Experiment XIV

**Aim**: Implement intermediate code generation for simple expressions.

Algorithm

1. Start
2. The lexical analyzer parses the input one character at a time
   1. If it is a digit or a lowercase alphabet, it is passed to the yacc file
   2. If it is a newline character, it passes 0 to the yacc file.
3. The context free grammar corresponding the the conversion is given as   
   S -> id = E  
   E -> E + T | E - T | T  
   T -> T \* F | T / F | T ^ F | F  
   F -> id | digit | (E) | -F
4. If the expression does not satisfy the context free grammar, it exits, else the intermediate code is generated.
5. The intermediate code is obtained as output.
6. Stop

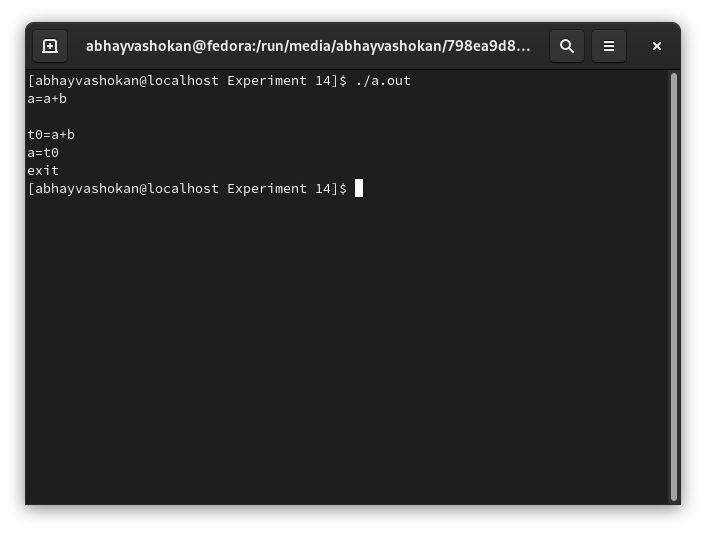
Output

a=a+b

t0=a+b

a=t0

exit

Screenshot

Readme

1. Compile the lex and yacc file using the command

**lex 2Abhay-P14.l && yacc 2Abhay-P14.y -d**

2.Compile and run the C files generated using the command  
 **gcc y.tab.c && ./a.out**

3. Enter an expression as input

4. The intermediate code is obtained as output.

**Result**: Successfully implemented a program to generate intermediate codes for simple expressions.