The data structure used by LL(1) are (i) input Buffers (ii) stack (iii) parsing table. The LL(1) parser uses input buffer to store the tokens. The stack is used to hold the left sentential form. The symbols in RHS are pushed into the stack in reverse order. Thus use of stack makes it nonrecursive. The table is basically a two dimensional array. The table has row for non-terminal and column for terminals. The table can be represented as M[A, a] where A is non-terminals and a is current input symbols. The parser work as follows.

The parsing program reads top of the stack and a current input symbol. With the help of these two symbol the parsing action is determined.

```
procedure match (token t)
{
        If (lookahead = = k)
        lookahead = next_token;
        else
        error;
}
```

let us consider another example:

$S \rightarrow aSb \mid ab$

```
procedure S( )
{
        If (lookahead = = a)
        {
                match (a);
                S(a);
        If (lookahead = = b)
        {
                match (b);
        }
else
        show error;
}
        else
show error;
        If (lookahead = = $)
        {       declare success}
        else
                show error;
        }
procedure match (token t)
{
        If (lookahead = = t)
                lookahead = next_token;
        else
        error;
}
```


Predictive LL(1) Parser:

This top down parsing algorithm is of non-recursive type. In this type of parsing a table is built. For LL(1) first L means input is scanned from left to right. The second L means it used left most derivation for input string and the number (1) in the input symbol means one construction of LL(1) parser (Predictive parser)

The construction of predictive LL(1) parser is based as two very important functions and those are FIRST and FOLLOW for constructing predictive LL(1) parser we have to follow the following steps.

1. Compute the first and follow function.
2. Construct the predictive parsing table using FIRST and FOLLOW function.
3. Parse the input string with the help of predictive parsing table.

FIRST FUNCTION: First (α) is the set of terminal symbol that are first symbols appearing at RHS. In derivation of α. If $\alpha \Rightarrow \varepsilon$ then ε is the first (α). Following are the rules to compute first functions.

If $\alpha = XYZ$ then FIRST(α) is computed as follows.

FIRST (α) = FIRST (XYZ) = {X} if X is terminal
FIRST (α) = FIRST (XYZ) = FIRST (X) if X does not derive to any empty string i.e. FIRST (X) does not contain $\in$
If FIRST (X) contains $\in$ then
FIRST (α) = FIRST (XYZ) = FIRST (X) – {$\in$} $\cup$ FIRST (YZ)
FIRST (YZ) is computed in the same manner.
FIRST (YZ) = {Y} if Y is a terminal otherwise.
FIRST (YZ) = FIRST (Y) if Y does not derive to an $\in$ i.e. FIRST (Y) does not contain $\in$
If FIRST (Y) contain $\in$ then
FIRST (YZ) = FIRST (Y) – {$\in$} $\cup$ FIRST (Z)

For example consider the grammar.

$S \rightarrow ACB \mid CbB \mid Ba$
$A \rightarrow da \mid BC$
$B \rightarrow g \mid \varepsilon$
$C \rightarrow h \mid \varepsilon$

FIRST (S) = FIRST (AcB) $\cup$ FIRST (CbB) $\cup$ FIRST (Ba) ... (1)

$$\text{FIRST}(A) = \text{FIRST}(da) \cup \text{FIRST}(BC)$$
$$= \{d\} \cup \text{FIRST}(BC) \quad \text{... (2)}$$

$$\text{FIRST}(B) = \text{FIRST}(g) \cup \text{FIRST}(\varepsilon)$$
$$= \{g\} \cup \{\varepsilon\} = \{g, \varepsilon\}$$

$$\text{FIRST}(C) = \text{FIRST}(h) \cup \text{FIRST}(\varepsilon)$$
$$= \{h\} \cup \{\varepsilon\} = \{h, \varepsilon\}$$

Therefore,

$$\text{FIRST}(BC) = \text{FIRST}(B) - \{\varepsilon\} \cup \text{FIRST}(C)$$
$$= \{g, \varepsilon\} - \{\varepsilon\} \cup \{h, \varepsilon\}$$
$$= \{h, g, \varepsilon\}$$

Substituting in (2) we get,

$$\text{FIRST}(A) = \{d\} \cup \{g, h, \varepsilon\} = \{d, g, h, \varepsilon\}$$