Analyse Syntaxique : TP1

Question 1-6 : Flex

lexer.l :

```
%{
#include <stdio.h>
#include <stdlib.h>
%}

%option warn 8bit nodefault noyywrap

DIGIT    [0-9]
HEXDIGIT [0-9a-fA-F]
LETTER   [a-zA-Z]
IDENTIFIER [a-zA-Z_][a-zA-Z0-9_]*
TURTLE_ID [A-Z][A-Z0-9]*

%%

true      { fprintf(stdout, "Keyword true recognized\n"); }

0|[1-9]{DIGIT}*  { printf("Integer: %s\n", yytext); }
0[xX]{HEXDIGIT}+ { printf("Hexadecimal: %s\n", yytext); }
{DIGIT}+"."{DIGIT}*+([eE][-+]?{DIGIT}+)?|{DIGIT}+[eE][-+]?{DIGIT}+ { printf("Floating point: %s\n", yytext); }
{TURTLE_ID}      { printf("Turtle identifier: %s\n", yytext); }
{IDENTIFIER}     { printf("C identifier: %s\n", yytext); }
[\n\t ]*         /* ignore white space */
.                { fprintf(stderr, "Unknown token: '%s'\n", yytext); exit(EXIT_FAILURE); }

%%
```

```
$ echo "
      true
      0
      123
      02
      0x0
      0x29A
      0xDeadBeef
      I
      PI
      CIRCLE
      POLY5
```

```
      500.0
      5e2
      5e+2
      5e-2
      5.0e-2
      hello
      _world Hello
      "
| ./flex-test
```

```
>
Keyword true recognized
Integer: 0
Integer: 123
Integer: 0
Integer: 2
Hexadecimal: 0x0
Hexadecimal: 0x29A
Hexadecimal: 0xDeadBeef
Turtle identifier: I
Turtle identifier: PI
Turtle identifier: CIRCLE
Turtle identifier: POLY5
Floating point: 500.0
Floating point: 5e2
Floating point: 5e+2
Floating point: 5e-2
Floating point: 5.0e-2
C identifier: hello
C identifier: _world
C identifier: Hello
```

Question 7 : L'importance de l'ordre des règles

Oui, l'ordre des règles dans un fichier Flex est crucial. Flex donne la priorité à la règle qui correspond en premier. Si une entrée correspond à plusieurs règles, la première règle définie dans le fichier est utilisée. Cela signifie que les règles les plus spécifiques doivent généralement être placées avant les règles plus générales.

Question 8 : À quoi sert la règle [\n\t]* ?

Cette règle sert à ignorer les espaces blancs, y compris les espaces, les tabulations et les sauts de ligne. Elle ne correspond à aucune action spécifique,

ce qui signifie que lorsqu'un espace blanc est rencontré, il est simplement ignoré et l'analyse continue avec le prochain caractère.

```
[\n\t ]*    { /* Ignorer les espaces blancs */ }
```

Question 9 : À quoi sert la règle ‘.’ placée en dernière position ?

La règle . correspond à n'importe quel caractère autre que ceux déjà couverts par les règles précédentes. Placée en dernière position, elle sert de règle de capture générique pour tout caractère non reconnu par les autres règles. Cela peut être utilisé pour signaler une erreur ou effectuer une action par défaut, comme afficher un message d'erreur ou ignorer le caractère.

```
.    { /* action pour tout autre caractère */ }
```