/Fonctions Des Expressions Régulières

2.1.1 Variables Et Fonctions

Variables

- **Variable Entière :** `nb_line` qu'on va incrémenter pour chaque saut de ligne `\n` initialisé a 1.
- **Bibliothèque :** `<stdio.h>` pour afficher des messages avec `printf`.
- **Fonction :** `nb_line_comment` calcule le nombre de saut de ligne contenue dans un commentaire.

Code

```
%{

#include<stdio.h>
int nb_line = 1;

 int nb_line_comment(char *comment) {
   int count = 0;
   int i = 0;

   while (comment[i]!='\0') {
     if (comment[i] == '\n') {++count;}
     ++i;
    }
 return count;
 }

%}
```

2.1.2 Expressions Régulières

Expressions Régulières

- **Word :** Correspond aux lettres de l'alphabet, en minuscules ou en majuscules, ainsi qu'au caractère underscore (_).
- **Digit :** Correspond aux chiffres de 0 à 9.
- **IDF :** Correspond à un identifiant qui commence par une lettre alphabétique, suivie d'une séquence de **Word** et de **Digit**.
- **NumberCst :** Correspond à une constante entière, soit une suite de **Digit**.
- **StringCst :** Correspond à une constante chaîne de caractères, qui commence et finit par un guillemet simple ('), et qui peut contenir n'importe quel caractère à l'exception du saut de ligne et du guillemet simple.
- **Cst :** Correspond soit à **StringCst**, soit à **NumberCst**.
- **InlineComment :** Correspond à un commentaire écrit sur une seule ligne, qui commence par '##' suivi de n'importe quel caractère, à l'exception du saut de ligne.
- **MultiLineComment :** Correspond à un commentaire sur plusieurs lignes, qui commence par '{ --' et se termine par '-- }', entre les deux, il peut y avoir n'importe quel caractère et des sauts de ligne.
- **Comment :** Correspond soit à **MultiLineComment**, soit à **InlineComment**.
- **ArithmeticOperation :** Correspond soit à '-', '+', '*', '/'.
- **Separators :** Correspond soit aux parenthèses '(', ')', aux accolades '{' et '}', au point-virgule ';', à la virgule ',', au signe d'affectation ':=' , aux deux-points ':', ou à **ArithmeticOperation**.
- **KeyWords :** Correspond soit à 'Entier' , 'Reel' , 'Begin' , 'end' , 'Body' , 'MainProgram' , 'Declaration' .

Code

```
Word [a-zA-Z_]
Digit [0-9]
IDF [a-zA-Z]({Word}|{Digit})*
NumberCst {Digit}+
StringCst '[^\n']*'
Cst {NumberCst}|{StringCst}
InlineComment ##[^\n]*
MultiLineComment "{ --"(.|\n)*"-- }"
Comment ({InlineComment}|{MultiLineComment})
ArithmeticOperation [+/*\-]
Separators (\(|\)|:=|;|\{|\}|:|,|{ArithmeticOperation})
KeyWords "Entier"|"Reel"|"MainProgram"|"Declaration"|"Body"|"Begin"|"end"
```

2.2 Les Règles

Les Règles

- **{KeyWords}** : Afficher le mot clé avec le numéro de ligne.
- **{IDF}** : Vérifier que la taille de l'identifiant n'excède pas 12 avec la fonction `yyleng` , si oui, l'afficher, sinon afficher un message d'erreur avec le numéro de ligne.
- **{Cst}** : Afficher la constante avec le numéro de ligne.
- **{Separators}** : Afficher le séparateur avec le numéro de ligne.
- **{Comment}** : Afficher le commentaire avec le numéro de ligne, et mettre à jour `nb_line` avec le nombre de sauts de ligne de l'entité grâce à la fonction `nb_line_comment`.
- **[\t]** : Ignorer les tabulations et les espaces.
- **\n** : Incrémenter la variable `nb_line` après chaque saut de ligne.
- **.** : Si l'entité lexicale ne correspond à aucune des expressions régulières définies, cela signifie que l'entité est erronée et ne fait pas partie du langage. Afficher un message d'erreur avec le numéro de ligne.

Code

```
%%

{KeyWords}   printf("Matched Keyword : %s In Line %d\n",yytext,nb_line);
{IDF} {
    if(yyleng <=12){
    printf("Matched Identifier : %s In Line %d\n",yytext,nb_line);
    }
    else {
    printf("Error Identifier : %s Exceded Maximum Length Characters In Line %d\n",yytext,nb_line);
    }
      }

{Cst}        printf("Matched Constant : %s In Line %d\n",yytext,nb_line);
{Separators} printf("Matched Separator : %s In Line %d\n",yytext,nb_line);
{Comment}   { printf("Matched Comment : %s In Line %d\n",yytext,nb_line);
              nb_line = nb_line + nb_line_comment(yytext);
              }

[ \t]
\n ++nb_line;
. {printf("Lexical Error At line %d \n",nb_line) ;}

%%
```

Remarque

- **Pourquoi `nb_line_comment` ?** : Parce que la règle qui incrémente `nb_line` ne suffit pas. Elle ne détecte que les `\n` à la fin de chaque ligne d'entité, mais pas ceux compris dans les commentaires multi-lignes.
- **{KeyWords} doit être placé avant {IDF} dans les règles** : Les mots-clés correspondent aussi à des {IDF}. Ce sont en réalité des cas particuliers de {IDF}, c'est-à-dire un sous-ensemble. Pour éviter que les mots-clés soient reconnus à la fois comme {IDF} et comme mots-clés, il faut s'assurer que {KeyWords} soit placé avant {IDF}.

2.3 Fonctions Prédéfinies

Fonctions Prédéfinies

Pour la section des fonctions prédéfinies, on la laisse telle quelle sans changer la définition des fonctions.

Code

```c
int main ()
{
    yylex () ;
    return 0;
}

int yywrap() { return 1; }
```

3 Execution

Code Bat

```
flex solution.l
gcc lex.yy.c -o dm_1
dm_1 < test.txt
```

Code Source

```
MainProgram L3_Software;
Declaration
{ -- Partie
  Declaration
-- }
L,S,K : Entier;
F,W : Reel;

Body
Begin
{
 ## Partie Instruction
  sum := a+b;
   Prod := D *F - 3 + 4 / 2;
  readln(a,b);
  writeln('Hello World!');
}
end
```

Resultat

```
C:\Users\Administrator\Downloads>run.bat

C:\Users\Administrator\Downloads>flex solution.l

C:\Users\Administrator\Downloads>gcc lex.yy.c -o dm_1

C:\Users\Administrator\Downloads>dm_1  0<test.txt
Matched Keyword : MainProgram In Line 1
Matched Identifier : L3_Software In Line 1
Matched Separator : ; In Line 1
Matched Keyword : Declaration In Line 2
Matched Comment : { -- Partie
  Declaration
-- } In Line 3
Matched Identifier : L In Line 6
Matched Separator : , In Line 6
Matched Identifier : S In Line 6
Matched Separator : , In Line 6
Matched Identifier : K In Line 6
Matched Separator : : In Line 6
Matched Keyword : Entier In Line 6
Matched Separator : ; In Line 6
Matched Identifier : F In Line 7
Matched Separator : , In Line 7
Matched Identifier : W In Line 7
Matched Separator : : In Line 7
Matched Keyword : Reel In Line 7
Matched Separator : ; In Line 7
Matched Keyword : Body In Line 9
Matched Keyword : Begin In Line 10
Matched Separator : { In Line 11
Matched Comment : ## Partie Instruction In Line 12
Matched Identifier : sum In Line 13
Matched Separator : := In Line 13
Matched Identifier : a In Line 13
Matched Separator : + In Line 13
Matched Identifier : b In Line 13
Matched Separator : ; In Line 13
Matched Identifier : Prod In Line 14
Matched Separator : := In Line 14
Matched Identifier : D In Line 14
Matched Separator : * In Line 14
Matched Identifier : F In Line 14
Matched Separator : - In Line 14
Matched Constant : 3 In Line 14
Matched Separator : + In Line 14
Matched Constant : 4 In Line 14
Matched Separator : / In Line 14
Matched Constant : 2 In Line 14
Matched Separator : ; In Line 14
Matched Identifier : readln In Line 15
Matched Separator : ( In Line 15
Matched Identifier : a In Line 15
Matched Separator : , In Line 15
Matched Identifier : b In Line 15
Matched Separator : ) In Line 15
Matched Separator : ; In Line 15
Matched Identifier : writeln In Line 16
Matched Separator : ( In Line 16
Matched Constant : 'Hello World!' In Line 16
Matched Separator : ) In Line 16
Matched Separator : ; In Line 16
Matched Separator : } In Line 17
Matched Keyword : end In Line 18

C:\Users\Administrator\Downloads>
```