Language Translator

IT-009 Kishan Bhingradiya 17ITUBS050

IT-010 Devanshu Brahmbhatt 17ITUBS050

Language: Infix to Postfix expression

Features:

1. While humans mostly use infix notation of algebraic expressions, Reverse Polish notation or postfix notation is much easier to parse algorithmically.

2. When you are implementing your very first expression parser, postfix and prefix notations are the best way to go. Onto stack: Push, push, pop, push, push, pop. Pop.

Lexical Analyzer Source Code:

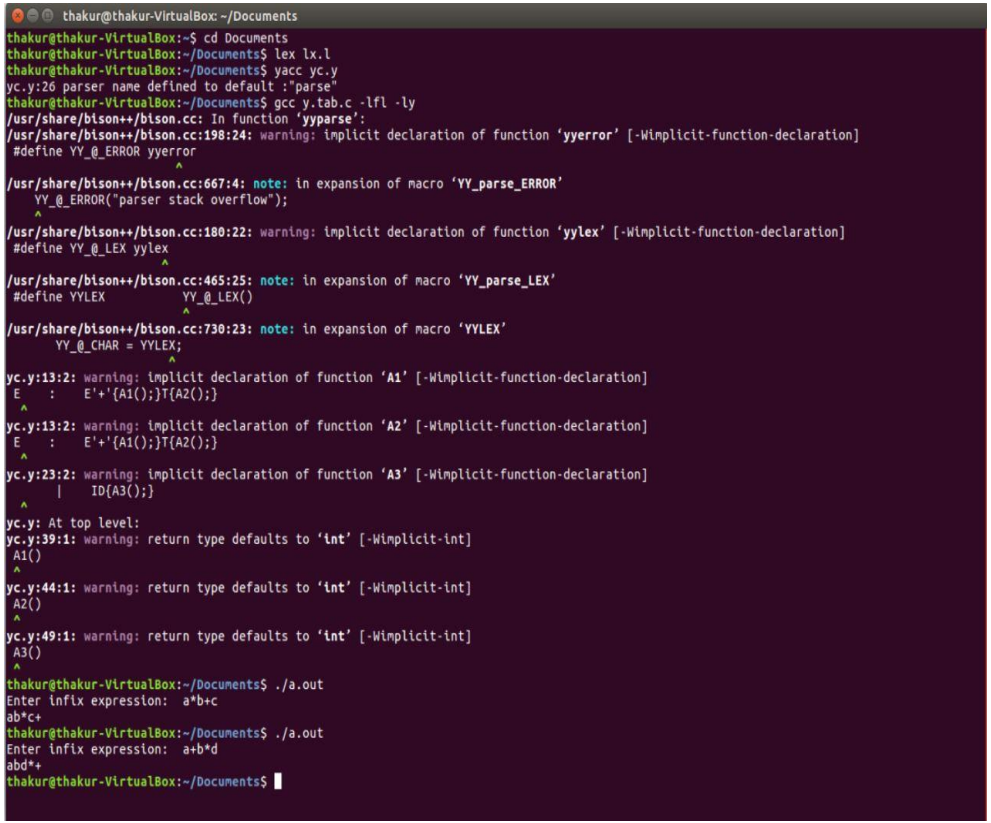
```
%{
  /* Definition section */
%}
ALPHA [A-Z a-z]
DIGIT [0-9]

/* Rule Section */
%%
{ALPHA}({ALPHA}|{DIGIT})*  return ID;
{DIGIT}+                   {yylval=atoi(yytext); return ID;}
[\n \t]                    yyterminate();
.                          return yytext[0];
%%
```

Parser Source Code:

```
%{
/* Definition section */
#include <stdio.h>
#include <stdlib.h>
%}

%token ID
%left '+' '-'
%left '*' '/'
%left UMINUS

/* Rule Section */
%%

S : E
E : E'+'{A1();}T{A2();}
| E'-'{A1();}T{A2();}
| T
;
T : T'*'{A1();}F{A2();}
| T'/'{A1();}F{A2();}
| F
;
F : '('E{A2();}')'
| '-'{A1();}F{A2();}
| ID{A3();}
```

```
ID{A3();};
%%
#include"lex.yy.c"
char st[100];
int top=0;
//driver code
int main()
{printf("Enter infix expression:  ");
  yyparse();
 printf("\n");
  return 0;
}
A1()
{st[top++]=yytext[0];}

A2()
{printf("%c", st[--top]);}

A3()
{printf("%c", yytext[0]);
}
```

Output:


thakur@thakur-VirtualBox: ~/Documents
thakur@thakur-VirtualBox:~$ cd Documents
thakur@thakur-VirtualBox:~/Documents$ lex lx.l
thakur@thakur-VirtualBox:~/Documents$ yacc yc.y
yc.y:26 parser name defined to default :"parse"
thakur@thakur-VirtualBox:~/Documents$ gcc y.tab.c -lfl -ly
/usr/share/bison++/bison.cc: In function 'yyparse':
/usr/share/bison++/bison.cc:198:24: warning: implicit declaration of function 'yyerror' [-Wimplicit-function-declaration]
 #define YY_@_ERROR yyerror
/usr/share/bison++/bison.cc:667:4: note: in expansion of macro 'YY_parse_ERROR'
    YY_@_ERROR("parser stack overflow");
/usr/share/bison++/bison.cc:180:22: warning: implicit declaration of function 'yylex' [-Wimplicit-function-declaration]
 #define YY_@_LEX yylex
/usr/share/bison++/bison.cc:465:25: note: in expansion of macro 'YY_parse_LEX'
 #define YYLEX   YY_@_LEX()
/usr/share/bison++/bison.cc:730:23: note: in expansion of macro 'YYLEX'
       YY_@_CHAR = YYLEX;
yc.y:13:2: warning: implicit declaration of function 'A1' [-Wimplicit-function-declaration]
 E   :   E'+'{A1();}T{A2();}
yc.y:13:2: warning: implicit declaration of function 'A2' [-Wimplicit-function-declaration]
 E   :   E'+'{A1();}T{A2();}
yc.y:23:2: warning: implicit declaration of function 'A3' [-Wimplicit-function-declaration]
      | ID{A3();}
yc.y: At top level:
yc.y:39:1: warning: return type defaults to 'int' [-Wimplicit-int]
 A1()
yc.y:44:1: warning: return type defaults to 'int' [-Wimplicit-int]
 A2()
yc.y:49:1: warning: return type defaults to 'int' [-Wimplicit-int]
 A3()
thakur@thakur-VirtualBox:~/Documents$ ./a.out
Enter infix expression: a*b+c
ab*c+
thakur@thakur-VirtualBox:~/Documents$ ./a.out
Enter infix expression: a+b*d
abd*+
thakur@thakur-VirtualBox:~/Documents$