The part surrounded by '{' and '}' is included as a part of the generated lexer code. It includes definitions of types, variables, and functions used in the lexer code.

```
{
(* lexerが利用する変数、関数、型などの定義 *)
open Parser
open Type
}

(* 正規表現の略記 *)
let space = [' ' '\t' '\n' '\r']
let digit = ['0'-'9']
let lower = ['a'-'z']
let upper = ['A'-'Z']

rule token = parse
| space+
    { token lexbuf }
| "(*"
    { comment lexbuf; (* ネストしたコメントのためのトリック *)
      token lexbuf }
| '('
    { LPAREN }
| ')'
    { RPAREN }
| "true"
    { BOOL(true) }
| "false"
    { BOOL(false) }
| "not"
    { NOT }
```

A rule to recognize (non-negative) integers.

```
| digit+ (* 整数を字句解析するルール (caml2html: lexer_int) *)
    { INT(int_of_string (Lexing.lexeme lexbuf)) }
| digit+ ('.' digit*)? (['e' 'E'] ['+' '-']? digit+)?
    { FLOAT(float_of_string (Lexing.lexeme lexbuf)) }
| '-' (* -.より後回しにしなくても良い? 最長一致? *)
    { MINUS }
| '+' (* +.より後回しにしなくても良い? 最長一致? *)
    { PLUS }
```


*The detail of *.mll format and ocamllex is described in 12.1-12.2 of ocaml-4.03-refman.pdf.*

```
| "-."
    { MINUS_DOT }
| "+."
    { PLUS_DOT }
| "*."
    { AST_DOT }
| "/."
    { SLASH_DOT }
| '='
    { EQUAL }
| "<>"
    { LESS_GREATER }
| "<="
    { LESS_EQUAL }
| ">="
    { GREATER_EQUAL }
| '<'
    { LESS }
| '>'
    { GREATER }
| "if"
    { IF }
| "then"
    { THEN }
| "else"
    { ELSE }
| "let"
    { LET }
| "in"
    { IN }
| "rec"
    { REC }
| ','
    { COMMA }
| '_'
    { IDENT(Id.gentmp Type.Unit) }
```

```
73  | "Array.create" (* [XX] ad hoc *)
74      { ARRAY_CREATE }
75  | '.'
76      { DOT }
77  | "<-"
78      { LESS_MINUS }
79  | ';'
80      { SEMICOLON }
81  | eof
82      { EOF }
83  | lower (digit|lower|upper|'_')* (* 他の「予約語」より後でないといけない *)
84      { IDENT(Lexing.lexeme lexbuf) }
85  | _
86      { failwith
87      (Printf.sprintf "unknown token %s near characters %d-%d"
88         (Lexing.lexeme lexbuf)
89         (Lexing.lexeme_start lexbuf)
90         (Lexing.lexeme_end lexbuf)) }
91  and comment = parse
92  | "*)"
93      { () }
94  | "(*"
95      { comment lexbuf;
96        comment lexbuf }
97  | eof
98      { Format.eprintf "warning: unterminated comment@." }
99  | _
100     { comment lexbuf }
```

Names that do not match reserved words are considered general identifiers. General rule must follow specific rules for reserved words (otherwise, reserved words are considered general terms)

Occurrence of unrecognized token is considered an error.

The following rule skips over comments "(* ... *)". This part is a bit technical. Refer to the reference of ocamllex (sections 12.1-12.2, ocaml-4.03-refman.pdf)