Left Recursion and Left Factoring

# 

# Submitted to:

* Dr. Talha Waheed

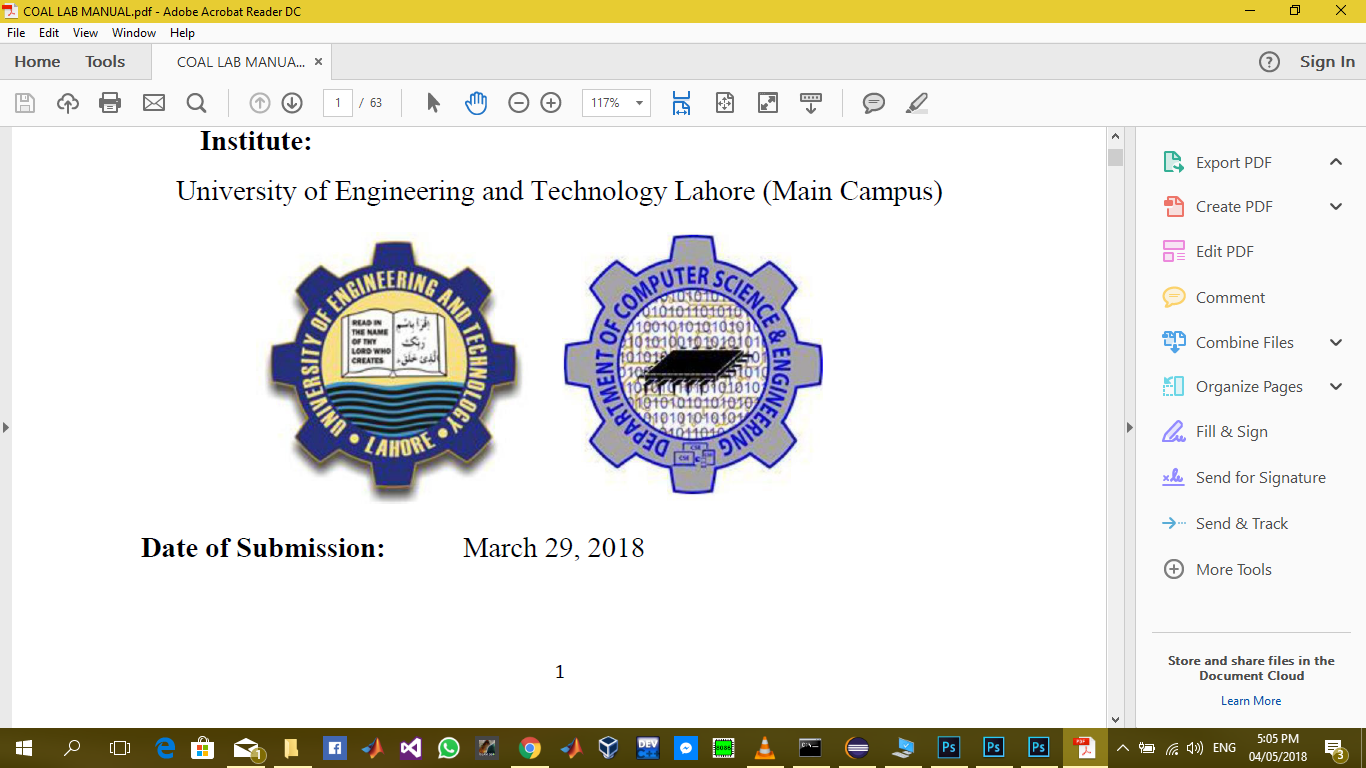
# Submitted by:

* Komal Shehzadi (2016-CS-178)

# Department of Computer Science and Engineering

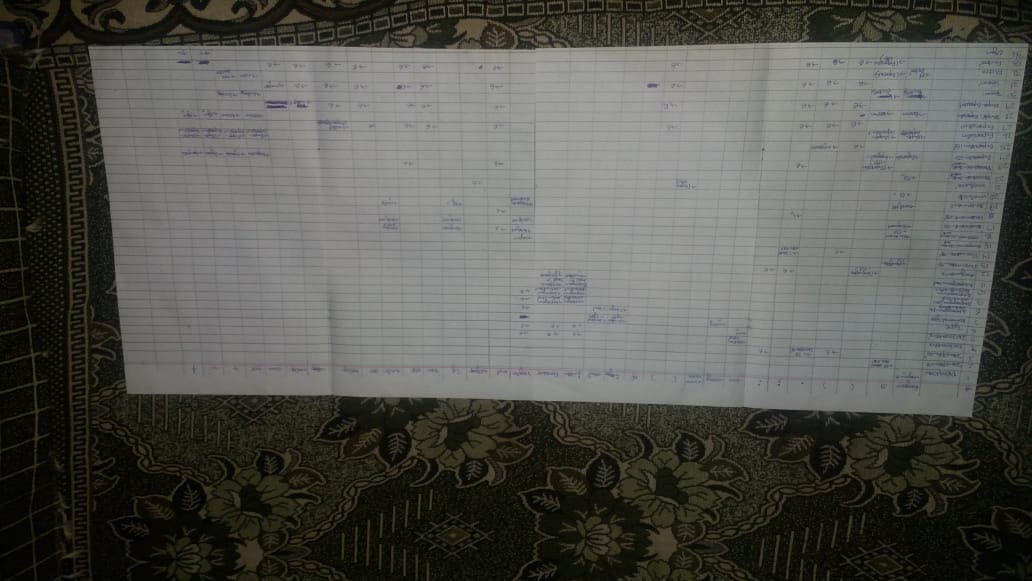
# UET Lahore



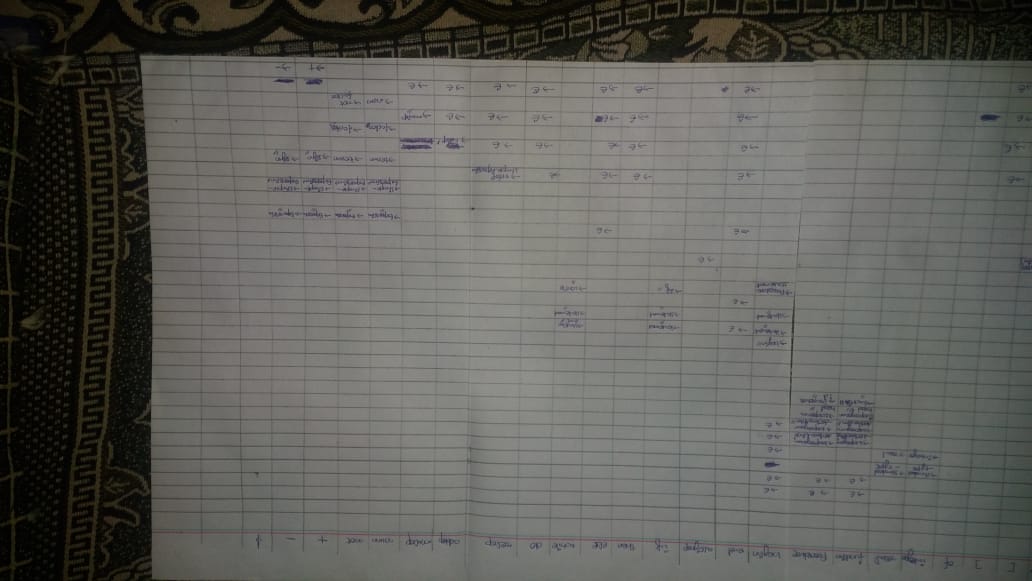


**IN MANUAL FORMAT:**

For some cases I have not wrote the complete productions in hand written assignment because of low space. Compete productions are available in word table at the end of the document.

****

****

****

**IN WORD TABLE**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **program** | **id** | **(** | **)** | **,** | **;** | **:** | **var** | **array** | **num . . num** | **[** | **]** | **of** | **integer** | **real** | **function** | **procedure** | **begin** | **end** | **assignop** | **if** | **then** | **else** | **while** | **do** | **relop** | **addop** | **mulop** | **num** | **not** | **+** | **-** | **$** |
| PROGRAM | PROGRAM -> **program id** ( identifier\_list ) ; declarations subprogram\_declarations compound\_statement |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| identifier\_list |  | identifier\_list -> **id** identifier\_list’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| identifier\_list’ |  |  |  | identifier\_list’ ->  | identifier\_list’ -> , **id** identifier\_list’ |  | identifier\_list’ ->  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| declarations |  |  |  |  |  |  |  | declarations -> declarations’ |  |  |  |  |  |  |  | declarations ->  | declarations ->  | declarations ->  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| declarations’ |  |  |  |  |  |  |  | declarations’ -> **var** identifier\_list : type ; declarations’ |  |  |  |  |  |  |  | declarations’ ->  | declarations’ ->  | declarations’ ->  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| type |  |  |  |  |  |  |  |  | type ->  **array [ num . . num ] of** standard\_type |  |  |  |  | type -> standard\_type | type -> standard\_type |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| standard\_type |  |  |  |  |  |  |  |  |  |  |  |  |  | standard\_type -> **integer** | standard\_type -> **real** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| subprogram\_declarations |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | subprogram\_declarations -> subprogram\_declarations’ | subprogram\_declarations -> subprogram\_declarations’ | subprogram\_declarations ->  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| subprogram\_declarations’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | subprogram\_declarations’ -> subprogram\_declaration ; subprogram\_declarations’ | subprogram\_declarations’ -> subprogram\_declaration ; subprogram\_declarations’ | subprogram\_declarations’ ->  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| subprogram\_declaration |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | subprogram\_declaration -> subprogram\_head declarations compound\_statement | subprogram\_declaration -> subprogram\_head declarations compound\_statement |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| subprogram\_head |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | subprogram\_head -> **function id** arguments : standard\_type | subprogram\_head ->  **procedure id** arguments ; |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| arguments |  |  | arguments -> ( parameter\_list ) |  |  | arguments ->  | arguments ->  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| parameter\_list |  | parameter\_list -> identifier\_list : type parameter\_list’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| parameter\_list’ |  |  |  | parameter\_list’ ->  |  | parameter\_list’ -> ; identifier\_list : type parameter\_list’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| compound\_statement |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | compound\_statement -> **begin** optional\_statements **end** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| optional\_statements |  | optional\_statements -> statement\_list |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | optional\_statements -> statement\_list | optional\_statements ->  |  | optional\_statements -> statement\_list |  |  | optional\_statements -> statement\_list |  |  |  |  |  |  |  |  |  |
| statement\_list |  | statement\_list -> statement statement\_list’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | statement\_list -> statement statement\_list’ |  |  | statement\_list -> statement statement\_list’ |  |  | statement\_list -> statement statement\_list’ |  |  |  |  |  |  |  |  |  |
| statement\_list’ |  |  |  |  |  | statement\_list’ -> ; statement statement\_list’ |  |  |  |  |  |  |  |  |  |  |  |  | statement\_list’ ->  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| statement |  | statement -> variable **assignop** expression |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | statement -> procedure\_statement |  |  | statement -> **if** expression **then** statement **else** |  |  | statement -> **while** expression **do** statement |  |  |  |  |  |  |  |  |  |
| variable |  | variable -> **id** variable’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| variable’ |  |  |  |  |  |  |  |  |  |  | variable’ -> [ expression ] |  |  |  |  |  |  |  |  | variable’ ->  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| procedure\_statement |  | procedure\_statement -> **id** procedure\_statement’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| procedure\_statement’ |  |  | procedure\_statement’ -> expression\_list |  |  | procedure\_statement’ ->  |  |  |  |  |  |  |  |  |  |  |  |  | procedure\_statement’ ->  |  |  |  | procedure\_statement’ ->  |  |  |  |  |  |  |  |  |  |  |
| expression\_list |  | expression\_list -> expression expression\_list’ | expression\_list -> expression expression\_list’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | expression\_list -> expression expression\_list’ | expression\_list -> expression expression\_list’ | expression\_list -> expression expression\_list’ | expression\_list -> expression expression\_list’ |  |
| expression\_list’ |  |  |  | expression\_list’ ->  | expression\_list’ -> , expression expression\_list’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| expression |  | expression -> simple\_expression expression’ | expression -> simple\_expression expression’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | expression -> simple\_expression expression’ | expression -> simple\_expression expression’ | expression -> simple\_expression expression’ | expression -> simple\_expression expression’ |  |
| expression’ |  |  |  | expression’ ->  | expression’ ->  | expression’ ->  |  |  |  |  |  | expression’ ->  |  |  |  |  |  |  | expression’ ->  |  |  | expression’ ->  | expression’ ->  |  | expression’ ->  | expression’ -> **relop** simple\_expression |  |  |  |  |  |  |  |
| simple\_expression |  | simple\_expression -> term simple\_expression’ | simple\_expression -> term simple\_expression’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | simple\_expression -> term simple\_expression’ | simple\_expression -> term simple\_expression’ | simple\_expression -> sign term simple\_expression’ | simple\_expression -> sign term simple\_expression’ |  |
| simple\_expression’ |  |  |  | simple\_expression’ ->  | simple\_expression’ ->  | simple\_expression’ ->  |  |  |  |  |  | simple\_expression’ ->  |  |  |  |  |  |  | simple\_expression’ ->  |  |  | simple\_expression’ ->  | simple\_expression’ ->  |  | simple\_expression’ ->  | simple\_expression’ ->  | simple\_expression’ -> **addop** term simple\_expression’ |  |  |  |  |  |  |
| term |  | term -> factor term’ | term -> factor term’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | term -> factor term’ | term -> factor term’ |  |  |  |
| term’ |  |  |  | term’ ->  | term’ ->  | term’ ->  |  |  |  |  |  | term’ ->  |  |  |  |  |  |  | term’ ->  |  |  | term’ ->  | term’ ->  |  | term’ ->  | term’ ->  | term’ ->  | term’ -> **mulop** factor term’ |  |  |  |  |  |
| factor |  | factor -> **id** factor’ | factor -> ( expression ) |  |  |  |  |  |  |  |  | ’ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | factor -> **num** | factor -> **not** |  |  |  |
| factor’ |  |  | factor’ -> ( expression\_list ) | factor’ ->  | factor’ ->  | factor’ ->  |  |  |  |  |  | factor’ ->  |  |  |  |  |  |  | factor’ ->  |  |  | factor’ ->  | factor’ ->  |  | factor’ ->  | factor’ ->  | factor’ ->  | factor’ ->  |  |  |  |  |  |
| sign |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | sign -> **+** | sign -> **-** |  |