Problem:

Construct the predictive parse table for the following grammar

$S \rightarrow a \mid \uparrow \mid (T)$

$T \rightarrow T, S \mid S$

Is the parser LL(1). Show the actions for the input string (a, a).

Solution:

(i) Elimination of Left recursion.

$S \rightarrow a \mid \uparrow \mid (T)$ → (1)

$T \rightarrow T, S \mid S$ → (2)

Left Recursion is there in statement (2).

Note: The form was
$A \rightarrow A\alpha \mid \beta$
changed to
$A \rightarrow \beta A'$
$A' \rightarrow \alpha A' \mid \varepsilon$

$T \rightarrow S T'$

$T' \rightarrow , S T' \mid \varepsilon$

The final statement are,

$S \rightarrow a \mid \uparrow \mid (T)$

~~$S \Rightarrow T, S$~~

$S \Rightarrow T \rightarrow S T'$

$T' \rightarrow , S T' \mid \varepsilon$

(ii.) Left Factoring:

It is not needed.

(iii.) FIRST:

FIRST(S) = { a, ↑, (}

FIRST(T) = FIRST(ST') = FIRST(S)

= { a, ↑, (}

FIRST(T') = { , , ε }.

(iv.) FOLLOW:

~~FOLLOW(T) = {) }~~

~~FOLLOW(T') = FOLLOW(T) = {) }.~~

~~FOLLOW(S) = {~~

FOLLOW(S) = FIRST(T') = { , , ε }

Take "," alone, when ε is there in FIRST(T') ~~take~~ we need to take the FOLLOW(T) also,

FOLLOW(T) = {) } from statement (1)

Now FOLLOW(S) = { , ,), $ }.

FOLLOW(T') = {) }.

Non Terminals	FIRST	FOLLOW
S	{ a, ↑, (}	{ , ,), $ }
T(T) ← 2	{ a, ↑, (}	{) }
T'	{ , , ε }	{) }

(v) Construction of Parse Table :

	a	↑	(	)	,	$
S	S→a	S→↑	S→(T)			
T	T→ST'	T→ST'	T→ST'			
T'				T'→ε	T'→ST'	

(vi) Action of the Parser for the input string (a, a).

Stack	Input	Action
$ S	(a, a)$	S → (T)
$) T (	(a, a) $	Pop and remove '('
$) T	a, a) $	T → S T'
$) T' S	a, a) $	S → a
$) T' a	a, a) $	Pop & remove 'a'
$) T'	, a) $	T' → , S T'
$) T' S ,	, a) $	Pop and remove ','
$) T' S	a) $	S → a
$) T' a	a) $	Pop and remove 'a'.
$) T'	) $	T' → ε
$)	) $	Pop and remove ')'.
$	$	Pop & remove $