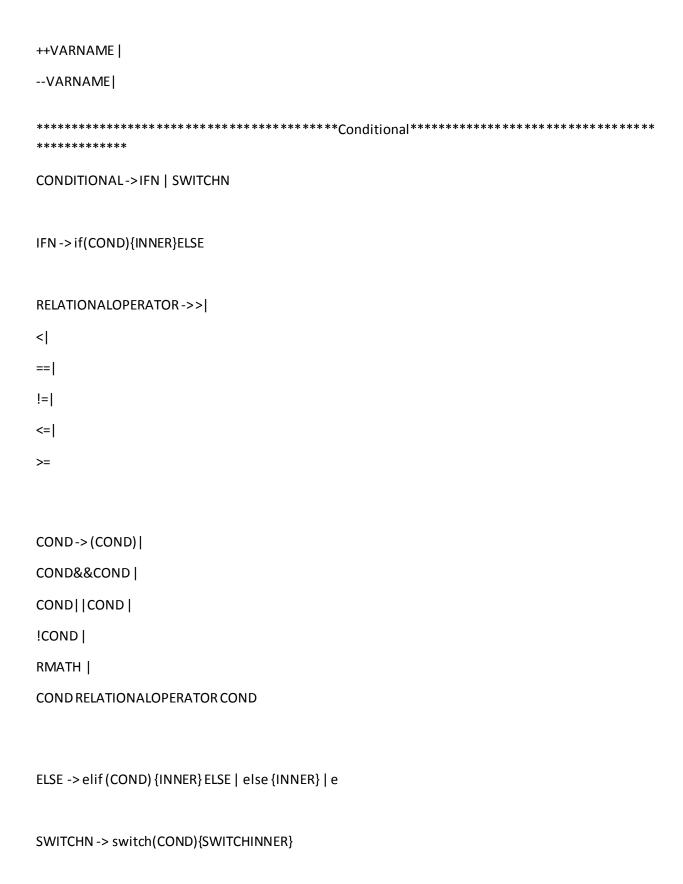


```
MVARDEF -> MVARDEF VARDEF; | e
FUNCTION -> TYPE VARNAME(FARG){INNER}
Function*****************
INNER -> COMMENT INNER |
LOOP INNER
CONDITIONALINNER |
VARDEFINNER |
STRUCT INNER |
FCALL INNER |
RETURN INNER |
;|
e |
INPUT INNER |
OUTPUT INNER |
MATH INNER |
ASSIGN INNER
**********************************Variable Names
***********
VARNAME -> *VARNAME | &VARNAME | IDENTIFIER BRACKET
BRACKET -> BRACKET[INTMATH] | e
MARG -> MARG TYPE VARNAME, | e
FARG -> MARG TYPE VARNAME
Definitions********************
VARDEF ->TYPE MVAR VARNAME; |
```

```
TYPE MVAR VARNAME = CONST; |
TYPE MVAR VARNAME = FCALL; |
TYPE MVAR VARNAME = RMATH; |
TYPE MVAR VARNAME = VARNAME; |
TYPE MVAR VARNAME = {MCONST CONST};
MVAR -> MVAR VARNAME, |
MVAR VARNAME = CONST, |
MVAR VARNAME = FCALL, |
MVAR VARNAME = RMATH, |
MVAR VARNAME = VARNAME, |
MVAR VARNAME = MCONST, |
MCONST -> MCONST CONST, | e
OPERATOR ->+
-|
*|
/|
&|
^|
\prod
******
MATH -> VARNAME = RMATH; |
;|
```

```
TYPE VARNAME = RMATH; |
VARNAME OPERATOR=RMATH; |
VARNAME++; |
VARNAME--; |
++VARNAME; |
--VARNAME;
RMATH = VARNAME
FCALL|
VARNAME++|
VARNAME-- |
++VARNAME |
--VARNAME
(RMATH)|
!RMATH |
CONST
RMATH OPERATOR RMATH
Indexing*****************
INTMATH -> VARNAME |
FCALL|
(INTMATH) |
!INTMATH |
INTCONST|
INTMATH OPERATOR INTMATH
VARNAME++|
VARNAME--|
```



```
SWITCHINNER -> case CONST:{INNER} SWITCHINNER |
case CONST: INNER SWITCHINNER |
default:{INNER} WODEFAULT|
default: INNER WODEFAULT
e
WODEFAULT -> case CONST:{INNER} WODEFAULT |
case CONST: INNER WODEFAULT
e
ARGT -> MARGT RMATH
MARGT -> MARGT RMATH, | e
FCALL -> VARNAME(ARGT)
LOOP -> FOR | WHILE
FL1-> MATH | e
FL2-> COND | e
FL3 -> VARNAME = RMATH |
TYPE VARNAME = RMATH|
VARNAME OPERATOR=RMATH |
VARNAME++|
VARNAME--|
++VARNAME|
--VARNAME
```

```
FOR -> for(FL1 FL2; FL3){INNER}
WHILE->while(COND){INNER}
DOWHILE -> do{INNER}while(COND);
MIN -> MIN >> VARNAME | e
MOUT -> MOUT <<VARNAME |
MOUT << CONST |
MOUT << WHITESPACE |
e
INPUT -> in MIN >> VARNAME;
OUTPUT -> out MOUT << VARNAME;|
out MOUT << CONST;
out MOUT << WHITESPACE;
```