Faculté des sciences exactes **TP compilation2**
Département d'informatique **Master 1ère année RSSI/WIC/ISI**

TP n°3 : analyse syntaxique avec Bison

Soit le code suivant d'un analyseur syntaxique des expressions arithmétiques en Bison qui génère la suite de dérivation pour les expressions syntaxiquement correctes

```
%{
#include <stdio.h>
%}
%token  Tnb
%start  S
%%
S : E               { printf ("1 \n");}
        | S E       { printf ("2 \n");}
        ;
E : T               { printf ("3 \n");}
      | E '+' T     { printf ("4 \n");}
      | E '-' T     { printf ("5 \n");}
      ;
T  : F                { printf ("6\n");}
      | T '*' F       { printf ("7\n");}
      | T '/' F       { printf ("8\n");}
      ;
F    : Tnb            { printf ("9\n ");}
      | '('E')'       { printf ("10\n");}
      ;
%%
#include <stdio.h>
#include "lex.yy.c"
yyerror () { printf(stderr,"Syntaxe incorrecte\n");}
main()
{
 yyparse();
}
```

Le travail demandé :

1. Enregistrer le code dans un fichier nommé cal.y
2. Compiler le fichier enregistré avec Bison à l'aide des commandes suivantes :

```
flex cal.lex            //cal.lex est  l'analyseur lexical en Flex

bison cal.y

gcc cal.tab.c –lfl –o calcul
```

3. Analyser les expressions suivantes: 8*-6, 5+2 et 6-2/2