190905514 MOHAMMAD TOFIK BATCH : C3
ROLL NO : 62 SECTION : C SEM : FIFTH

-: CD–LAB-3 :-

EXCERCISE :

1. Design alexical analyzer which containsgetNextToken( ) for a simple C program to create a structure of token each time and return, which includes row number, column number and token type. The tokens to be identified are arithmetic operators,relational operators, logical operators,special symbols, keywords, numerical constants, string literalsand identifiers. Also, getNextToken() should ignore all the tokens when encountered inside single line or multiline comment or inside string literal. Preprocessor directive should also be stripped.

pgm1.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define FILEINPUT "input.c"

struct token
{
char lexeme[64];
int row, col;
char type[20];
};

static int row = 1, col = 1;
char buf[2048];

const char specialsymbols[] = {'?', ';', ':', ','};
const char *keywords[] = {"const", "char", "int", "return", "for",
"while", "do",
"switch", "if", "else", "unsigned", "case", "break"};

const char arithmeticsymbols[] = {'*'};
```

```
int isKeyword(const char *str)
{
for (int i = 0; i < sizeof(keywords) / sizeof(char *); i++)
{
if (strcmp(str, keywords[i]) == 0)
{
return 1;
}
}
return 0;
}

int charBelongsTo(int c, const char *arr)
{
int len;

if (arr == specialsymbols)
{
len = sizeof(specialsymbols) / sizeof(char);
}

else if (arr == arithmeticsymbols)
{
len = sizeof(arithmeticsymbols) / sizeof(char);
}

for (int i = 0; i < len; i++)
{
if (c == arr[i])
{
return 1;
}
}

return 0;
}

void fillToken(struct token *tkn, char c, int row, int col, char *type)
{
tkn->row = row;
```

```
tkn->col = col;
strcpy(tkn->type, type);
tkn->lexeme[0] = c;
tkn->lexeme[1] = '\0';
}

void newLine()
{
++row;
col = 1;
}

struct token getNextToken(FILE *f1)
{
int c;
struct token tkn =
{
.row = -1};

int gotToken = 0;

while (!gotToken && (c = fgetc(f1)) != EOF)
{
if (charBelongsTo(c, specialsymbols))
{
fillToken(&tkn, c, row, col, "SS");
gotToken = 1;
++col;
}

else if (charBelongsTo(c, arithmeticsymbols))
{
fillToken(&tkn, c, row, col, "ARITHMETIC OPERATOR");
gotToken = 1;
++col;
}

else if (c == '(')
{
fillToken(&tkn, c, row, col, "LB");
```

```
gotToken = 1;
++col;
}

else if (c == ')')
{
fillToken(&tkn, c, row, col, "RB");
gotToken = 1;
++col;
}

else if (c == '{')
{
fillToken(&tkn, c, row, col, "LC");
gotToken = 1;
++col;
}

else if (c == '}')
{
fillToken(&tkn, c, row, col, "RC");
gotToken = 1;
++col;
}

else if (c == '+')
{
int d = fgetc(f1);

if (d != '+')
{
fillToken(&tkn, c, row, col, "ARITHMETIC OPERATOR");
gotToken = 1;
++col;
fseek(f1, -1, SEEK_CUR);
}

else
{
fillToken(&tkn, c, row, col, "UNARY OPERATOR");
```

```
strcpy(tkn.lexeme, "++");
gotToken = 1;
col += 2;
}
}

else if (c == '-')
{
int d = fgetc(f1);

if (d != '-')
{
fillToken(&tkn, c, row, col, "ARITHMETIC OPERATOR");
gotToken = 1;
++col;
fseek(f1, -1, SEEK_CUR);
}

else
{
fillToken(&tkn, c, row, col, "UNARY OPERATOR");
strcpy(tkn.lexeme, "--");
gotToken = 1;
col += 2;
}
}

else if (c == '=')
{
int d = fgetc(f1);

if (d != '=')
{
fillToken(&tkn, c, row, col, "ASSIGNMENT OPERATOR");
gotToken = 1;
++col;
fseek(f1, -1, SEEK_CUR);
}

else
```

```
{
fillToken(&tkn, c, row, col, "RELATIONAL OPERATOR");
strcpy(tkn.lexeme, "==");
gotToken = 1;
col += 2;
}
}

else if (isdigit(c))
{
tkn.row = row;
tkn.col = col++;
tkn.lexeme[0] = c;

int k = 1;

while ((c = fgetc(f1)) != EOF && isdigit(c))
{
tkn.lexeme[k++] = c;
col++;
}

tkn.lexeme[k] = '\0';
strcpy(tkn.type, "NUMBER");
gotToken = 1;
fseek(f1, -1, SEEK_CUR);
}

else if (c == '#')
{
while ((c = fgetc(f1)) != EOF && c != '\n')
;
newLine();
}

else if (c == '\n')
{
newLine();
c = fgetc(f1);
```

```
if (c == '#')
{
while ((c = fgetc(f1)) != EOF && c != '\n')
;
newLine();
}

else if (c != EOF)
{
fseek(f1, -1, SEEK_CUR);
}
}

else if (isspace(c))
{
++col;
}

else if (isalpha(c) || c == '_')
{
tkn.row = row;
tkn.col = col++;
tkn.lexeme[0] = c;
int k = 1;

while ((c = fgetc(f1)) != EOF && isalnum(c))
{
tkn.lexeme[k++] = c;
++col;
}

tkn.lexeme[k] = '\0';

if (isKeyword(tkn.lexeme))
{
strcpy(tkn.type, "KEYWORD");
}

else
{
```

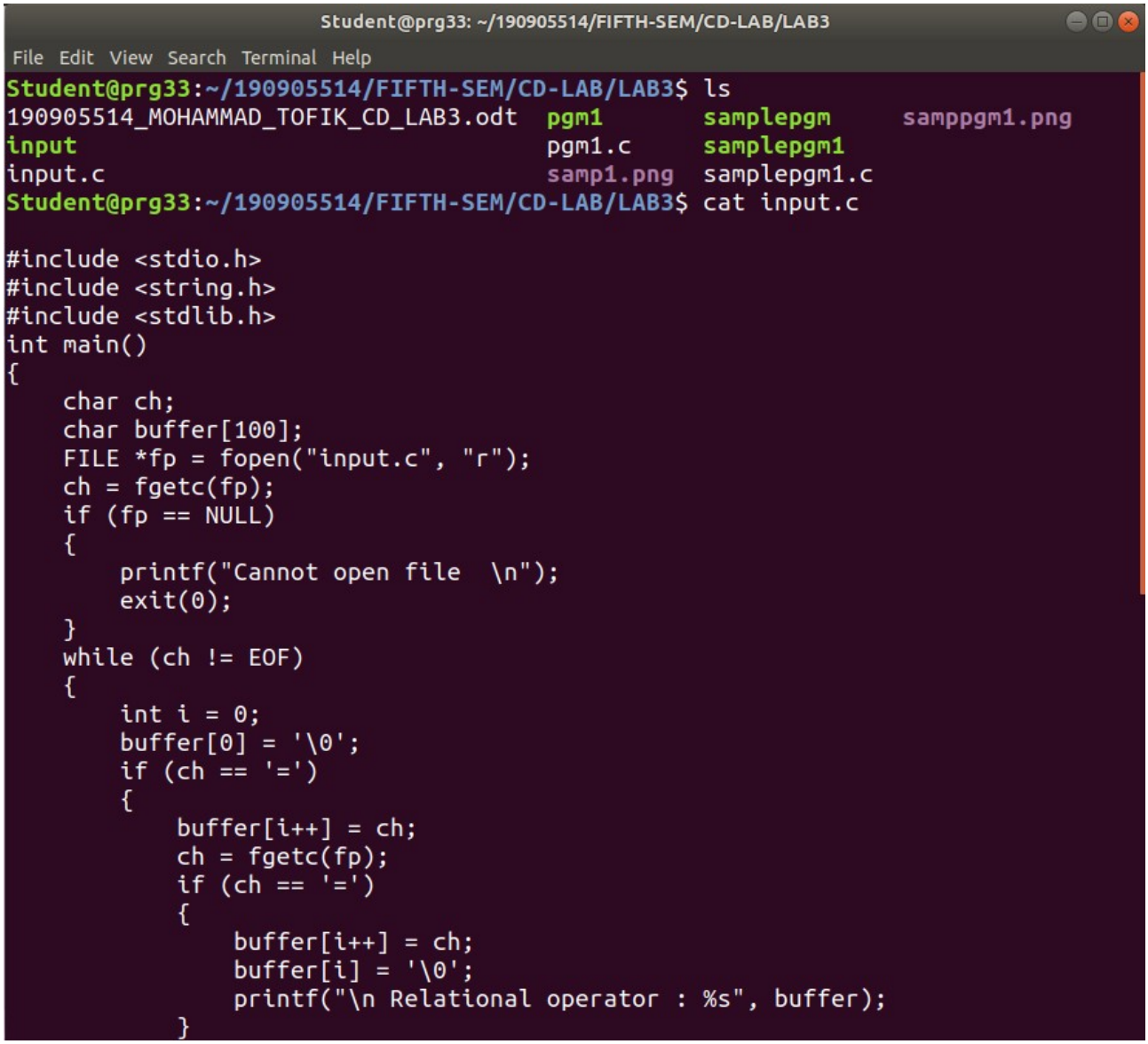
```
strcpy(tkn.type, "IDENTIFIER");
}

gotToken = 1;
fseek(f1, -1, SEEK_CUR);
}

else if (c == '/')
{
int d = fgetc(f1);
++col; //Do we check EOF here?

if (d == '/')
{
while ((c = fgetc(f1)) != EOF && c != '\n')
{
++col;
}

if (c == '\n')
{
newLine();
}
}

else if (d == '*')
{
do
{
if (d == '\n')
{
newLine();
}

while ((c == fgetc(f1)) != EOF && c != '*')
{
++col;

if (c == '\n')
{
```

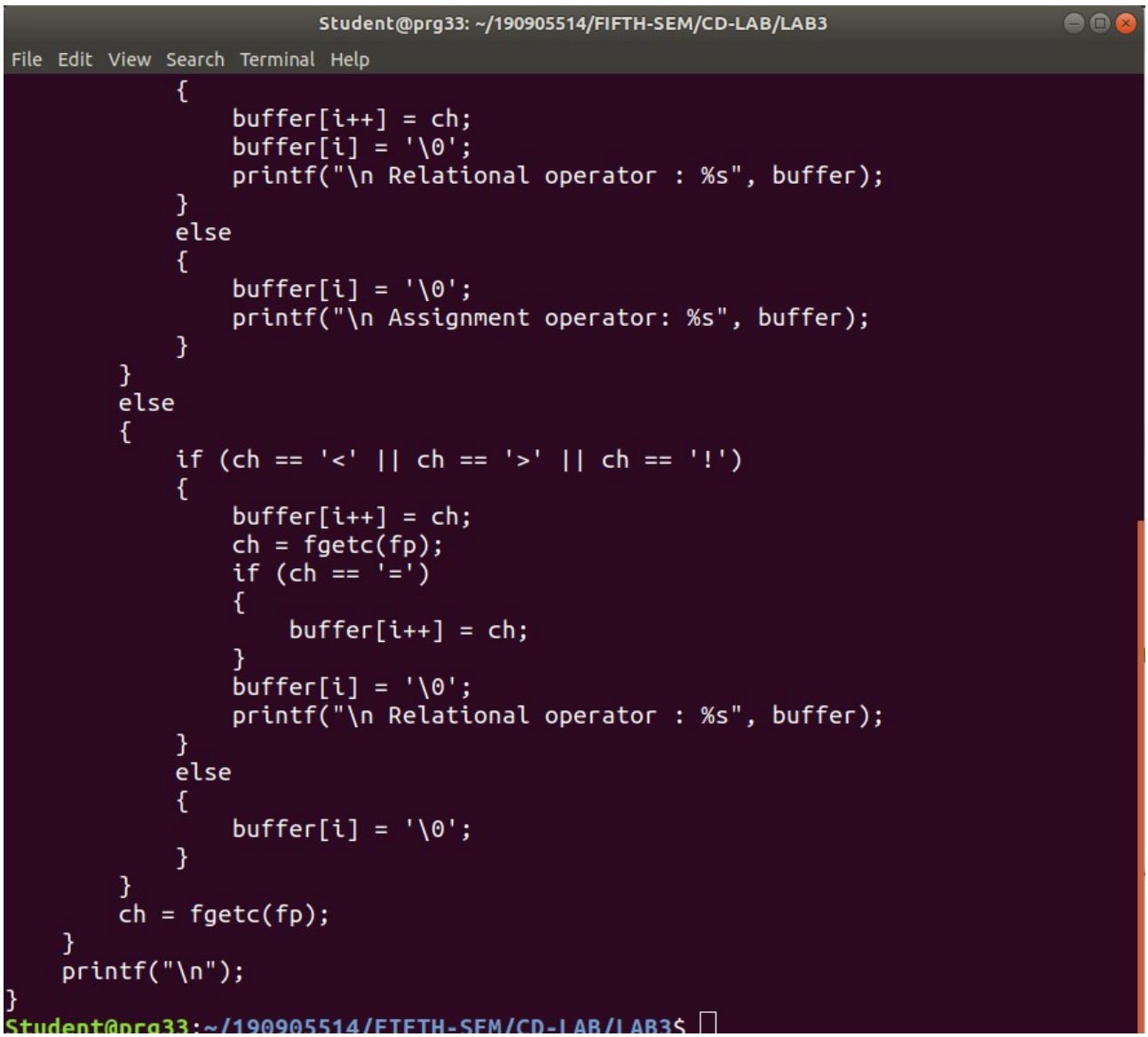
```
newLine();
}
}

++col;
} while ((d == fgetc(f1)) != EOF && d != '/' && (++col));
++col;
}

else
{
fillToken(&tkn, c, row, --col, "ARITHMETIC OPERATOR");
gotToken = 1;
fseek(f1, -1, SEEK_CUR);
}
}

else if (c == '"')
{
tkn.row = row;
tkn.col = col;
strcpy(tkn.type, "STRING LITERAL");
int k = 1;
tkn.lexeme[0] = '"';

while ((c = fgetc(f1)) != EOF && c != '"')
{
tkn.lexeme[k++] = c;
++col;
}

tkn.lexeme[k] = '"';
gotToken = 1;
}

else if (c == '<' || c == '>' || c == '!')
{
fillToken(&tkn, c, row, col, "RELATIONAL OPERATOR");
++col;
int d = fgetc(f1);
```

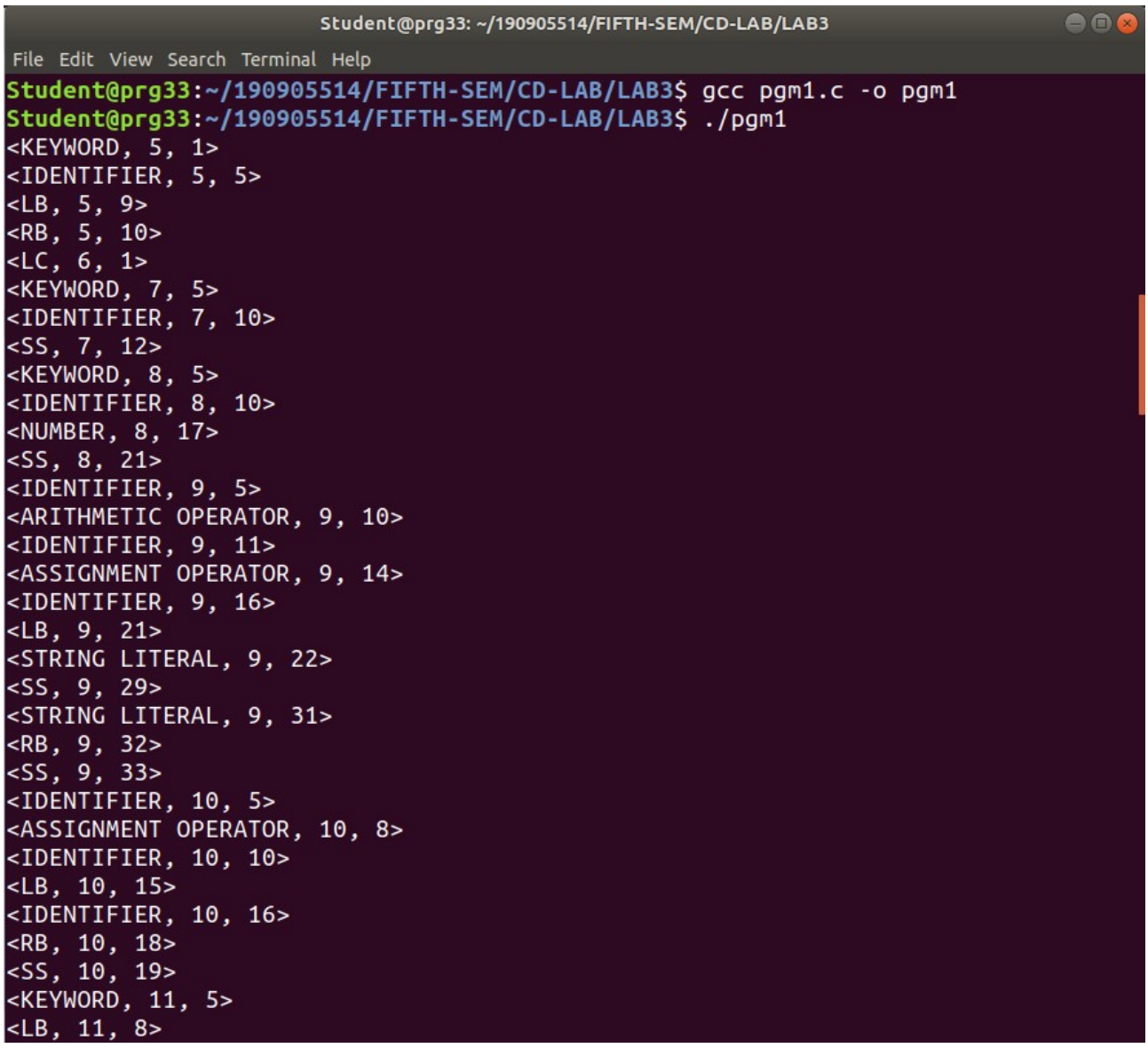
```
if (d == '=')
{
++col;
strcat(tkn.lexeme, "=");
}

else
{
if (c == '!')
{
strcpy(tkn.type, "LOGICAL OPERATOR");
}

fseek(f1, -1, SEEK_CUR);
}

gotToken = 1;
}

else if (c == '&' || c == '|')
{
int d = fgetc(f1);

if (c == d)
{
tkn.lexeme[0] = tkn.lexeme[1] = c;
tkn.lexeme[2] = '\0';
tkn.row = row;
tkn.col = col;
++col;
gotToken = 1;
strcpy(tkn.type, "LOGICAL OPERATOR");
}

else
{
fseek(f1, -1, SEEK_CUR);
}
++col;
}
```

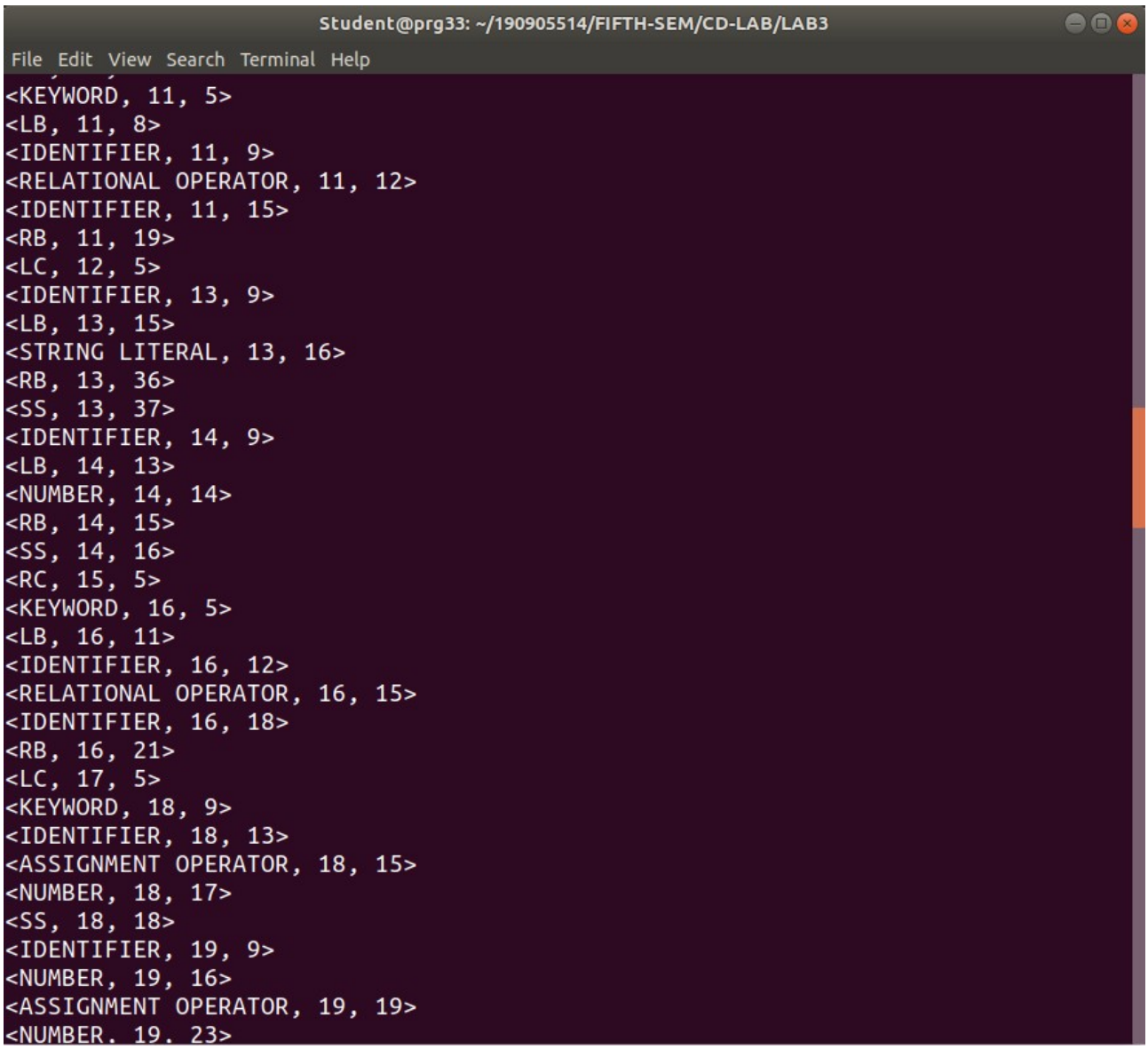
```
else
{
++col;
}
}
return tkn;
}

int main()
{
FILE *f1 = fopen(FILEINPUT, "r");
if (f1 == NULL)
{
printf("Error! File cannot be opened!\n");
return 0;
}

struct token tkn;
while ((tkn = getNextToken(f1)).row != -1)
{
printf("<%s, %d, %d>\n", tkn.type, tkn.row, tkn.col);
}

fclose(f1);
}
```

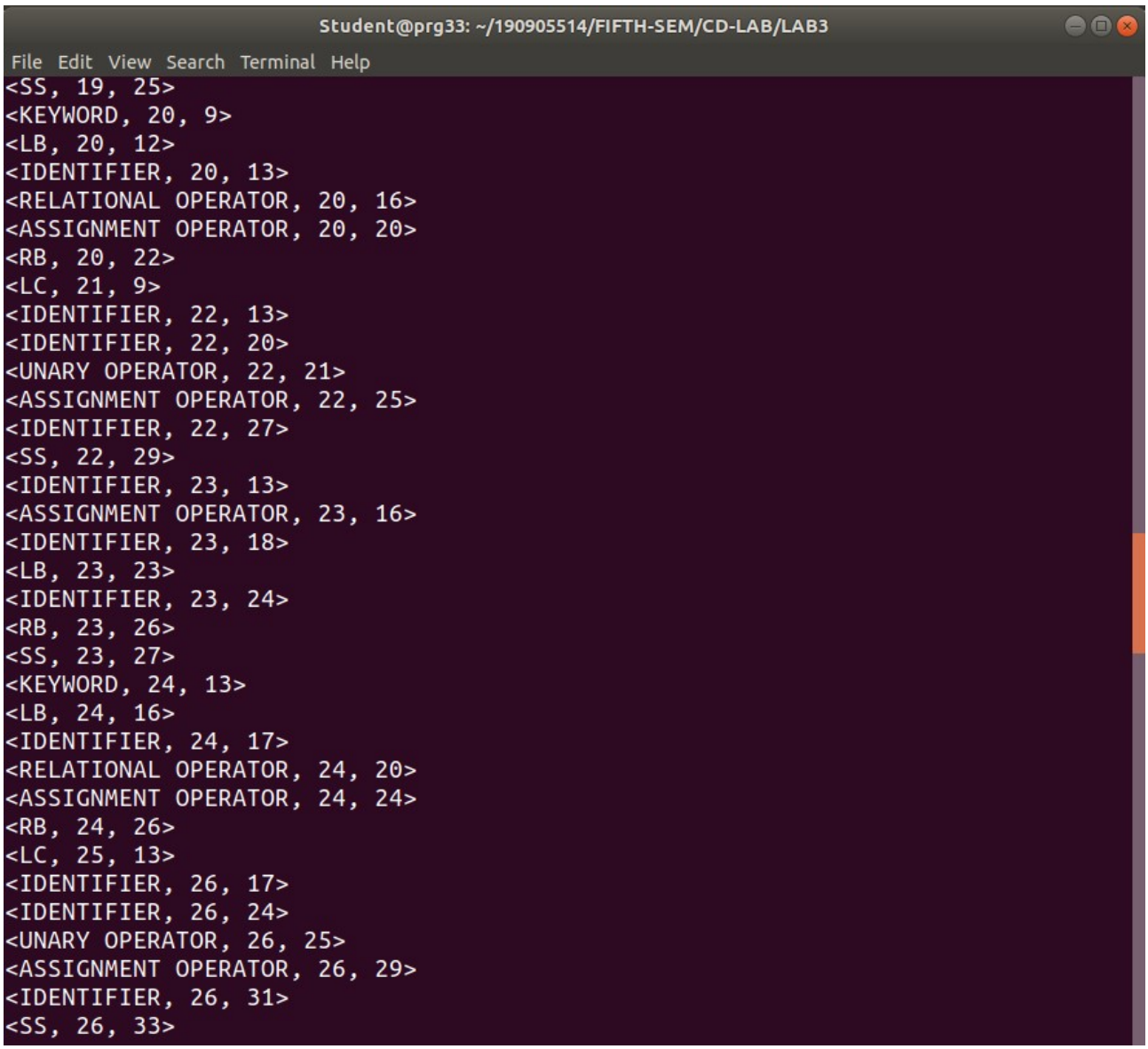
OUTPUT

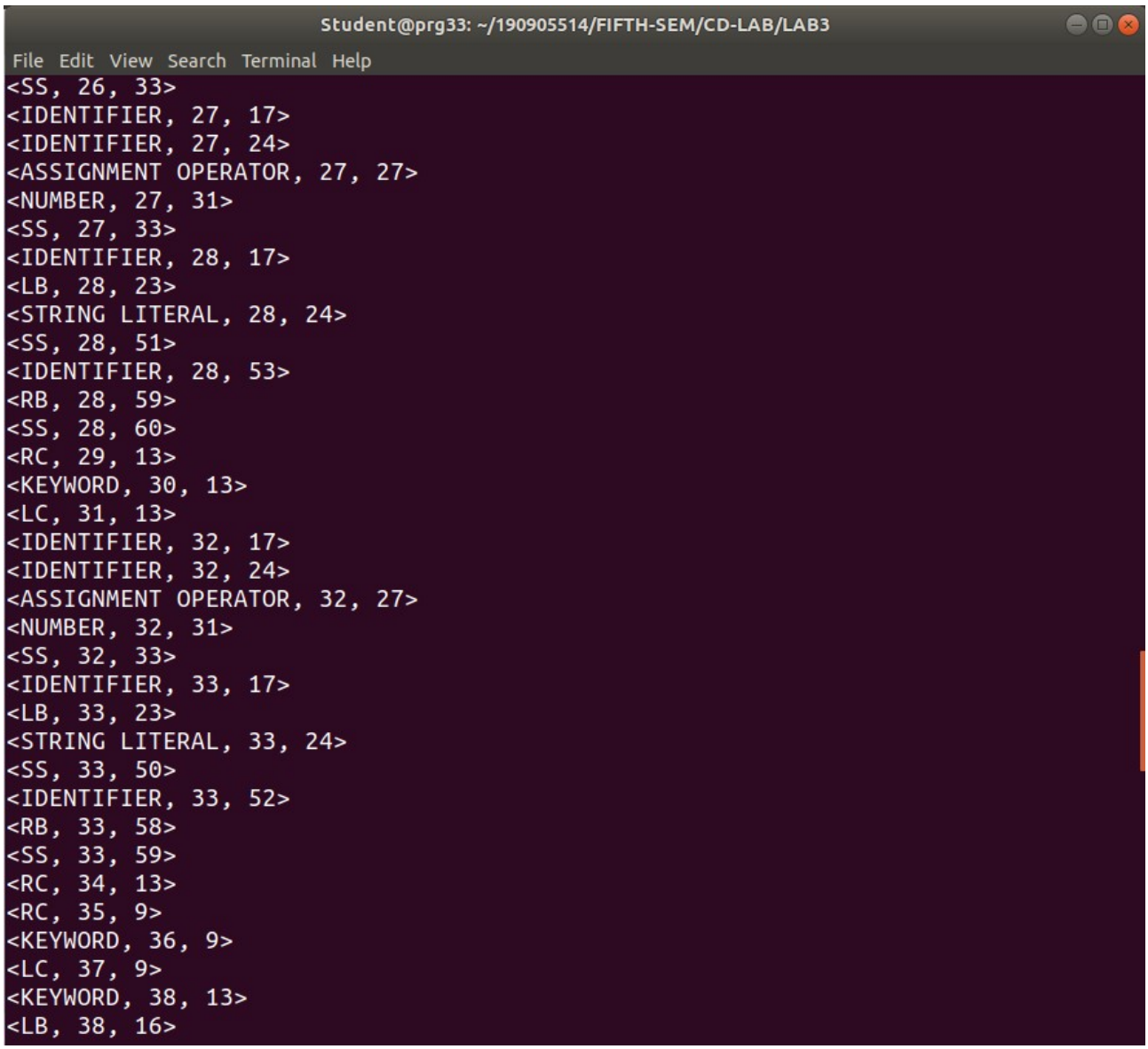

Student@prg33: ~/190905514/FIFTH-SEM/CD-LAB/LAB3
File Edit View Search Terminal Help
Student@prg33:~/190905514/FIFTH-SEM/CD-LAB/LAB3$ ls
190905514_MOHAMMAD_TOFIK_CD_LAB3.odt pgm1 samplepgm samppgm1.png
input pgm1.c samplepgm1
input.c samp1.png samplepgm1.c
Student@prg33:~/190905514/FIFTH-SEM/CD-LAB/LAB3$ cat input.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
int main()
{
char ch;
char buffer[100];
FILE *fp = fopen("input.c", "r");
ch = fgetc(fp);
if (fp == NULL)
{
printf("Cannot open file \n");
exit(0);
}
while (ch != EOF)
{
int i = 0;
buffer[0] = '\0';
if (ch == '=')
{
buffer[i++] = ch;
ch = fgetc(fp);
if (ch == '=')
{
buffer[i++] = ch;
buffer[i] = '\0';
printf("\n Relational operator : %s", buffer);
}

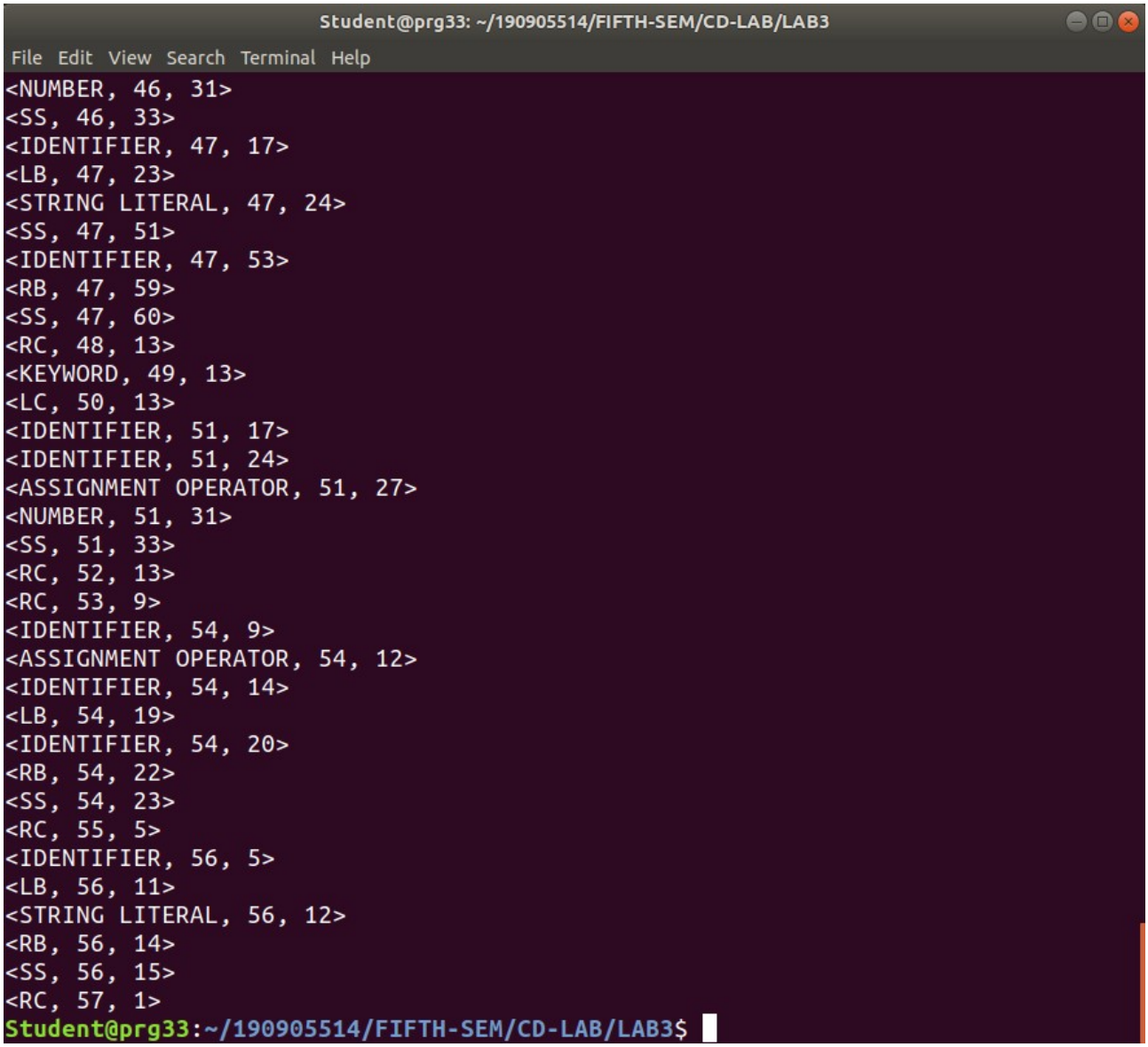

Student@prg33: ~/190905514/FIFTH-SEM/CD-LAB/LAB3
File Edit View Search Terminal Help
{
buffer[i++] = ch;
buffer[i] = '\0';
printf("\n Relational operator : %s", buffer);
}
else
{
buffer[i] = '\0';
printf("\n Assignment operator: %s", buffer);
}
}
else
{
if (ch == '<' || ch == '>' || ch == '!')
{
buffer[i++] = ch;
ch = fgetc(fp);
if (ch == '=')
{
buffer[i++] = ch;
}
buffer[i] = '\0';
printf("\n Relational operator : %s", buffer);
}
else
{
buffer[i] = '\0';
}
}
ch = fgetc(fp);
}
printf("\n");
}
Student@prg33:~/190905514/FIFTH-SEM/CD-LAB/LAB3$


Student@prg33: ~/190905514/FIFTH-SEM/CD-LAB/LAB3
File Edit View Search Terminal Help
Student@prg33:~/190905514/FIFTH-SEM/CD-LAB/LAB3$ gcc pgm1.c -o pgm1
Student@prg33:~/190905514/FIFTH-SEM/CD-LAB/LAB3$ ./pgm1
<KEYWORD, 5, 1>
<IDENTIFIER, 5, 5>
<LB, 5, 9>
<RB, 5, 10>
<LC, 6, 1>
<KEYWORD, 7, 5>
<IDENTIFIER, 7, 10>
<SS, 7, 12>
<KEYWORD, 8, 5>
<IDENTIFIER, 8, 10>
<NUMBER, 8, 17>
<SS, 8, 21>
<IDENTIFIER, 9, 5>
<ARITHMETIC OPERATOR, 9, 10>
<IDENTIFIER, 9, 11>
<ASSIGNMENT OPERATOR, 9, 14>
<IDENTIFIER, 9, 16>
<LB, 9, 21>
<STRING LITERAL, 9, 22>
<SS, 9, 29>
<STRING LITERAL, 9, 31>
<RB, 9, 32>
<SS, 9, 33>
<IDENTIFIER, 10, 5>
<ASSIGNMENT OPERATOR, 10, 8>
<IDENTIFIER, 10, 10>
<LB, 10, 15>
<IDENTIFIER, 10, 16>
<RB, 10, 18>
<SS, 10, 19>
<KEYWORD, 11, 5>
<LB, 11, 8>


Student@prg33: ~/190905514/FIFTH-SEM/CD-LAB/LAB3
File Edit View Search Terminal Help
<KEYWORD, 11, 5>
<LB, 11, 8>
<IDENTIFIER, 11, 9>
<RELATIONAL OPERATOR, 11, 12>
<IDENTIFIER, 11, 15>
<RB, 11, 19>
<LC, 12, 5>
<IDENTIFIER, 13, 9>
<LB, 13, 15>
<STRING LITERAL, 13, 16>
<RB, 13, 36>
<SS, 13, 37>
<IDENTIFIER, 14, 9>
<LB, 14, 13>
<NUMBER, 14, 14>
<RB, 14, 15>
<SS, 14, 16>
<RC, 15, 5>
<KEYWORD, 16, 5>
<LB, 16, 11>
<IDENTIFIER, 16, 12>
<RELATIONAL OPERATOR, 16, 15>
<IDENTIFIER, 16, 18>
<RB, 16, 21>
<LC, 17, 5>
<KEYWORD, 18, 9>
<IDENTIFIER, 18, 13>
<ASSIGNMENT OPERATOR, 18, 15>
<NUMBER, 18, 17>
<SS, 18, 18>
<IDENTIFIER, 19, 9>
<NUMBER, 19, 16>
<ASSIGNMENT OPERATOR, 19, 19>
<NUMBER, 19, 23>


Student@prg33: ~/190905514/FIFTH-SEM/CD-LAB/LAB3
File Edit View Search Terminal Help
<SS, 19, 25>
<KEYWORD, 20, 9>
<LB, 20, 12>
<IDENTIFIER, 20, 13>
<RELATIONAL OPERATOR, 20, 16>
<ASSIGNMENT OPERATOR, 20, 20>
<RB, 20, 22>
<LC, 21, 9>
<IDENTIFIER, 22, 13>
<IDENTIFIER, 22, 20>
<UNARY OPERATOR, 22, 21>
<ASSIGNMENT OPERATOR, 22, 25>
<IDENTIFIER, 22, 27>
<SS, 22, 29>
<IDENTIFIER, 23, 13>
<ASSIGNMENT OPERATOR, 23, 16>
<IDENTIFIER, 23, 18>
<LB, 23, 23>
<IDENTIFIER, 23, 24>
<RB, 23, 26>
<SS, 23, 27>
<KEYWORD, 24, 13>
<LB, 24, 16>
<IDENTIFIER, 24, 17>
<RELATIONAL OPERATOR, 24, 20>
<ASSIGNMENT OPERATOR, 24, 24>
<RB, 24, 26>
<LC, 25, 13>
<IDENTIFIER, 26, 17>
<IDENTIFIER, 26, 24>
<UNARY OPERATOR, 26, 25>
<ASSIGNMENT OPERATOR, 26, 29>
<IDENTIFIER, 26, 31>
<SS, 26, 33>


Student@prg33: ~/190905514/FIFTH-SEM/CD-LAB/LAB3
File Edit View Search Terminal Help
<SS, 26, 33>
<IDENTIFIER, 27, 17>
<IDENTIFIER, 27, 24>
<ASSIGNMENT OPERATOR, 27, 27>
<NUMBER, 27, 31>
<SS, 27, 33>
<IDENTIFIER, 28, 17>
<LB, 28, 23>
<STRING LITERAL, 28, 24>
<SS, 28, 51>
<IDENTIFIER, 28, 53>
<RB, 28, 59>
<SS, 28, 60>
<RC, 29, 13>
<KEYWORD, 30, 13>
<LC, 31, 13>
<IDENTIFIER, 32, 17>
<IDENTIFIER, 32, 24>
<ASSIGNMENT OPERATOR, 32, 27>
<NUMBER, 32, 31>
<SS, 32, 33>
<IDENTIFIER, 33, 17>
<LB, 33, 23>
<STRING LITERAL, 33, 24>
<SS, 33, 50>
<IDENTIFIER, 33, 52>
<RB, 33, 58>
<SS, 33, 59>
<RC, 34, 13>
<RC, 35, 9>
<KEYWORD, 36, 9>
<LC, 37, 9>
<KEYWORD, 38, 13>
<LB, 38, 16>


Student@prg33: ~/190905514/FIFTH-SEM/CD-LAB/LAB3
File Edit View Search Terminal Help
<NUMBER, 46, 31>
<SS, 46, 33>
<IDENTIFIER, 47, 17>
<LB, 47, 23>
<STRING LITERAL, 47, 24>
<SS, 47, 51>
<IDENTIFIER, 47, 53>
<RB, 47, 59>
<SS, 47, 60>
<RC, 48, 13>
<KEYWORD, 49, 13>
<LC, 50, 13>
<IDENTIFIER, 51, 17>
<IDENTIFIER, 51, 24>
<ASSIGNMENT OPERATOR, 51, 27>
<NUMBER, 51, 31>
<SS, 51, 33>
<RC, 52, 13>
<RC, 53, 9>
<IDENTIFIER, 54, 9>
<ASSIGNMENT OPERATOR, 54, 12>
<IDENTIFIER, 54, 14>
<LB, 54, 19>
<IDENTIFIER, 54, 20>
<RB, 54, 22>
<SS, 54, 23>
<RC, 55, 5>
<IDENTIFIER, 56, 5>
<LB, 56, 11>
<STRING LITERAL, 56, 12>
<RB, 56, 14>
<SS, 56, 15>
<RC, 57, 1>
Student@prg33:~/190905514/FIFTH-SEM/CD-LAB/LAB3$