I the input String, i the current position in the input string, initially 0.

$$\text{true\_literal\_first\_set} = \{\texttt{t}\}$$

$$\text{false\_literal\_first\_set} = \{\texttt{f}\}$$

$$\text{recursion\_variable\_first\_set} = \{\texttt{A}\ldots\texttt{Z}\}$$

$$\text{logic\_formula\_first\_set} = \{\texttt{(}\}$$

$$\text{mu\_formula\_first\_set} = \{\texttt{m}\}$$

$$\text{nu\_formula\_first\_set} = \{\texttt{n}\}$$

$$\text{diamond\_formula\_first\_set} = \{\texttt{<}\}$$

$$\text{box\_formula\_first\_set} = \{\texttt{[}\}$$

$$\text{formula\_first\_set} = \text{union of the above}$$

$$\text{and\_operator\_first\_set} = \{\texttt{\&}\}$$

$$\text{or\_operator\_set} = \{\texttt{|}\}$$

$$\text{operator\_set} = \{\texttt{|}, \texttt{\&}\}$$

$$\text{action\_name\_first\_set} = \{\texttt{a}\ldots\texttt{z}\}$$

---

function PARSE
  SKIPWHITESPACE
  if $I[i] \in$ formula_first_set **then**
    $f =$ PARSEFORMULA
    if not end of input **then**
      throw parse exception
    else
      return f
    end if
  else
    throw parse exception
  end if
end function

```
function parseFormula
    if I[i] ∈ true_literal_first_set then
        return parseTrueLiteral
    else if I[i] ∈ false_literal_first_set then
        return parseFalseLiteral
    else if I[i] ∈ recursion_variable_first_set then
        return parseRecursionVariable
    else if I[i] ∈ logic_formula_first_set then
        return parseLogicFormula
    else if I[i] ∈ mu_formula_first_set then
        return parseMuFormula
    else if I[i] ∈ nu_formula_first_set then
        return parseNuFormula
    else if I[i] ∈ diamond_formula_first_set then
        return parseDiamondFormula
    else if I[i] ∈ box_formula_first_set then
        return parseBoxFormula
    end if
end function
```

```
function parseTrueLiteral
    expect("true")
    skipWhiteSpace
    return new true literal object
end function
```

```
function parseFalseLiteral
    expect("false")
    skipWhiteSpace
    return new false literal object
end function
```

```
function parseRecursionVariable
    n = I[i]
    i := i + 1
    skipWhiteSpace
    return new recursion variable with name n
end function
```

```
function parseLogicFormula
    expect("(")
    skipWhiteSpace
    if I[i] ∈ formula_first_set then
        f=parseFormula
    else
        throw parse exception
    end if
    if I[i] ∈ operator_first_set then
        o =parseOperator
    else
        throw parse exception
    end if
    if I[i] ∈ formula_first_set then
        g=parseFormula
    else
        throw parse exception
    end if
    expect(")")
    skipWhiteSpace
    return new logic formula object with lhs f, rhs r and operator o
end function
```

```
function parseOperator
    if I[i] ∈ and_operator_first_set then
        return parseLogicAndOperator
    else if I[i] ∈ or_operator_first_set then
        return parseLogicOrOperator
    end if
end function
```

```
function parseLogicAndOperator
    expect("&&")
    skipWhiteSpace
    return new logic and operator object
end function
```

```
function parseLogicAndOperator
    expect("||")
    skipWhiteSpace
    return new logic or operator object
end function
```

```
function parseMuFormula
    expect("mu")
    requireWhitespace
    if I[i] ∈ recursion_variable_first_set then
        r=parseRecursionVariable
    else
        throw parse exception
    end if
    expect(".")
    skipWhitespace
    if I[i] ∈ formula_first_set then
        f=parseFormula
    else
        throw parse exception
    end if
    return new mu formula object with variable r and formula f
end function
```

```
function parseNuFormula
    expect("nu")
    requireWhitespace
    if I[i] ∈ recursion_variable_first_set then
        r=parseRecursionVariable
    else
        throw parse exception
    end if
    expect(".")
    skipWhitespace
    if I[i] ∈ formula_first_set then
        f=parseFormula
    else
        throw parse exception
    end if
    return new nu formula object with variable r and formula f
end function
```

```
function parseDiamondFormula
    expect("<")
    skipWhitespace
    if I[i] ∈ action_name_first_set then
        a=parseActionName
    else
        throw parse exception
    end if
    expect(">")
    skipWhitespace
    if I[i] ∈ formula_first_set then
        f=parseFormula
    else
        throw parse exception
    end if
    return new diamond formula object with action a and formula f
end function
```

```
function parseBoxFormula
    expect("[")
    skipWhitespace
    if I[i] ∈ action_name_first_set then
        a=parseActionName
    else
        throw parse exception
    end if
    expect("]")
    skipWhitespace
    if I[i] ∈ formula_first_set then
        f=parseFormula
    else
        throw parse exception
    end if
    return new box formula object with action a and formula f
end function
```

```
function parseActionName
    n := []
    while I[i] ∈action_name_first_set do
        n := n + I[i]
        i := i + 1
    end while
    return n
end function
```

function EXPECT(e)
    if the value of e is not exactly matched in I at i **then**
        throw a parse exception.
    else
        $i := i + |e|$
    end if
end function

function SKIPWHITESPACE
    while the $I[i]$ is whitepace or comment **do** $i := i + 1$
    end while
end function

function REQUIREWHITESPACE
    if $I[i]$ is not whitespace or comment **then**
        throw new parse exception
    end if
    SKIPWHITESPACE
end function