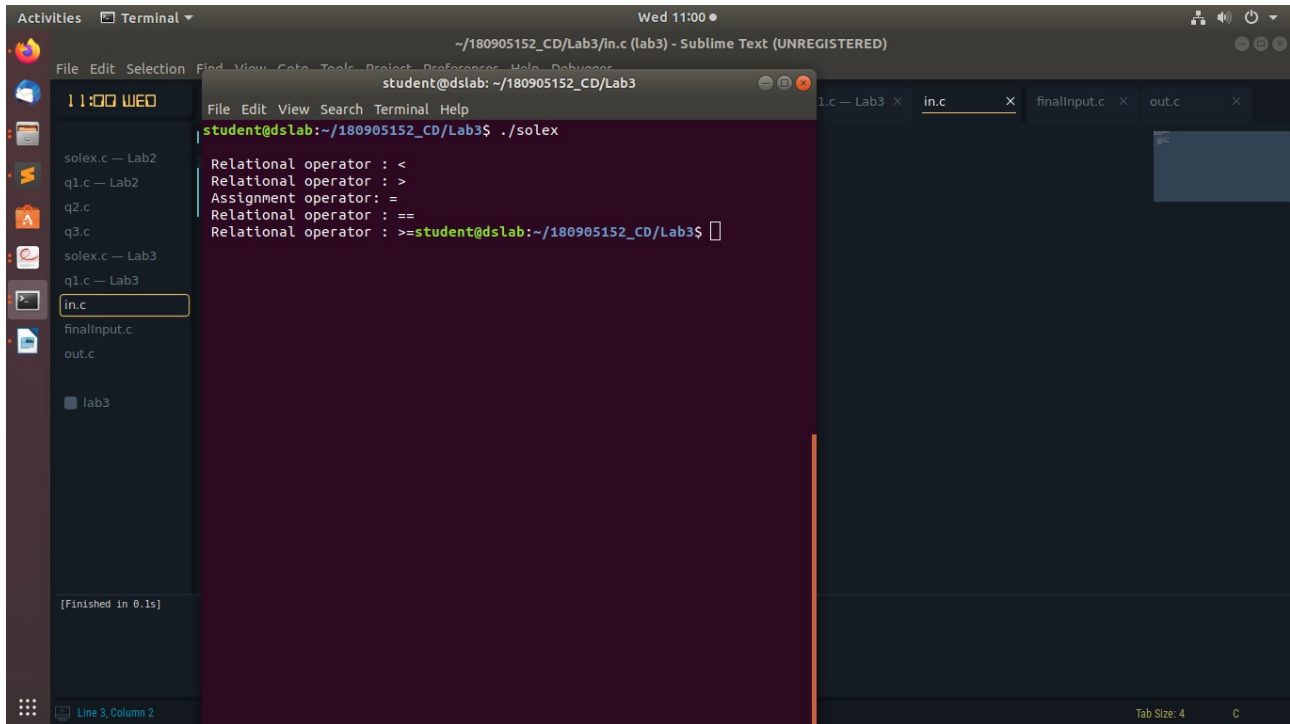


Lab 3

Solved Exercise



Code

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int main()
{
    char c,buf[10];
    FILE *fp=fopen("in.c","r");
    c = fgetc(fp);
    if (fp == NULL)
    {
        printf("Cannot open file \n");
        exit(0);
    }
    while(c!=EOF)
    {
        int i=0;
        buf[0]='\0';
        if(c=='=')
        {
            buf[i++]=c;
            c = fgetc(fp);
            if(c=='=')
            {
                buf[i++]=c;
                buf[i]='\0';
            }
        }
    }
}
```

```

        printf("\n Relational operator : %s",buf);
    }
    else
    {
        buf[i]='\0';
        printf("\n Assignment operator: %s",buf);
    }
}
else
{
    if(c=='<'||c=='>'||c=='!')
    {
        buf[i++]=c;
        c = fgetc(fp);
        if(c=='=')
        {
            buf[i++]=c;
        }
        buf[i]='\0';
        printf("\n Relational operator : %s",buf);
    }
    else
    {
        buf[i]='\0';
    }
}
c = fgetc(fp);
}
}

```

Q1

The screenshot shows a terminal window with the following content:

```

~/180905152_CD/Lab3/in.c (lab3)
student@dslab: ~/180905152_CD/Lab3

File Edit Selection Find View Goto Tools Project Preferences Help Debugger
104.0 WEO
solex.c — Lab2 x q1.c — Lab2 x q2.c x q3.c
solex.c — Lab2
q1.c — Lab2
q2.c
q3.c
solex.c — Lab3
q1.c — Lab3
in.c
finalInput.c
out.c
lab3

1 #include <stdio.h>
2 int main()
3 {
4     //some comment
5     sum = 547;
6     sum = a && b;
7     p=q;
8 }

[Finished in 0.1s]

<7 , 5 , 7>
<; , 5 , 8>
<sum , 6 , 2>
<== , 6 , 3>
<a , 6 , 6>
<&& , 6 , 7>
<b , 6 , 10>
<; , 6 , 10>
<p , 7 , 2>
<>= , 7 , 2>
<q , 7 , 4>
<; , 7 , 4>
<} , 8 , 1>
student@dslab:~/180905152_CD/Lab3$ ./q1
<int , 2 , 1>
<main , 2 , 5>
<( , 2 , 9>
<) , 2 , 10>
<[ , 3 , 1>
<sum , 5 , 2>
<= , 5 , 6>
<5 , 5 , 8>
<+ , 5 , 9>
<7 , 5 , 10>
<; , 5 , 11>
<sum , 6 , 2>
<== , 6 , 6>
<a , 6 , 9>
<&& , 6 , 11>
<b , 6 , 14>
<; , 6 , 15>
<p , 7 , 2>
<>= , 7 , 3>
<q , 7 , 5>
<; , 7 , 6>
<} , 8 , 1>
student@dslab:~/180905152_CD/Lab3$

```

Code

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

struct token
{
    char name[20];
    int row,col;
};

char buffer[20];
char keywords[][20] =
{"int","char","float","string","double","if","else","break","continue","for","while","do","main"};
char specialchars[] = {',',';','?',':',';','(',')','{','}'};
char arithmeticsymbols[] = {'+','-','*','/'};
int row = 1,col=0,ca,cb;
FILE *fa,*fb;

struct token getNextToken()
{
    while(ca!=EOF)
    {
        //PreProcessors
        if(ca=='#'){
            while(ca!='\n'){
                col++;
                ca=getc(fa);
            }
        }
        //Comments
        if(ca == '/')
        {
            col++;
            cb = getc(fa);
            if(cb == '/')
            {
                while(ca!='\n')
                {
                    col++;
                    ca = getc(fa);
                }
            }
            else if(cb == '*')
            {
                do{
                    while(ca!='*')
                    {
                        col++;
                        if(ca=='\n'){
                            col=1;
                        }
                    }
                } while(1);
            }
        }
    }
}
```

```

                                row++;
                                }
                                ca = getc(fa);
                                }
                                ca = getc(fa);
                                }while(ca != '/');
                                }
                                }
//Literal
if(ca=="")
{
    int i=0;
    ca = getc(fa);
    col++;
    while(ca!="")
    {
        buffer[i++] = ca;
        ca = getc(fa);
    }
    buffer[i] = '\0';
    struct token t;
    int j=0;
    while(buffer[j]!='\0')
    {
        t.name[j] = buffer[j];
        j++;
    }
    t.name[j] = '\0';
    t.row = row;
    t.col = col;
    col = col + strlen(buffer);
    ca = getc(fa);
    return t;
}
//blank space
if(ca == ' ')
{
    col++;
    ca = getc(fa);
}
//new line
if(ca == '\n')
{
    row++;
    col=1;
    ca = getc(fa);
}
//Special Character
for(int i=0;i<9;i++)
{
    if(ca==specialchars[i])
    {

```

```

        struct token t;
        t.name[0] = ca;
        t.name[1] = '\0';
        t.row = row;
        t.col = col;
        ca = getc(fa);
        col++;
        return t;
    }
}
//Arithmetic Operators
for(int i=0;i<4;i++)
{
    if(ca == arithmeticsymbols[i])
    {
        struct token t;
        t.name[0] = ca;
        t.name[1] = '\0';
        t.row = row;
        t.col = col;
        ca = getc(fa);
        col++;
        return t;
    }
}
//Assignment or equals op
if(ca == '=')
{
    ca = getc(fa);

    if(ca == '=')
    {
        struct token t;
        t.name[0] = '=';
        t.name[1] = '=';
        t.name[2] = '\0';
        t.row = row;
        t.col = col;
        ca = getc(fa);
        col+=2;
        return t;
    }
    else{
        struct token t;
        t.name[0] = '=';
        t.name[1] = '\0';
        t.row = row;
        t.col = col;
        col++;
        return t;
    }
}
}

```

```

//RelOps
if(ca == '<')
{
    ca = getc(fa);

    if(ca == '=')
    {
        struct token t;
        t.name[0] = '<';
        t.name[1] = '=';
        t.name[2] = '\0';
        t.row = row;
        t.col = col;
        ca = getc(fa);
        col+=2;
        return t;
    }
    else{
        struct token t;
        t.name[0] = '<';

        t.name[1] = '\0';
        t.row = row;
        t.col = col;
        col++;
        return t;
    }
}
else if(ca == '>')
{
    ca = getc(fa);

    if(ca == '=')
    {
        struct token t;
        t.name[0] = '>';
        t.name[1] = '=';
        t.name[2] = '\0';
        t.row = row;
        t.col = col;
        ca = getc(fa);
        col+=2;
        return t;
    }
    else{
        struct token t;
        t.name[0] = '>';

        t.name[1] = '\0';
        t.row = row;
        t.col = col;
        col++;
    }
}

```

```

        return t;
    }
}
//logical ops
if(ca == '&')
{
    ca = getc(fa);

    if(ca == '&')
    {
        struct token t;
        t.name[0] = '&';
        t.name[1] = '&';
        t.name[2] = '\0';
        t.row = row;
        t.col = col;
        ca = getc(fa);
        col+=2;
        return t;
    }
    else{
        struct token t;
        t.name[0] = '&';

        t.name[1] = '\0';
        t.row = row;
        t.col = col;
        col++;
        return t;
    }
}
if(ca == '|')
{
    ca = getc(fa);

    if(ca == '|')
    {
        struct token t;
        t.name[0] = '|';
        t.name[1] = '|';
        t.name[2] = '\0';
        t.row = row;
        t.col = col;
        ca = getc(fa);
        col+=2;
        return t;
    }
    else{
        struct token t;
        t.name[0] = '|';
        t.name[1] = '\0';
        t.row = row;

```

```

        t.col = col;
        col++;
        return t;
    }
}
if(ca == '^')
{
    struct token t;
    t.name[0] = '^';
    t.name[1] = '\0';
    t.row = row;
    t.col = col;
    col++;
    return t;
}
//Numeric
int i=0;
if(isdigit(ca)){
    while(isdigit(ca)){
        buffer[i++] = ca;
        ca = getc(fa);
    }
    buffer[i] = '\0';
    struct token t;
    strcpy(t.name,buffer);
    t.row = row;
    t.col = col;
    col+=strlen(buffer);
    return t;
}
//Keywords
i=0;
while(isalpha(ca)){
    buffer[i++] = ca;
    ca = getc(fa);
}
buffer[i] = '\0';
for(int j=0;j<13;j++)
{
    if(strcmp(buffer,keywords[j])==0)
    {
        struct token t;
        strcpy(t.name,buffer);
        t.row = row;
        t.col = col;
        col = col + strlen(buffer);
        return t;
    }
}
if(buffer[0]!='\0')
{

```



```

        struct token t;
        strcpy(t.name,buffer);
        t.row = row;
        t.col = col;
        col+=strlen(buffer);
        return t;
    }
    ca = getc(fa);
    col++;
}
struct token t;
t.row = -1;
}

int main()
{
    fa = fopen("in.c","r");
    if(fa == NULL)
    {
        printf("Cannot open file \n");
        return 0;
    }
    ca = getc(fa);
    col = 1;
    while(ca!=EOF)
    {
        struct token t = getNextToken();
        if(t.row == -1)
            break;
        printf("<%s , %d , %d>\n",t.name,t.row,t.col );
    }
}

```

Name: Shruti Verma
RegNo: 180905152
CSE B -25