EXPERIMENT 3C

AIM

Convert the BNF rules into YACC form and write code to generate abstract syntax trees.

ALGORITHM

1. Start
2. Read the input file
3. Convert it into abstract syntax tree using three address code
4. Represent three address code in tabular form
5. Stop

OUTPUT

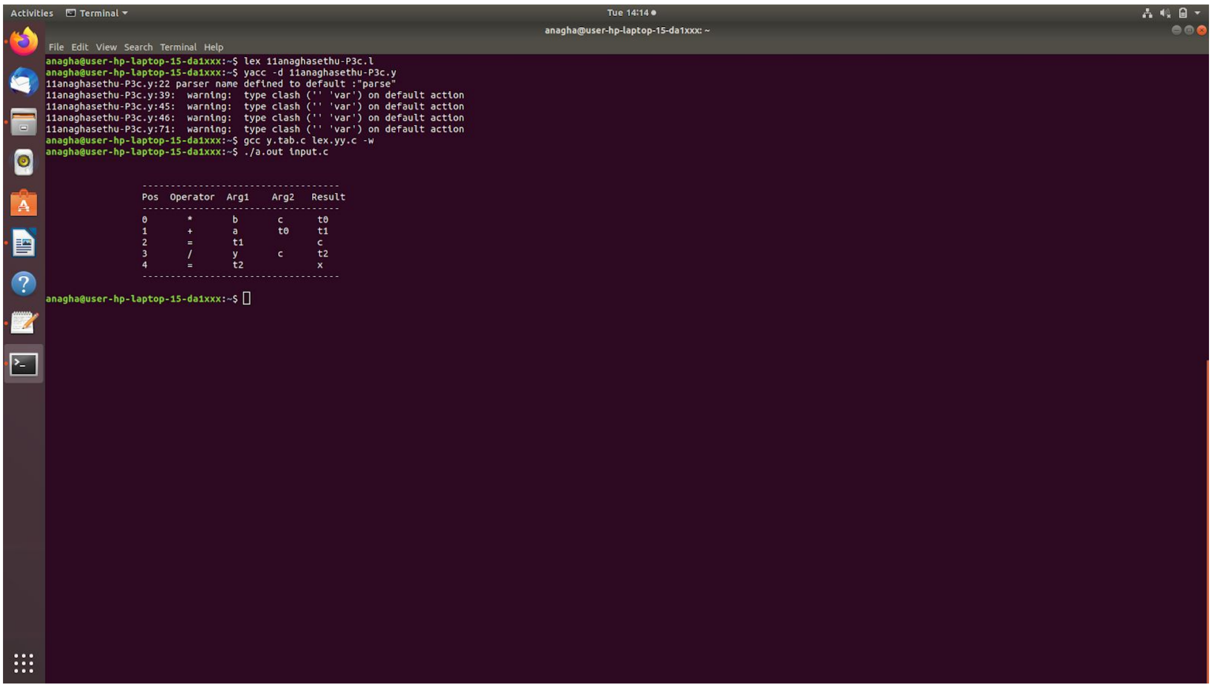
```
lex 11anaghasethu-p3c.l
```

```
yacc -d 11anaghasethu-p3c.y
```

```
gcc y.tab.c lex.yy.c
```

```
./a.out
```

Screenshot


Activities
Terminal
Tue 14:14
anagha@user-hp-laptop-15-da1xxx: ~
File Edit View Search Terminal Help

```
anagha@user-hp-laptop-15-da1xxx:~$ lex 11anaghasethu-P3c.l
anagha@user-hp-laptop-15-da1xxx:~$ yacc -d 11anaghasethu-P3c.y
11anaghasethu-P3c.y:22 parser name defined to default :"parse"
11anaghasethu-P3c.y:39:  warning:  type clash ('' 'var') on default action
11anaghasethu-P3c.y:45:  warning:  type clash ('' 'var') on default action
11anaghasethu-P3c.y:46:  warning:  type clash ('' 'var') on default action
11anaghasethu-P3c.y:71:  warning:  type clash ('' 'var') on default action
anagha@user-hp-laptop-15-da1xxx:~$ gcc y.tab.c lex.yy.c -w
anagha@user-hp-laptop-15-da1xxx:~$ ./a.out input.c


                 -----------------------------------
                 Pos  Operator  Arg1    Arg2   Result
                 -----------------------------------
                 0       *       b       c      t0
                 1       +       a       t0     t1
                 2       =       t1             c
                 3       /       y       c      t2
                 4       =       t2             x
                 -----------------------------------

anagha@user-hp-laptop-15-da1xxx:~$ 
```