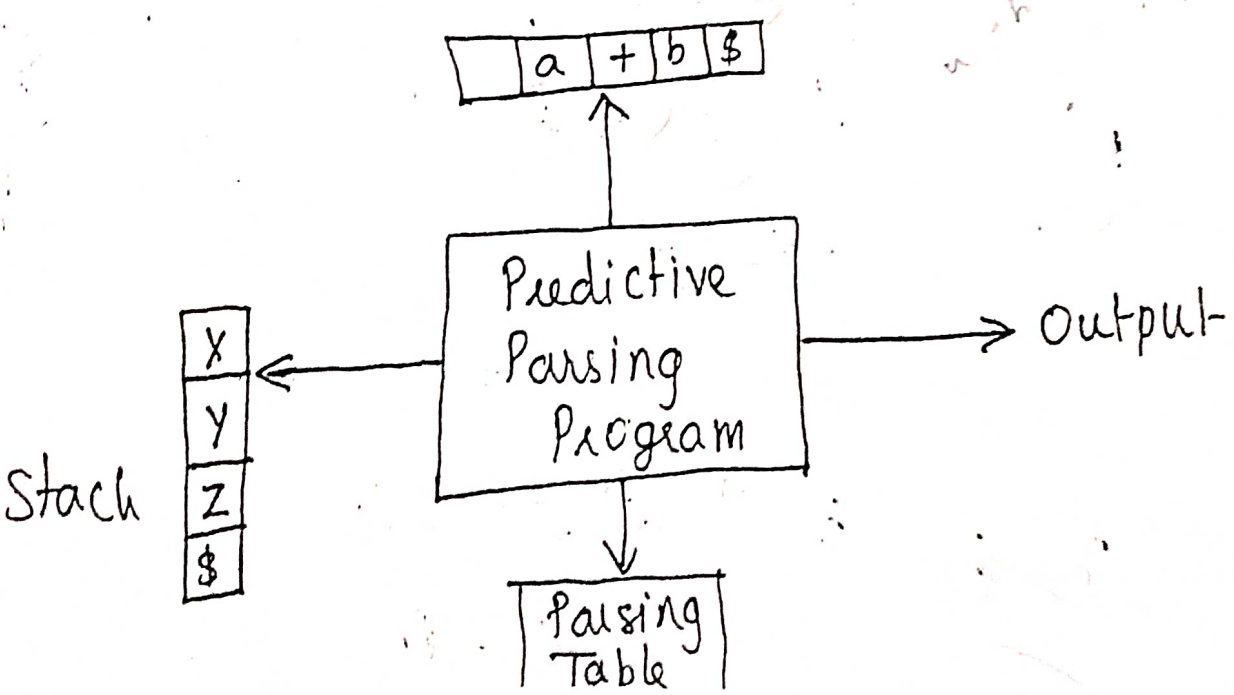
## Non-recursive Predictive Parsing:

* The key problem during predictive parsing is that of determining production to be applied for a non-terminal.
* The non-recursive parser looks up the production to be applied in a parsing table.



A table driven predictive parser has

(i) Input buffer
(ii) a stack
(iii) a parsing table
(iv) an output stream

* The input buffer contains the string to be parsed followed by $, a symbol used as a right end marker to indicate the end of the input string.
* The stack contains a sequence of grammar symbols with $ on the bottom, indicating the bottom of the stack.
* Initially stack contains a start symbol of the grammar on top of $.
* The parsing table is a two dimensional array M[A, a], where A is a non-terminal and a is a terminal or the symbol $.
* The predictive parsing program considers X, the symbol on the top of the stack and a, the current input symbol.
* These two symbols determine the action of the parser.

a) $E \rightarrow E+T \mid T$.

$A \rightarrow A\alpha \mid \beta$

$A \rightarrow \beta A'$

$A' \rightarrow \alpha A' \mid \epsilon$

1) $E \rightarrow E+T \mid T$.

$\rightarrow E \rightarrow TE'$

$\rightarrow E' \rightarrow +TE' \mid \epsilon$.

2) $+ \rightarrow T*E \mid F$

$\rightarrow T \rightarrow FT'$

$\rightarrow T' \Rightarrow *FT' \mid \epsilon$.

3) $\Rightarrow F \rightarrow (E) \mid id$.

Example:

Consider the Grammar

$E \to TE'$

$E' \to +TE' \mid \epsilon$

$T \to FT'$

$T' \to *FT' \mid \epsilon$

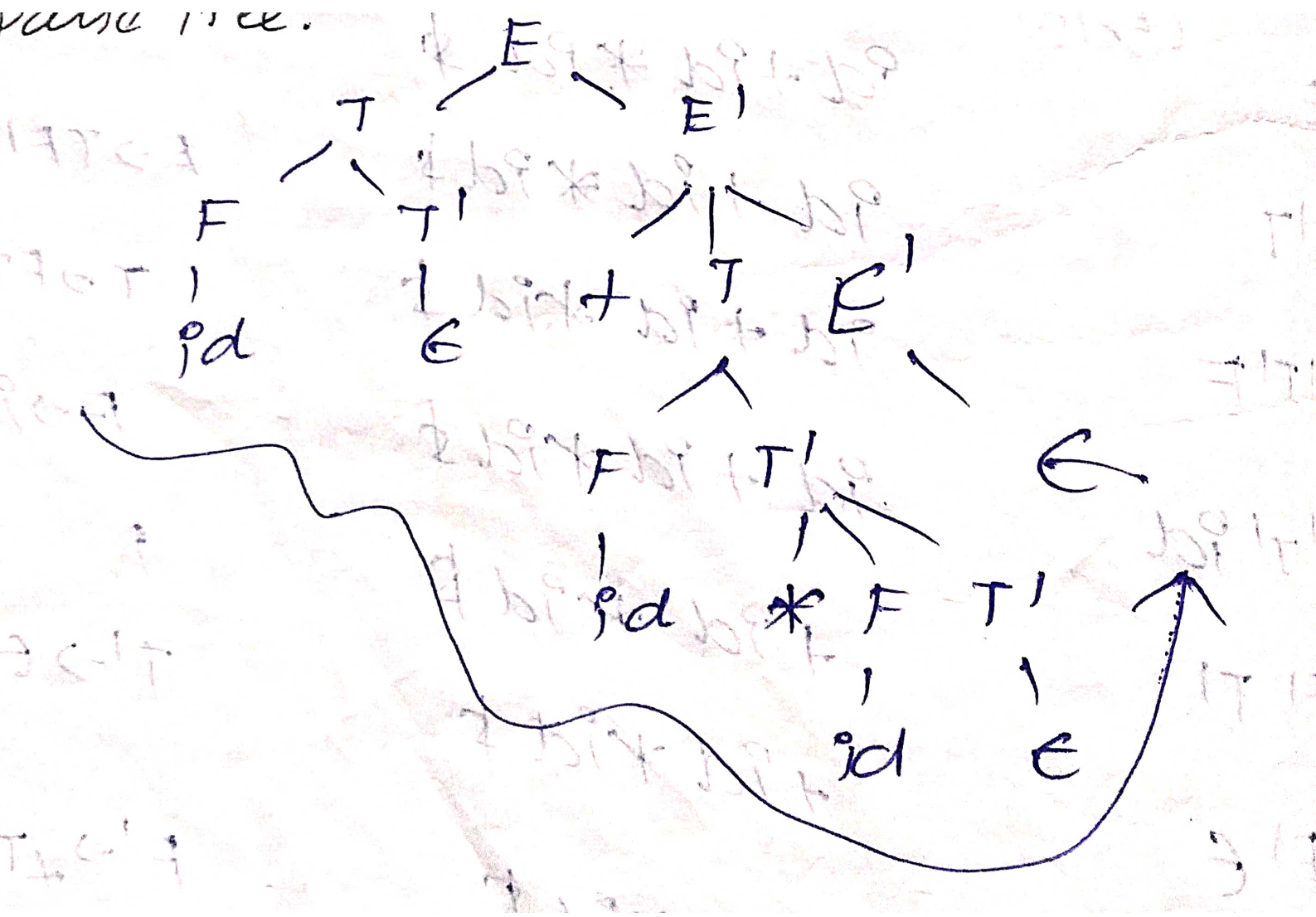
$F \to (E) \mid id.$

* In this predictive parsing table, blanks are error entries and non-blanks indicate a production with which to expand the top non-terminal on the stack.

| Non Terminal | Input Symbol | | | | | |
|---|---|---|---|---|---|---|
| | id | + | * | ( | ) | $ |
| E | $E \to TE'$ | | | $E \to TE'$ | | |
| E' | | $E' \to +TE'$ | | | $E' \to \epsilon$ | $E' \to \epsilon$ |
| T | $T \to FT'$ | | | $T \to FT'$ | | |
| T' | | $T' \to \epsilon$ | $T' \to *FT'$ | | $T' \to \epsilon$ | $T' \to \epsilon$ |
| F | $F \to id$ | | | $F \to (E)$ | | |

* The Parser traces out a left most derivation for this input. i.e) the productions output are those of a left most derivation.

| STACK | INPUT | OUTPUT |
|---|---|---|
| $E | id+id*id$ | |
| $E'T | id+id*id$ | E → TE' |
| $E'T'F | id+id*id$ | T → FT' |
| $E'T'id | id+id*id$ | F → id |
| $E'T' | +id*id$ | |
| $E' | +id*id$ | T' → ∈ |
| $E'T+ | +id*id$ | E' → +TE' |
| $E'T | id*id$ | |
| $E'T'F | id*id$ | T → FT' |
| $E'T'id | id*id$ | |
| $E'T' | *id$ | |
| $E'T'F* | *id$ | T' → *FT' |
| $E'T'F | id$ | |
| $E'T'id | id$ | F → id |
| $E'T' | $ | T' → ∈ |
| $E' | $ | E' → ∈ |


E
T
E'
F
T'
+
T
E'
id
ϵ
F
T'
ϵ
id
*
F
T'
id
ϵ

Example:

Consider the Grammar

$E \rightarrow TE'$

$E' \rightarrow +TE' \mid \epsilon$

$T \rightarrow FT'$

$T' \rightarrow *FT' \mid \epsilon$

$F \rightarrow (E) \mid id$.

Construct a predictive parsing table.

Solution:

FIRST(E) = { (, id }

FIRST(E') = { +, ε }

FIRST(T) = { (, id }

FIRST(T') = { *, ε }.

FIRST(F) = { (, id }

FIRST(+) = { + }.

FIRST(*) = { * }.

FIRST( ( ) = { ( }

FIRST( ) ) = { ) }.

FIRST(id) = { id }

* We can apply this algorithm to the grammar,

$E \rightarrow TE'$

$E' \rightarrow +TE' \mid \epsilon$

$T \rightarrow FT'$

$T' \rightarrow *FT' \mid \epsilon$

$F \rightarrow (E) \mid id$.

* According to step 2

$FIRST(TE') = FIRST(T) = \{(, id\}$

$\therefore M[T, (] = E \rightarrow TE'$

$M[T, id] = E \rightarrow TE'$

* This step 2 is applied to all the productions

$FOLLOW(E) = \{\$, )\}$

$FOLLOW(E') = \{\$, )\}$.

$FOLLOW(T) = \{), \$, +\}$

$FOLLOW(T') = \{), \$, +\}$.

$FOLLOW(F) = \{+, *, ), \$\}$.

* Applying the Algorithm, The non-recursive predictive parsing table is,

| Non Terminal | INPUT SYMBOL | | | | | |
|---|---|---|---|---|---|---|
| | id | + | * | ( | ) | $ |
| E | $E \to TE'$ | | | $E \to TE'$ | | |
| $E'$ | | $E' \to +TE'$ | | | $E' \to \epsilon$ | $E' \to \epsilon$ |
| T | $T \to FT'$ | | | $T \to FT'$ | | |
| $T'$ | | $T' \to \epsilon$ | $T' \to *FT'$ | | $T' \to \epsilon$ | $T' \to \epsilon$ |
| F | $F \to id$ | | | $F \to (E)$ | | |