CS6109 - COMPILER DESIGN – LAB

Week 8 – 05.10.2022

(Observations)

YACC Program

1. Write a yacc program to implement arithmetic operators(+ , -, *, /)

   Yacc code :

```
%{
#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
%}
%token NUM
%left '+'
%left '*'
%left '-'
%left '/'
%%
start: expr '\n' {printf("%d\n", $1);return 1; }
;
expr : expr'+'term { $$=$1 + $3;}
| expr'-'term { $$=$1-$3; }
| term { $$=$1;}
;
term : term'*'factor {$$=$1*$3;}
| term'/'factor {$$=$1/$3;}
|factor
;
factor : '('expr')' {$$=$2;}
| NUM
;
%%
yyerror(char const *s)
{
printf("yyerror %s",s);
}
```

```
int yylex() {
int c;
c=getchar();
if (isdigit(c)) {
yylval=c-'0';
return NUM;
}
return c;
}
int main(){
while(1){
yyparse();
}
return 1;
}
```

Input	**Output**
2 + 3 * 6	20
2 + 6 / 3	4
6 / 3 * 2	4
2 + 3 – 4 * 8 / 4	-3

2. Write a yacc program to implement Boolean operators (AND , OR , NOT).

Yaac code:

```
%{
#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
%}
%token NUM
%%
start: expr '\n' {if($1) printf("True\n");
else printf("False\n");
return 1; }
| rexpr '\n'
;
expr : rexpr'O''R'rexpr {$$=$1||$4;}
| rexpr'A''N''D'rexpr { $$=$1&&$5; }
| 'N''O''T'rexpr { $$=!$4;}
```

```
;
rexpr : rexpr'<'rexpr { $$=($1<$3);}
| rexpr'>'rexpr { $$=($1>$3);}
| '!'rexpr { $$=(!$2);}
| rexpr'=''='rexpr { $$= ($1==$4);}
|rexpr'!''='rexpr {$$= ($1!=$4);}
|'('rexpr')' {$$=$2;}
| NUM
;
%%
yyerror(char const *s)
{
printf("yyerror");
}
int yylex() {
int c;
c=getchar();
if (isdigit(c)) {
yylval=c-'0';
return NUM;
}
return c;
}
int main(){
while(1){
yyparse();
}
return 1;
}
```

Input	**Output**
3 == 4 OR 3 == 3	T
3! = 4 AND 3! == 3	F
2 < 3 AND 3 >4	F
NOT 0	T
NOT 1	F