Complitaion Phase 2 – Parser

**1.Eliminate left recursion and common left prefixes:**

\* PROGRAM -> Block

\* BLOCK -> **block** DEFINITIONS ;**begin** COMMANDS; **end**

\* OLD :DEFINITIONS -> DEFINITION | DEFINITIONS ; DEFINITION

NEW: DEFINITIONS -> DEFINITION | DEFINITIONS'

DEFINITIONS' ->; DEFINITION | DEFINITIONS'

DEFINITIONS' -> ᵋ

\* OLD :DEFINITION -> VAR\_DEFINITION | TYPE\_DEFINITIONS

NEW: VAR\_DEFINITIONS -> id: VAR\_DEFINITION'

VAR\_DEFINITION' -> BASIC\_TYPE

VAR\_DEFINITION' -> type\_name

\* TYPE\_DEFINITION -> **type** type\_name **is** TYPE\_INIDICATOR

\* TYPE\_INDICATOR -> BASIC\_TYPE | ARRAY\_TYPE | POINTER\_TYPE

\* BASIC\_TYPE -> **integer | real**

\* ARRAY\_TYPE -> **array** [SIZE] **of** BASIC\_TYPE

\* OLD :POINTER\_TYPE -> ^BASIC\_TYPE | ^type\_name

NEW: POINTER\_TYPE -> ^ POINTER\_TYPE'

POINTER\_TYPE'-> BASIC\_TYPE

POINTER\_TYPE'-> type\_name

\* SIZE -> int\_num

\* OLD :COMMANDS -> COMMAND | COMMANDS ; COMMAND

NEW: COMMANDS -> COMMANDS | COMMANDS'

COMMANDS'-> COMMAND | COMMANDS'

COMMANDS'-> ᵋ

\* Command -> RECEIVER = EXPRESSION |

**when**(EXPRESSION rel\_op EXPRESSION) **do** commands; **default**

COMMANDS; end**\_**when |

**for**(id=EXPRESSION;id rel\_op EXPRESSION;id++) COMMANDS;

**end\_for** |

id= **malloc**(**size\_of**(type\_name)) |

**free**(id) |

BLOCK

\* OLD :RECEIVER -> id | id[EXPRESSION] | id^

NEW: RECEIVER -> id RECEIVER'

RECEIVER'-> [EXPRESSION]

RECEIVER'-> ᵋ

RECEIVER'-> ^

\* OLD :EXPRESSION -> int\_num | real\_num | id | id[EXPRESSION] | &id | id^ |

**size\_of**(type\_name) | id ar\_op EXPRESSION

NEW: EXPRESSION -> int\_num | real\_num | id EXPRESSION' | &id |

**size\_of**(type\_name)

EXPRESSION '-> [EXPRESSION]

EXPRESSION '-> ᵋ

EXPRESSION '-> ar\_op EXPRESSION

EXPRESSION '-> ^

**2.For the obtained grammar, perform attributes Nullable, First and Follow for each of the grammar's variables**

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable** | **Nullable** | **First** | **Follow** |
| **PROGRAM** | X | {block} | {EOF} |
| **BLOCK** | X | {block} | }EOF{, ;} |
| **DEFINITIONS** | X | {id , type} | {;} |
| **DEFINITIONS'** | V | {;} | {;} |
| **DEFINITION** | X | {id , type} | {;} |
| **VAR\_DEFINITION** | X | {id} | {;} |
| **VAR\_DEFINITION'** | X | {type\_name,integer,real} | {;} |
| **TYPE\_DEFINITION** | X | {type} | {;} |
| **TYPE\_INDICATOR** | X | {integer,real,array,^} | {;} |
| **BASIC\_TYPE** | X | {ineger,real} | {;} |
| **ARRAY\_TYPE** | X | {array} | {;} |
| **POINTER\_TYPE** | X | {^} | {;} |
| **POINTER\_TYPE'** | X | {type\_name,integer,real} | {;} |
| **SIZE** | X | {int\_num} | {]} |
| **COMMANDS** | X | {id,when,for,free,block} | {;} |
| **COMMANDS'** | V | {;} | {;} |
| **COMMAND** | X | {id,when,for,free,block} | {;} |
| **RECEIVER** | X | {id } | {=} |
| **RECEIVER'** | V | { ^ , [ } | {=} |
| **EXPRESSION** | X | {int\_num,real\_num,id,&,size\_of} | {; , rel\_op , ) , ]} |
| **EXPRESSION'** | V | {^ , [ , ar\_op } | {; , rel\_op , ) , ]} |