UCS 1602 - Compiler Design

Exercise 5 - Implementation of Desk Calculator using Yacc Tool

Name:	Mahesh Bharadwaj K
Reg No:	185001089
Semester:	VI
Date:	March 11, 2021

Aim:

To implement desk calculator using YACC tool

Program

Lexer file

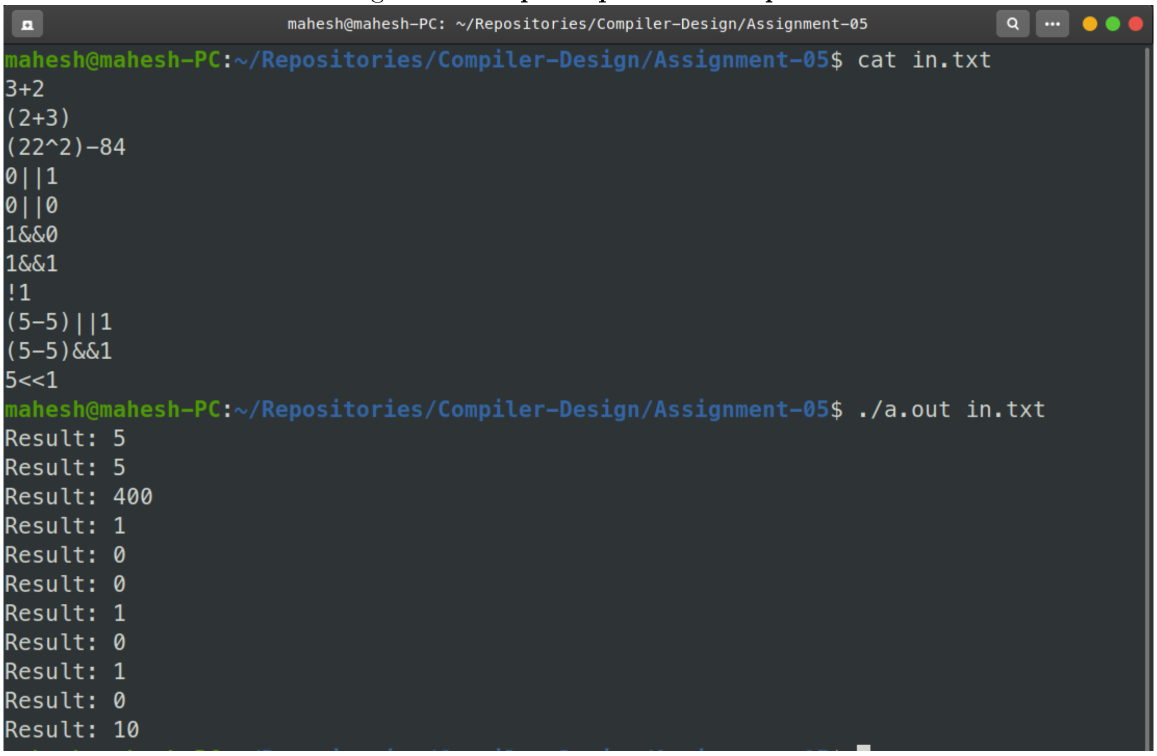
```
%{
    #include<stdio.h>
    #include<stdlib.h>
    #include<string.h>
    #include "y.tab.h"
    void yyerror(char*);
    extern int yylval;
%}
%%
[ \t]+ ;
[0-9]+ {
  yylval = atoi(yytext);
  return INT;
}

[+\-^*/] {return *yytext;}
"(" { return *yytext; }
")" { return *yytext; }
"||" { return OR; }
"&&" { return AND; }
"!"  { return NOT; }
"<<" { return LSHIFT; }
">>" { return RSHIFT; }
\n { return *yytext; }

. {
    char err[25] = {0};
    sprintf(err, "Invalid Character \"%s\"\n", yytext);
    yyerror(err);
}
%%
```

Yacc Program

```
%{
    #include <stdlib.h>
    #include <stdio.h>
    int yylex(void);
    extern FILE* yyin;
    #include "y.tab.h"

    int power(int a, int b){
        int prod = 1;
        for(int i = 0;i< b;i++)
            prod*=a;
        return prod;
  }

%}

%token INT AND OR NOT LSHIFT RSHIFT

%%
program : line program
        | line

line    : expr '\n' {printf("Result: %d\n", $1); }

expr    : expr '+' mulexpr { $$ = $1 + $3; }
        | expr '-' mulexpr { $$ = $1 - $3; }
        | mulexpr { $$ = $1;}

mulexpr : mulexpr '*' powexpr { $$ = $1 * $3; }
        | mulexpr '/' powexpr { $$ = $1 / $3; }
        | powexpr { $$ = $1; }

powexpr : powexpr '^' boolexpr {$$ = power($1, $3);}
        | boolexpr { $$ = $1;}

boolexpr: boolexpr AND bitexpr {$$ = ($1 & $3)? 1:0;}
        | boolexpr OR bitexpr {$$ = ($1 | $3)? 1:0;}
        | NOT boolexpr {$$ = (!$2)? 1:0;}
        | bitexpr {$$ = $1;}

bitexpr : bitexpr LSHIFT term {$$ = $1 << $3;}
        | bitexpr RSHIFT term {$$ = $1 >> $3;}
        | term {$$ = $1;}

term    : '(' expr ')' { $$ = $2; }
        | INT { $$ = $1; }
%%
int yyerror(char* str){
    fprintf(stderr, "%s\n", str);
    return 0;
}
int yywrap(){
    return 1;
}

int main(int argc, char **argv){
    if(argc != 2){
        fprintf(stderr, "Enter file name as argument!\n");
        return 1;
    }
    yyin = fopen(argv[1], "rt");
    if (!yyin){
        fprintf(stderr, "File not found!\n");
        return 2;
    }
    yyparse();
    return 0;
}
```

Output

Figure 1: **Sample Input and Output**



```
mahesh@mahesh-PC:~/Repositories/Compiler-Design/Assignment-05$ cat in.txt
3+2
(2+3)
(22^2)-84
0||1
0||0
1&&0
1&&1
!1
(5-5)||1
(5-5)&&1
5<<1
mahesh@mahesh-PC:~/Repositories/Compiler-Design/Assignment-05$ ./a.out in.txt
Result: 5
Result: 5
Result: 400
Result: 1
Result: 0
Result: 0
Result: 1
Result: 0
Result: 1
Result: 0
Result: 10
```

Learning Outcomes

1. We learn to write grammar for expressions.
2. We learn to Write rules to parse tokens in grammar.
3. We learn to create desk calculator using yacc tool.
