VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

Lab REPORT
on

Compiler Design

Submitted by

Harsh Ghiya (1BM21CS073)

Under the Guidance of
Prof. Prameetha Pai
Assistant Professor, BMSCE

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019

Nov-2023 to Feb-2024

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the lab work entitled "**Compiler Design**" carried out by **Harsh Ghiya (1BM21CS073)** who are bona fide students of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visveswaraiah Technological University, Belgaum during the year 2023-2024. The lab report has been approved as it satisfies the academic requirements in respect of compiler design lab **(22CS5PCCPD)** work prescribed for the said degree.

Prameetha Pai
Assistant Professor
Dept. of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Prof.& Head
Dept. of CSE
BMSCE, Bengaluru

Compiler Design Lab Record.

INDEX

NAME: Harish Ghiya STD.: 5th Sem SEC.: B ROLL NO.: 1BM21CS073

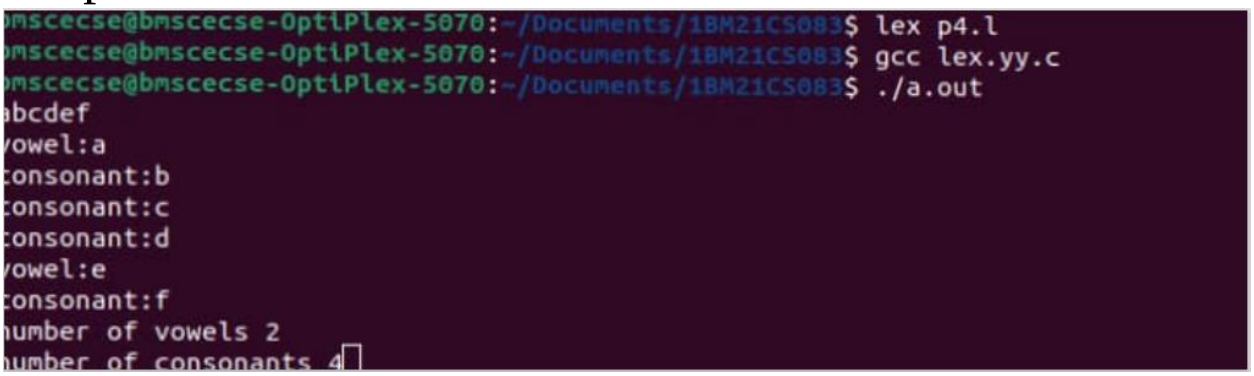
S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
		Lex	10/10	
1.	6/11/23	program to design lexical Analyzer to identify seperate keywords, identifier ~~in a stmt to check operation~~		
2.	20/11/23	count the number of vowels and consonants.		
3.	27/11/23	Floating point numbers		
4.	4/12/23	Replacing sequence of non-empty spaces with single space		
5.	11/12/2023	Recognize tokens over alphabets d 0, -- . 8, 9 3		
6.	18/12/23	Program to ~~analyze~~ design lexical analyzer		
7.	8/1/2024	Recursive descent program		
8.	8/1/2024	Design parsing using YACC		
9.	29/1/24	Yacc prog. to gen. syntax tree for a given arithmetic expression		
10.	29/1/24	Infix to postfix using YACC		
11.	29/1/24	YACC generate 3-add. code.		
12.	29/1/24	YACC program string grammer $a^n b^n$		

Code:

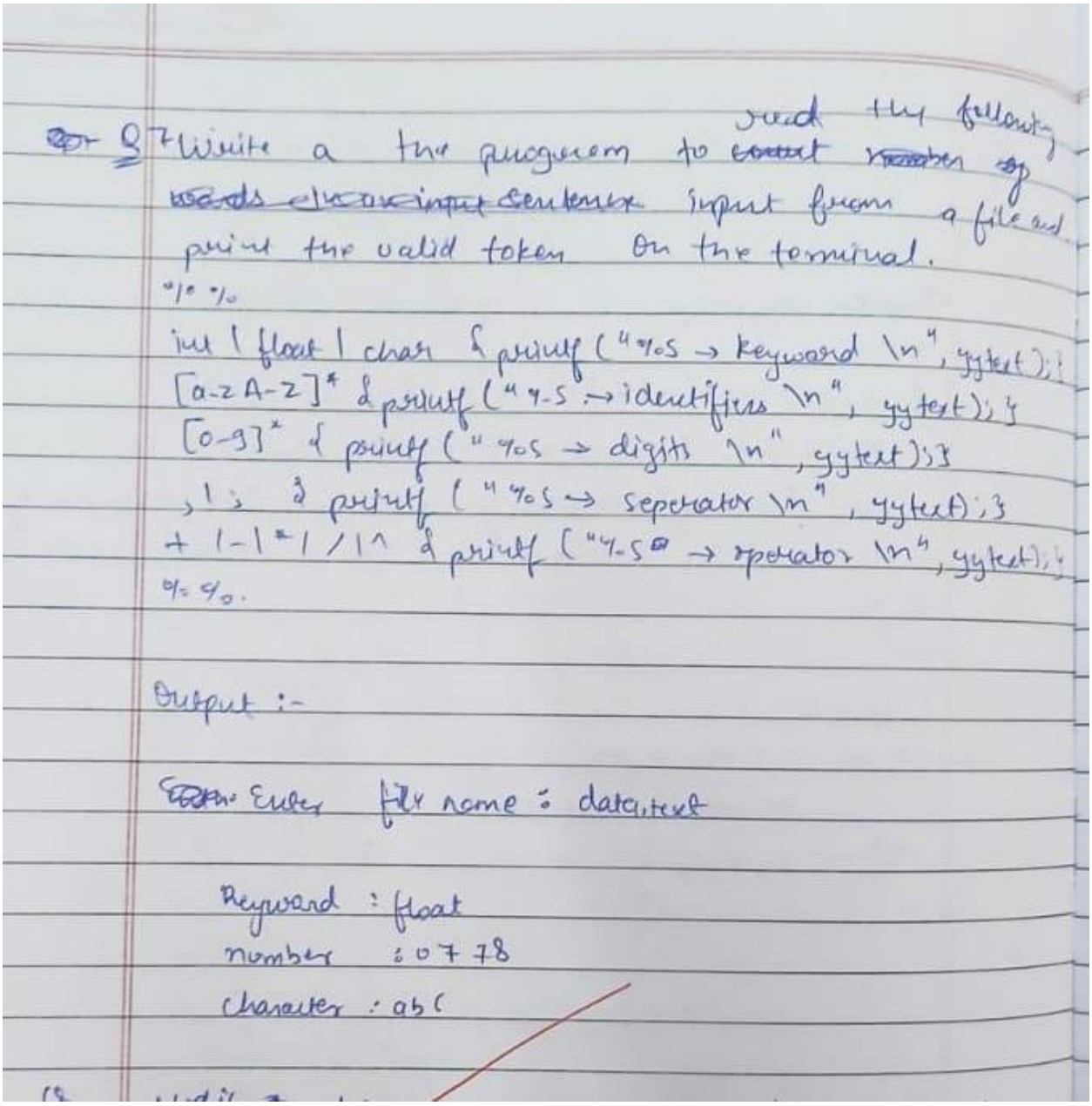
→ Write a LEX program to identify each character as a consonant or vowel in a given sentence.

```
=> %option noyywrap

%{
#include<stdio.h>
%}

%%
a|e|i|o|u|A|E|I|O|U { printf("vowel\n");}
[A-Za-z] { printf("Consonant");}
. { printf("Invalid\n");}
%%

int main()
{ printf("Enter");
  yylex();
  return 0;
}
```

Output:


bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex p4.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ gcc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
abcdef
vowel:a
consonant:b
consonant:c
consonant:d
vowel:e
consonant:f
number of vowels 2
number of consonants 4

Code:



Q7 Write a lex program to read the following input from a file and print the valid token on the terminal.

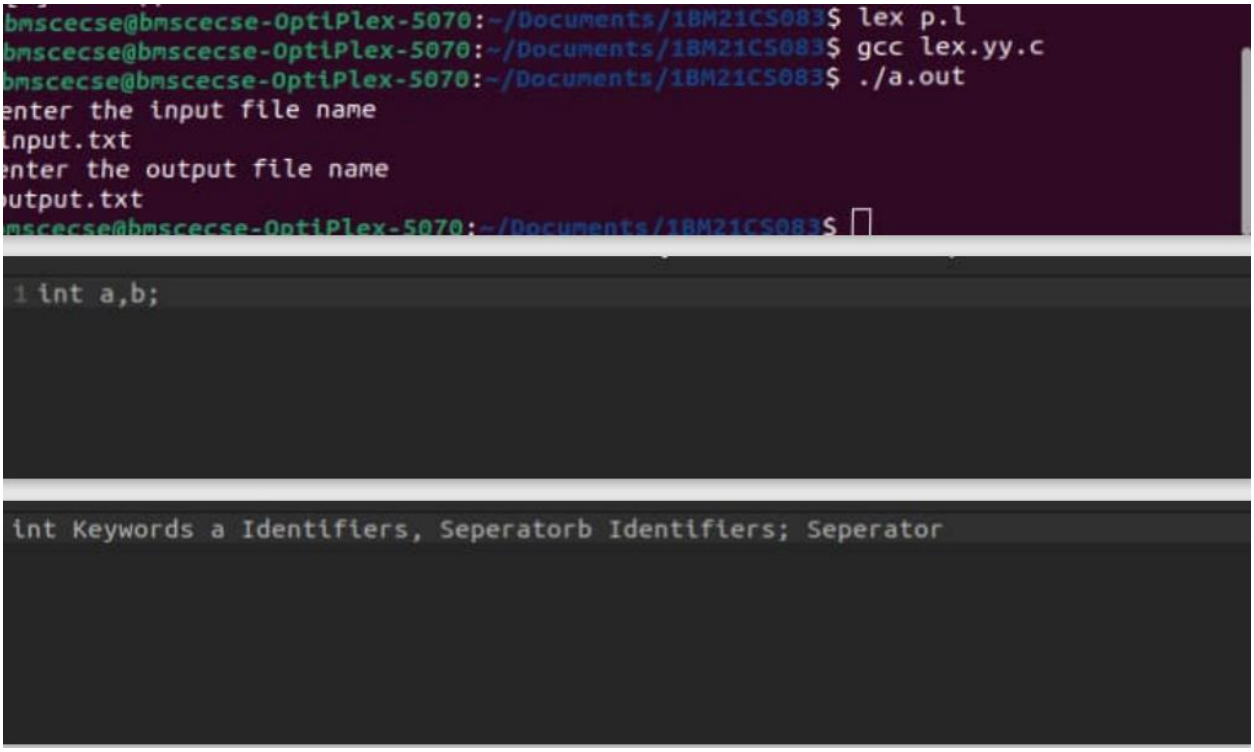
```
%{ %}
int | float | char { printf("%s → keyword \n", yytext); }
[a-zA-Z]* { printf("%s → identifiers \n", yytext); }
[0-9]* { printf("%s → digits \n", yytext); }
, | ; { printf("%s → seperator \n", yytext); }
+ | - | * | / | ^ { printf("%s → operator \n", yytext); }
%% %%.
```

Output :-

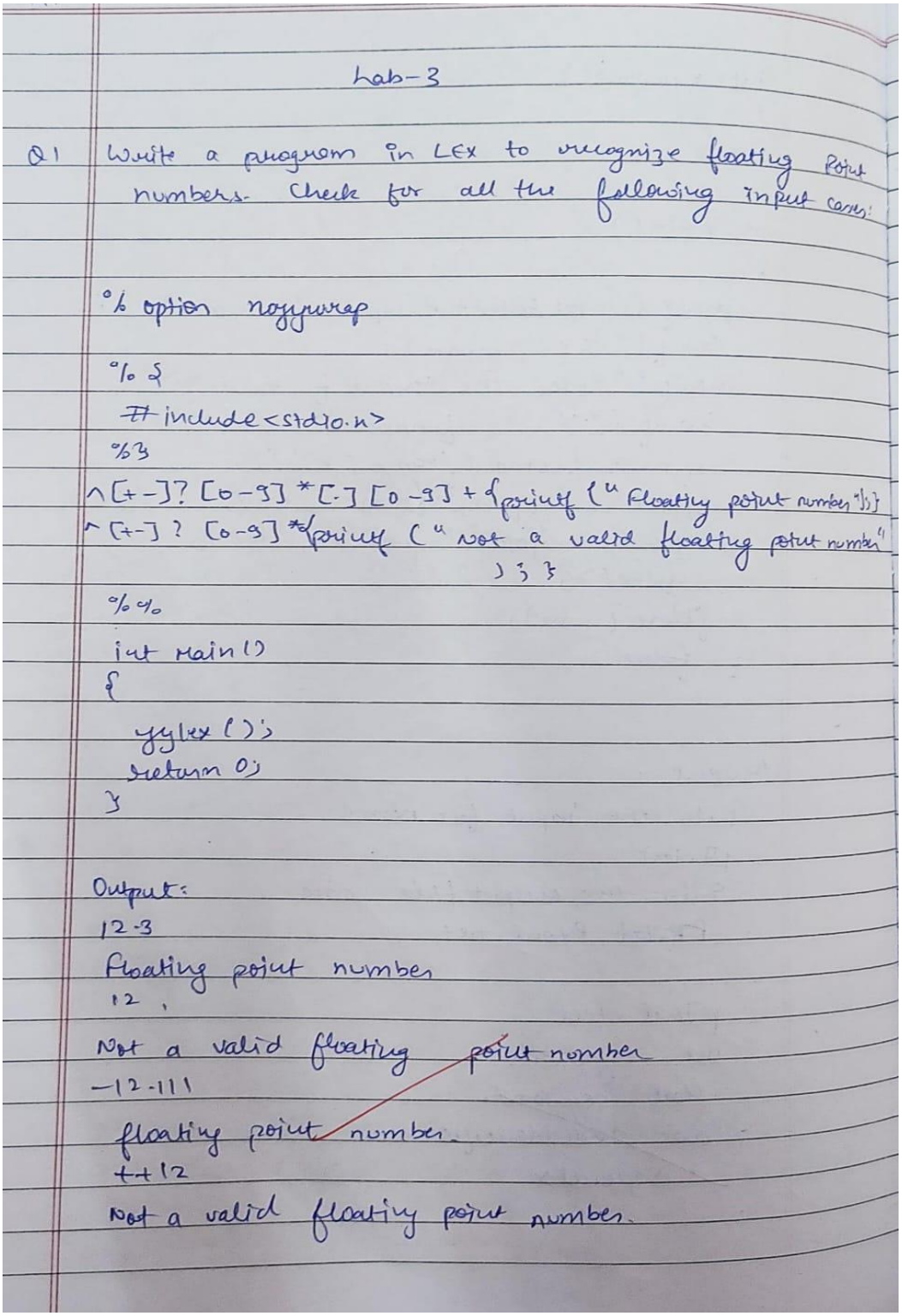
Enter file name : data.text

Keyword : float
number : 0 7 78
character : a b c

Output


bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex p.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ gcc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
enter the input file name
input.txt
enter the output file name
output.txt
1 int a,b;
int Keywords a Identifiers, Seperatorb Identifiers; Seperator

Code



Lab-3

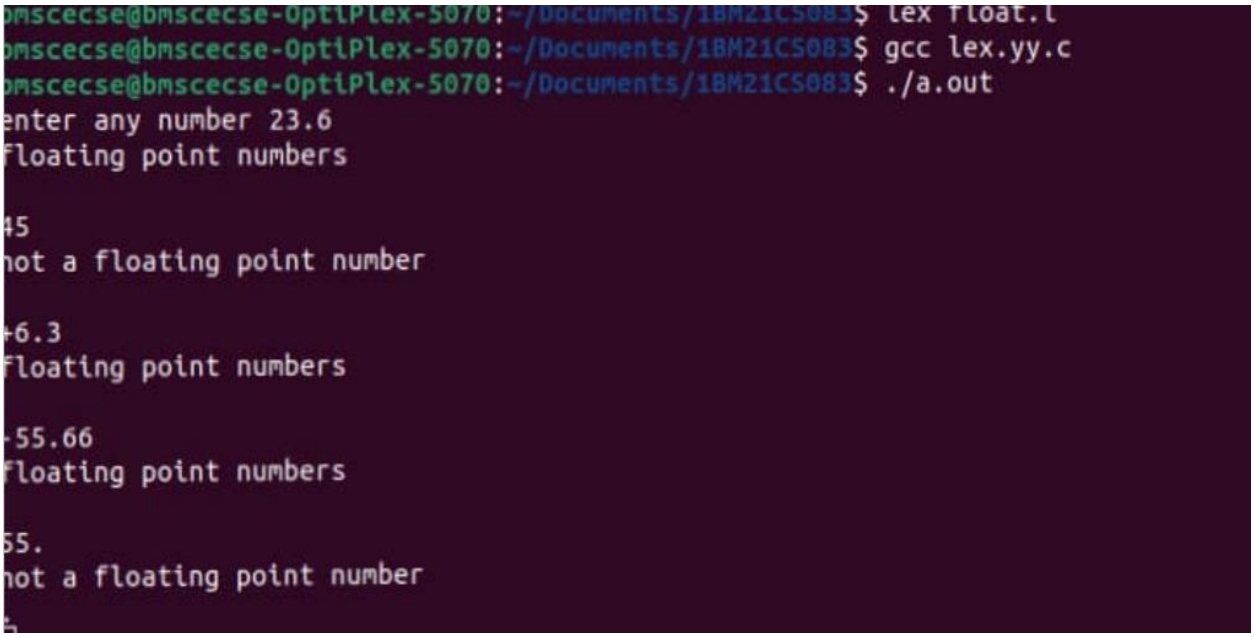
Q1 Write a program in LEx to recognize floating point numbers. Check for all the following input cases:

```
%{
#include<stdio.h>
%}
^[+-]?[0-9]*[.][0-9]+ {printf("Floating point number");}
^[+-]?[0-9]* {printf("Not a valid floating point number");}
%%
int main()
{
  yylex();
  return 0;
}
```

Output:

12-3
Floating point number
12.
Not a valid floating point number
-12.111
floating point number.
++12
Not a valid floating point number.

Output


bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex float.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ gcc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
enter any number 23.6
floating point numbers
45
not a floating point number
+6.3
floating point numbers
-55.66
floating point numbers
55.
not a floating point number

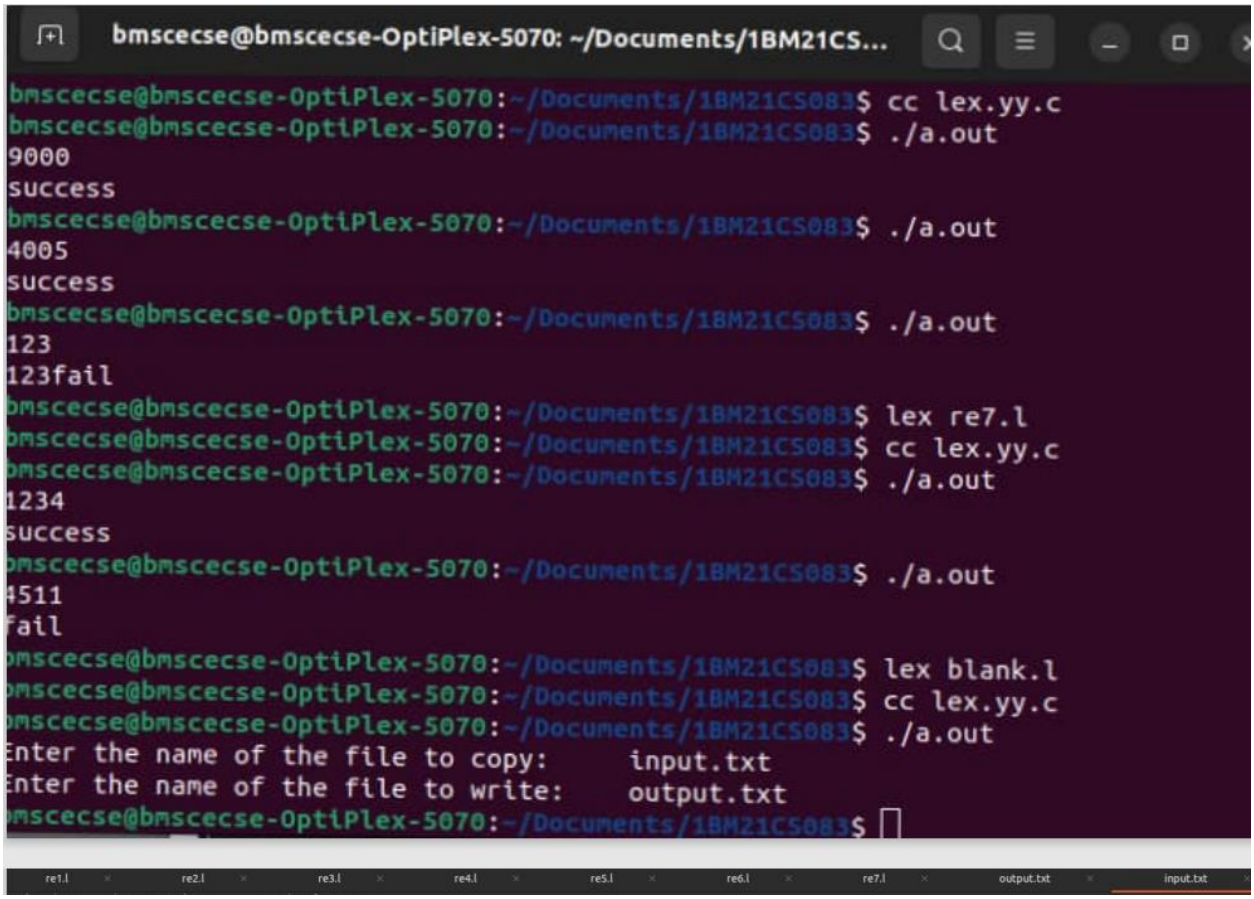
Lab-4

Write a LEX program that copies a file, replacing each non-empty sequence of white space by a single blank.

```
%{
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

  char str1[200];
%}

%%
[\n] {sprintf (yyout, "%s\n", str1); str[0]='\0';}
[ ]* {sprintf (yyout "%s" str1); str[0] = '\0';
      fprintf ('yyout "%s", " ");
      Strcat (str1, yytext);}

<<EOF>> {sprintf (yyout, "%s", str1); return 0;}

%%

int main()
{
    extern file *yyin, *yyout;

    char filename[100];
    printf ("Enter the name of the file to copy: \t
    scanf ("%s", filename);
```

```
yyin = fopen(filename "r");
if (yyin == NULL)
{
    exit(0);
}

printf("Enter the name of the file to write", \t");
    scanf("%s", filename);
    yyout = fopen(filename, "w");

    if (yyout == NULL)
    {
        exit(1);
    }
    yylex();
}

int yywrap(void)
{
}
```

11/11/23


bmscecse@bmscecse-OptiPlex-5070: ~/Documents/1BM21CS...
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
9000
success
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
4005
success
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
123
123fail
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re7.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
1234
success
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
4511
fail
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex blank.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
Enter the name of the file to copy: input.txt
Enter the name of the file to write: output.txt
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$
re1.l
re2.l
re3.l
re4.l
re5.l
re6.l
re7.l
output.txt
input.txt

Bafna Gold
Date: Page:

→ The set of all strings with 3 consecutive 2's

(b)
```
%%
[0-9]*[222]*[0-9]*{printf("string accepted");}
[0-9]*{printf("string rejected");}
%%

int main()
{
  yylex()
return 0;
}
int yywrap()
{
}
```

Output :-

12
String rejected.
222
String accepted.

→ The set of all string ending in 00.

(a)
```
%%
[0-9]*[00]{printf("string accepted");}
[0-9]*{printf("string rejected");}
%%
```

Output-

12300
String accepted
12
String rejected.

(g)

Code :-

```
%%
[0-9] {if (atoi(yytext) > prev) {prev = atoi(yytext); count++;}
      Else {flag=0; break;}}
\n    {if (flag==1 && count==4) {printf("yes\n");
      count=0 ;}}
.  {}
%%.
```

Output
1234
Success

(e) λ [0-9]

Code

```
%%
~~[0-9]* | [0-9]~~
%%
  (d d3) * | {d}{9} {printf("String accepted");}
%%
```

Output:
0 1 2 3 4 5 6 7 8 9 1

10th symbol from left is 9

14/12/2023

(8) The set of all 4 digits numbers whose sum is 9

code :-

```
%{
#include <stdio.h>
#include <stdlib.h>
int sum = 0;
int count = 0;
%}

%%
[0-9] { sum = sum + atoi(yytext); count = count+1; }
\n { if (sum%9==0 && count == 4)
    { printf("yes\n"); sum=0; count=0; }
  }
%%

int yywrap()
{
  }

int main()
{
  yylex();
return 0;
}
```

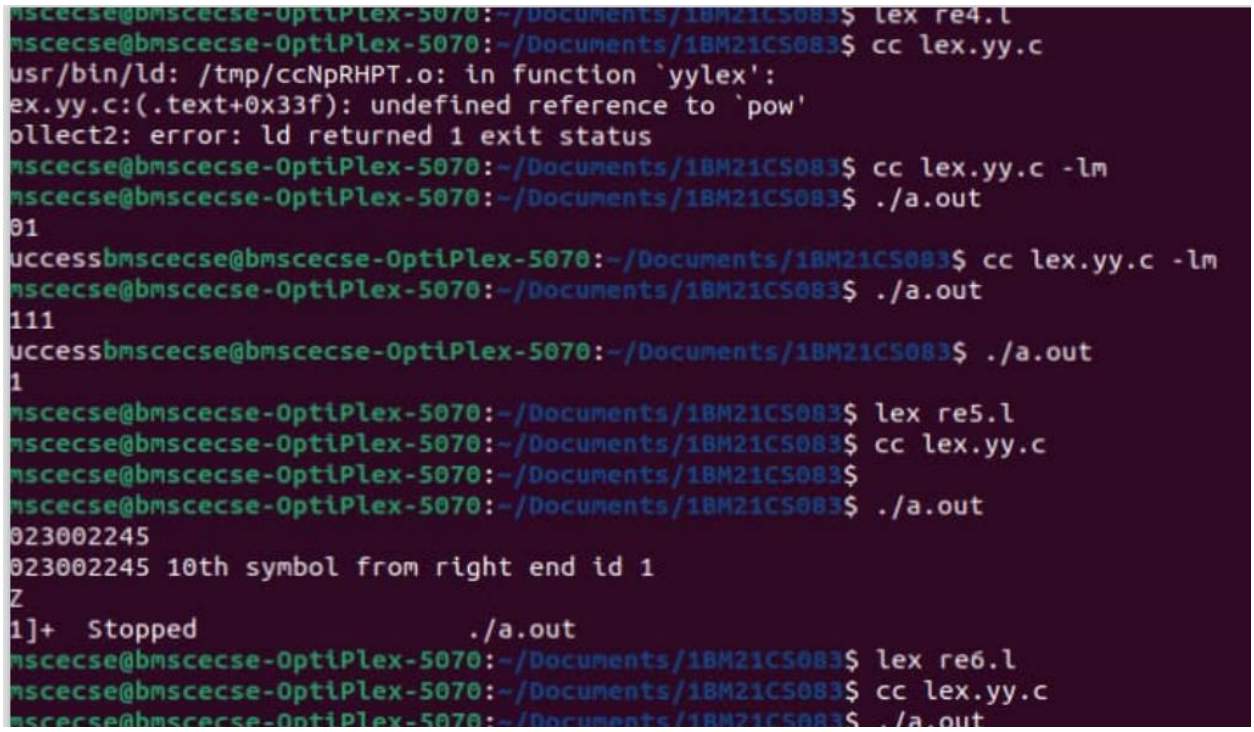
Output :

1 2 2 3

Success

```
11
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re5.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
1023002245
1023002245 10th symbol from right end id 1
^Z
[1]+  Stopped                 ./a.out
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re6.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
9000
success
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
4005
success
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
123
123fail
```

```
fail
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex blank.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
Enter the name of the file to copy:     input.txt
Enter the name of the file to write:    output.txt
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re1.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
24900
24900 string ends with 00
2352
2352 string does not end with 00
^Z
[2]+  Stopped                 ./a.out
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re2.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
12142
12142 string does not have 222
24322245
24322245 string has 222
```


mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re4.l
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
usr/bin/ld: /tmp/ccNpRHPT.o: in function `yylex':
ex.yy.c:(.text+0x33f): undefined reference to `pow'
ollect2: error: ld returned 1 exit status
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c -lm
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
01
uccessbmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c -lm
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
111
uccessbmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
1
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re5.l
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
023002245
023002245 10th symbol from right end id 1
Z
1]+ Stopped ./a.out
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re6.l
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c

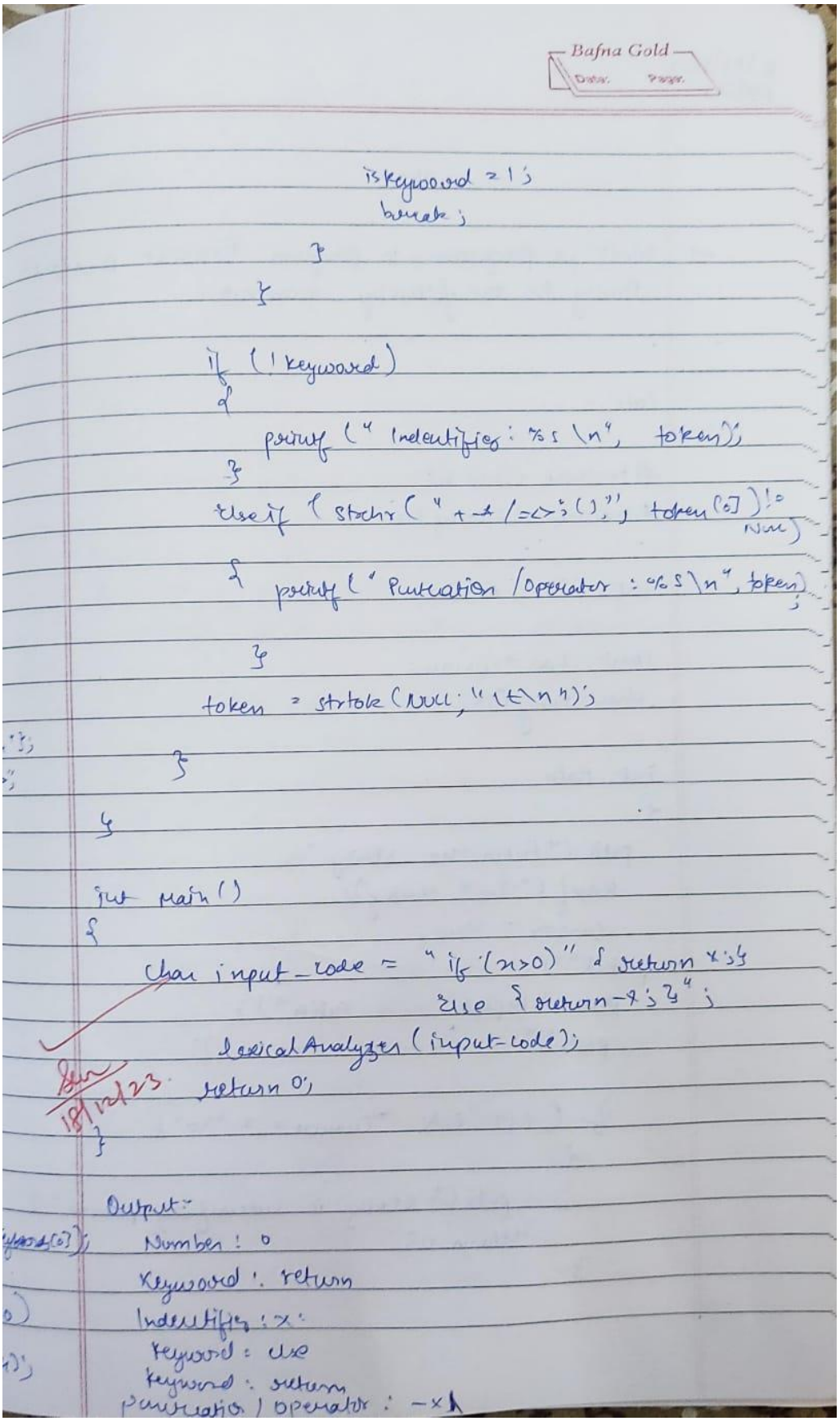
Lab-5

Q → write a program to design Lexical Analyzer in C/C++ to recognize any 5 keywords, identifiers, numbers, operators & punctuation.

Code :-

→

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

void lexicalAnalyzer (char input_code[])
{
    char *key[] = { "if", "else", "while", "for", "return" };
    char *op[] = { "+", "-", "*", "/", "=", "==", "<", ">",
                   "<=", ">=" };
    char *punct[] = { ",", ";", "(", ")", "{", "}" };
    char *token = strtok (input_code, " \t\n");

    while (token != NULL)
    {
        if (isdigit (token[0])) {
            printf ("Number: %s\n", token);
        else if (isalpha (token[0] || token[0] == '_')
        {
            int iskeyword = 0;
            for (int i=0 ; i< sizeof (keywords) / sizeof (keywords[0]);
                 ++i)
            { if (strcmp ( token, keywords[i] == 0)
                { printf ("Keywords: %s\n", token);
```



```
        isKeyword = 1;
        break;
    }
  }
  if (!keyword)
  {
    printf("Identifier: %s\n", token);
  }
  else if (strchr("+-*/=<>;(),", token[0]) != NULL)
  {
    printf("Punctuation/Operator: %s\n", token);
  }
  token = strtok(NULL, " \t\n");
 }
}

int main()
{
  char input_code = "if (x>0) { return x; } else { return -x; }";
  lexicalAnalyzer(input_code);
  return 0;
}
```

18/12/23

Output:

Number: 0
Keyword: return
Identifier: x
Keyword: else
Keyword: return
Punctuation/Operator: -x

```
enter c code
int a = 1234 ;
Keyword: int
Identifier: a
Punctuation/Operator: =
Number: 1234
Punctuation/Operator: ;
```

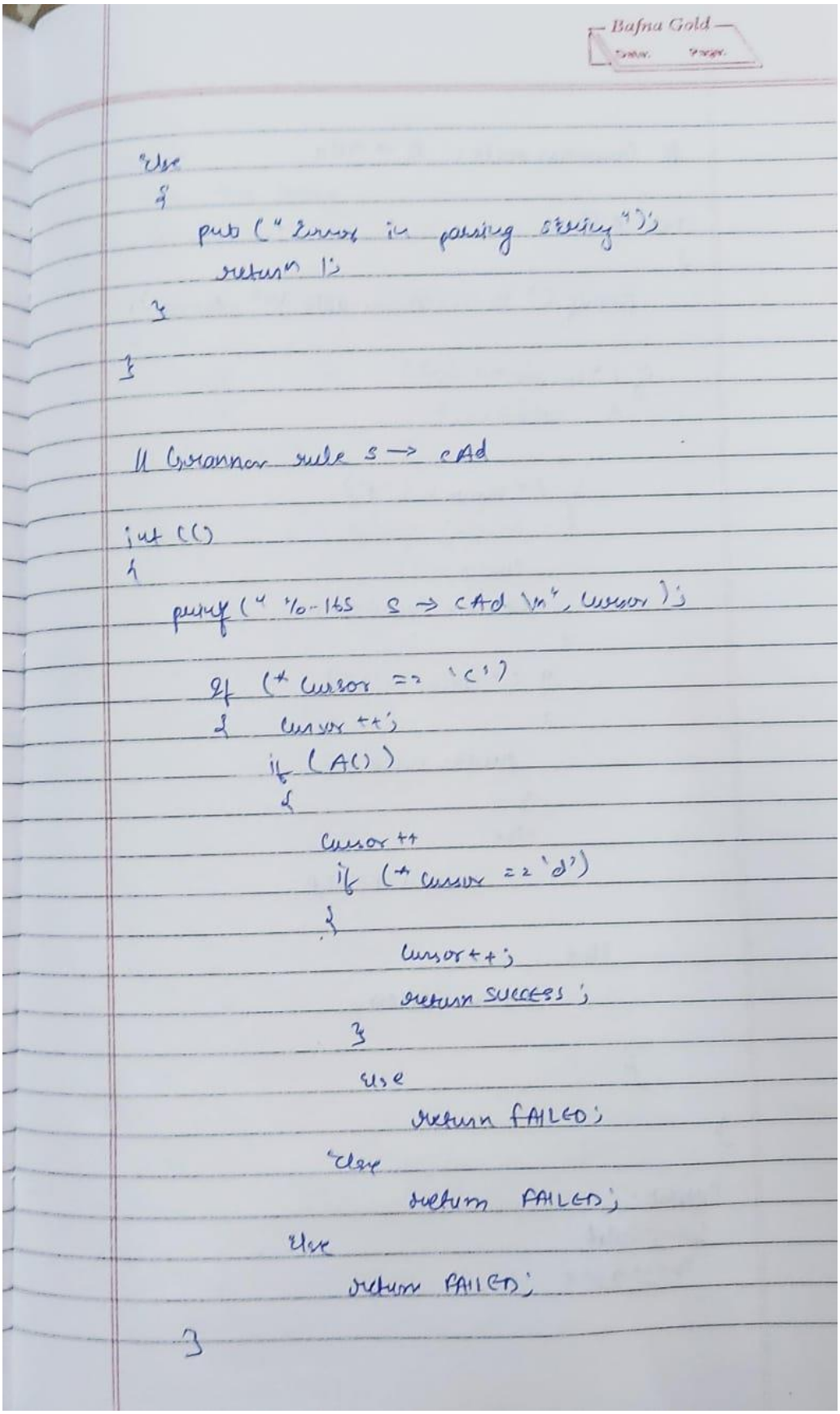
8/01/2023

Lab-6

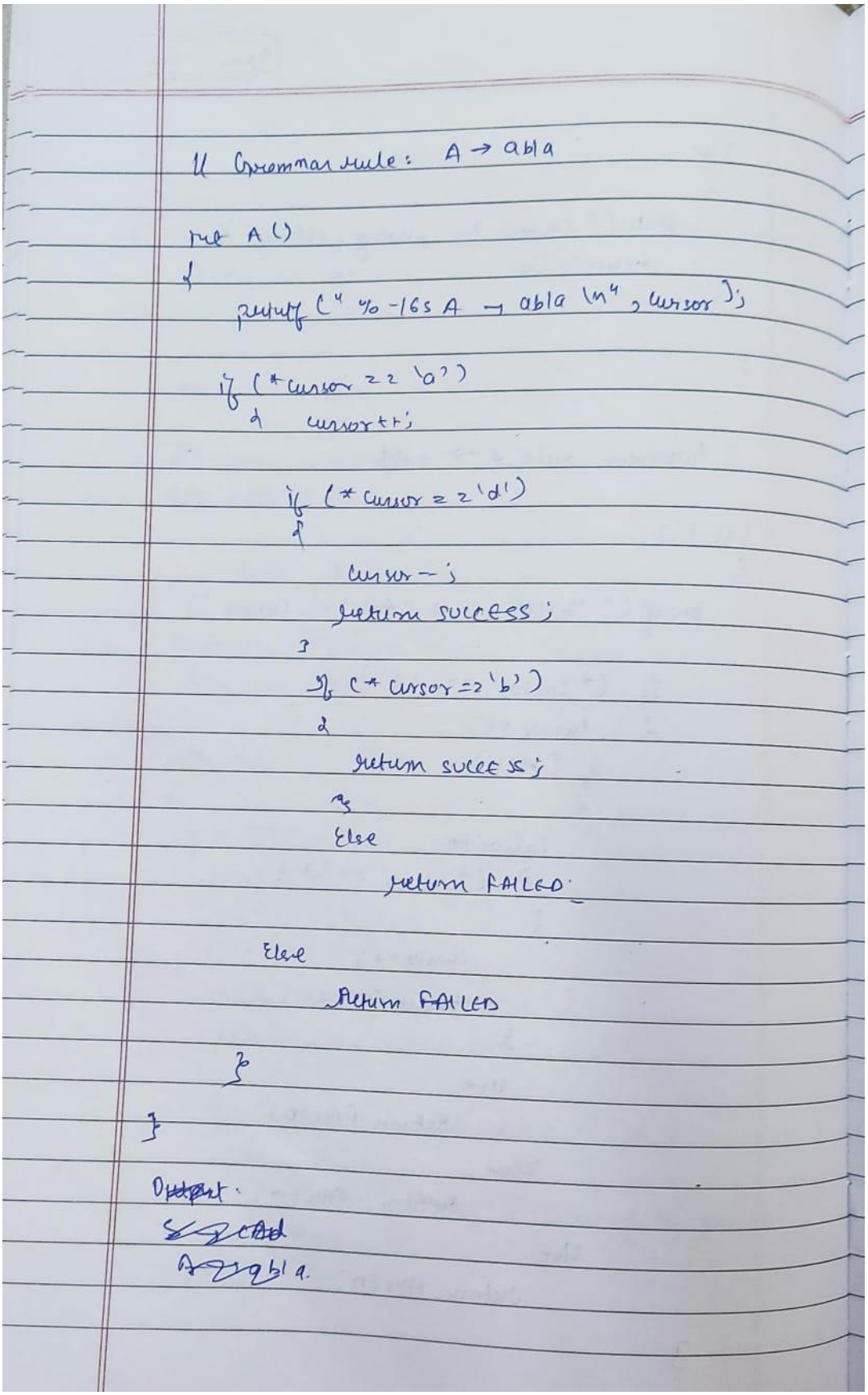
→ Write a program to perform Recursive Descent Parsing on the following grammar:

Code:-

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

int E(), A();

const char *cursor;
char string[64];

int main
{
  puts ("Enter the string");
  scanf ("%s", string);
   cursor = string;
puts("\n")
  puts("Input          Action");
  puts("--------------------");

  if (E() && *cursor == '\0')
   {
        puts ("String is successfully parsed");
        return 0;
   }
```



```
else
{
    puts("Error in parsing string");
    return 1;
}
}

// Grammar rule S -> cAd

int c()
{
    printf("%d-th S -> cAd \n", cursor);
    if (*cursor == 'c')
    {   cursor++;
        if (A())
        {
            cursor++
            if (*cursor == 'd')
            {
                cursor++;
                return SUCCESS;
            }
            else
                return FAILED;
        else
            return FAILED;
    else
        return FAILED;
}
```



// Grammar rule: A → abla

```
int A()
{
    printf("%-16s A → abla \n", cursor);
    if (*cursor == 'a')
    {
        cursor++;
        if (*cursor == 'd')
        {
            cursor--;
            return SUCCESS;
        }
        if (*cursor == 'b')
        {
            return SUCCESS;
        }
        Else
            return FAILED;
    }
    Else
        Return FAILED
}
```

Output:

~~SUCCESS~~

~~Aabla~~

Bafna Gold
Date: Page:

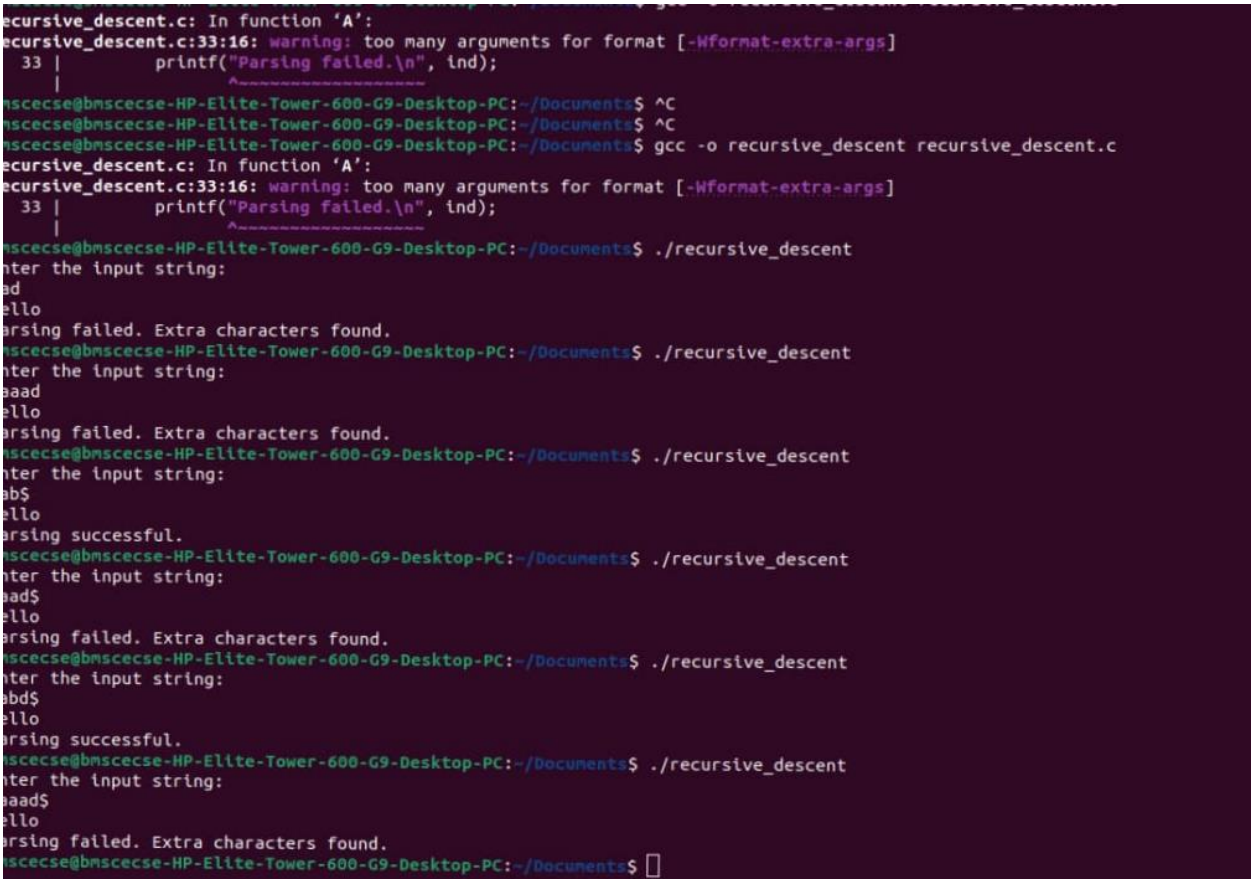
Output-

Enter the string

cabd

Input	Action
cabd	S → cAd
abd	A → abla.

String is succesfully parsed.


ecursive_descent.c: In function ‘A’:
ecursive_descent.c:33:16: warning: too many arguments for format [-Wformat-extra-args]
33 | printf("Parsing failed.\n", ind);
| ^~~~~~~~~~~~~~~~~~~~
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ^C
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ^C
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ gcc -o recursive_descent recursive_descent.c
ecursive_descent.c: In function ‘A’:
ecursive_descent.c:33:16: warning: too many arguments for format [-Wformat-extra-args]
33 | printf("Parsing failed.\n", ind);
| ^~~~~~~~~~~~~~~~~~~~
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
ad
ello
arsing failed. Extra characters found.
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
aad
ello
arsing failed. Extra characters found.
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
ab$
ello
arsing successful.
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
aad$
ello
arsing failed. Extra characters found.
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
abd$
ello
arsing successful.
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
aad$
ello
arsing failed. Extra characters found.
nscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$

a yacc program string grammar $a^n b^n$ $n \geq 5$;

Q4 Write the string matching program.

anbn.y

→

```
%{
#include <stdio.h>
#include <stdlib.h>
int yyerror(char *s);
int yylex(void);
%}

%token A
%token B
%token NL
%%
Start: A A A A A S B NL { printf("Parsed using the rule (a^n)b, n >=5. \n valid String ! \n"); }
;
S: S A
|
;
%%
void main()
{
    printf("Enter a string! \n");
    yyparse();
}

int yyerror(char *s)
{
    printf("Invalid string! \n");
    return 0;
}
```

Bafna Gold
Date: Page:

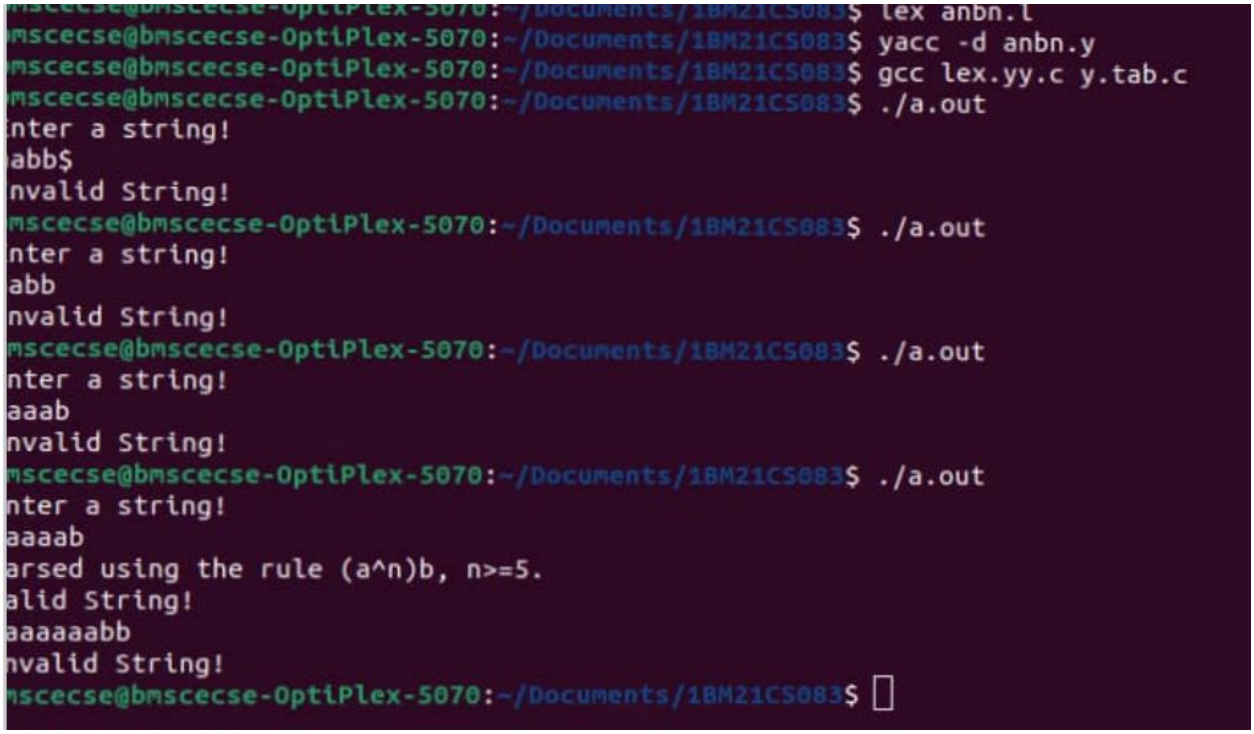
anbn.l

```
%{
#include<stdio.h>
#include<stdlib.h>
#include "y.tab.h"
extern int yylval;
%}
%%
[aA]  { yylval = yytext[0]; return A; }
[bB]  { yylval = yytext[0]; return B; }
\n    { return NL; }
.     { return yytext[0]; }
%%
int yywrap()
{
    return 1;
}
```

Output:

```
Enter a string:
aaaaab
Parsed using the rule (a^n)b , n>=5
valid string:
aabb
Invalid string!
```

29/1/2024


lex anbn.l
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ yacc -d anbn.y
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ gcc lex.yy.c y.tab.c
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
nter a string!
abb$
nvalid String!
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
nter a string!
abb
nvalid String!
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
nter a string!
aaab
nvalid String!
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
nter a string!
aaaab
arsed using the rule (a^n)b, n>=5.
alid String!
aaaaaabb
nvalid String!
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$

8/01/2023

Lab-6

→ Write a program to design parsing using YACC.

yacc code :-

```
%{
#include <stdio.h>
%}

%token NUM
%left '+'
%right '-'
%%

expr : e {printf("valid exp\n");
          printf("result : %d\n", $$); return 0;}

e : e '+' e {$$ = $1 + $3;}
  | e '-' e {$$ = $1 - $3;}
  | NUM      {$$ = $1;}
;
%%

int main()
{
  printf("\n Enter an arithmetic expression\n");
  yyparse();
  return 0;
}
```

```
int yyerror()
{
  printf("\n Invalid expression \n");
  return 0;
}

lex code:

%{ optional boilerplate
%{
  #include "y.tab.h"
%}

%%
[0-9]+ { yylval = atoi(yytext); return NUM; }
[\t];
\n return 0;
. return yytext[0];
%%
```

Output:-

Enter an arithmetic expression
5+6+3
Valid Expression
Result: 14

```
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ lex proo1.l
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ yacc -d proo1.y
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ gcc lex.yy.c y.tab.c
y.tab.c: In function ‘yyparse’:
y.tab.c:1022:16: warning: implicit declaration of function ‘yylex’ [-Wimplicit-function-declaration]
 1022 |       yychar = yylex ();
      |                ^~~~~
y.tab.c:1205:7: warning: implicit declaration of function ‘yyerror’; did you mean ‘yyerrok’? [-Wimplicit-function-declaration]
 1205 |       yyerror (YY_("syntax error"));
      |       ^~~~~~~
      |       yyerrok
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./a.out

Enter an arithmetic expression
5+6
Valid expression
Result : 11
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./a.out

Enter an arithmetic expression
5*6-2
Valid expression
Result : 28
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./a.out

Enter an arithmetic expression
5-6+*

Invalid expression
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$
```

9/01/2024

Lab-1

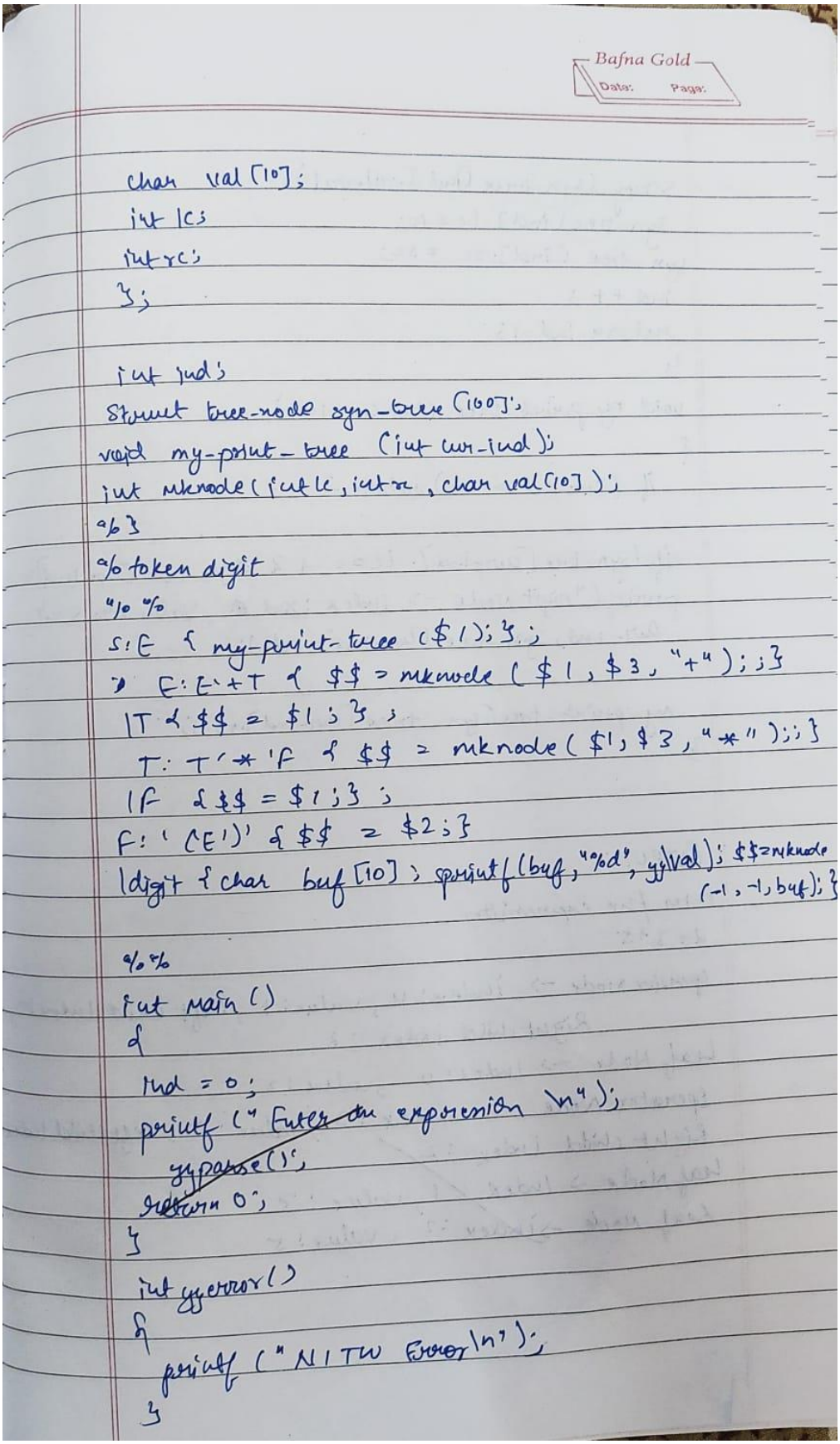
Q1 Write a Yacc program to generate syntax tree for a given arithmetic expression.

⇒ P1.l

```
%{
#include "y.tab.h"
  extern int yylval;
%}
%%
[0-9]+ { yylval = atoi(yytext); return digit; }
[\t];
[\n] return 0;
. return yytext[0];
%%
int yywrap()
{
}
```

⇒ p1.y

```
%{
#include <math.h>
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>

Struct tree-node
{
```


Bafna Gold
Date: Page:
char val[10];
int lc;
int rc;
};
int ind;
Struct tree-node syn-tree[100];
void my-print-tree (int cur-ind);
int mknode(int lc, int rc, char val[10]);
%}
% token digit
%%
S: E { my-print-tree ($1); };
E: E '+' T { $$ = mknode ($1, $3, "+"); }
| T { $$ = $1; };
T: T '*' F { $$ = mknode ($1, $3, "*"); }
| F { $$ = $1; };
F: '(' E ')' { $$ = $2; }
| digit { char buf[10]; sprintf(buf, "%d", yylval); $$=mknode
(-1, -1, buf); }
%%
int main()
{
ind = 0;
printf("Enter an expression \n");
yyparse();
return 0;
}
int yyerror()
{
printf("NITW Error\n");
}

```
strcpy (syn_tree[ind].val, val);
syn_tree[ind].lc = lc;
syn_tree[ind].rc = rc;
ind++;
return ind-1;
}
void my_print_tree (int cur_ind)
{
  if (cur_ind == -1) return;

  if (syn_tree[cur_ind].lc == -1 && syn_tree[cur_ind].rc == -1)
  printf("Digit Node -> Index : %d , Value : %s \n",
    cur_ind, syn_tree[cur_ind].val);

  my_print_tree(syn_tree[cur_ind].rc);
}
```

OUTPUT:

Enter the expression

2+3*5

Operator Node -> Index : 4, value : +, Left child Index : 0,
Right child Index : 3

Leaf Node -> Index : 0, value : 2

Operator Node -> Index : 3, value : *, Left child Index :

Right child Index : 2

Leaf Node -> Index : 1, value : 3

Leaf Node -> Index : 2, value : 5

```
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./a.out
Enter an expression
4+6*9
Operator Node -> Index : 4, Value : +, Left Child Index : 0,Right Child Index : 3
Digit Node -> Index : 0, Value : 4
Operator Node -> Index : 3, Value : *, Left Child Index : 1,Right Child Index : 2
Digit Node -> Index : 1, Value : 6
Digit Node -> Index : 2, Value : 9
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$
```

Bafna Gold
Date: Page:

Q2 Use YACC to convert : infix expression to Postfix expression.

⇒ P4.1

```
%{
#include "y.tab.h"
 extern int yylval;
%}
%.%.
[0-9]+ { yylval = atoi(yytext; return digit;}
 [\+];
 [\n] return 0;
. return yytext[0];
%%
 int yywrap()
{
}
```

P4.y

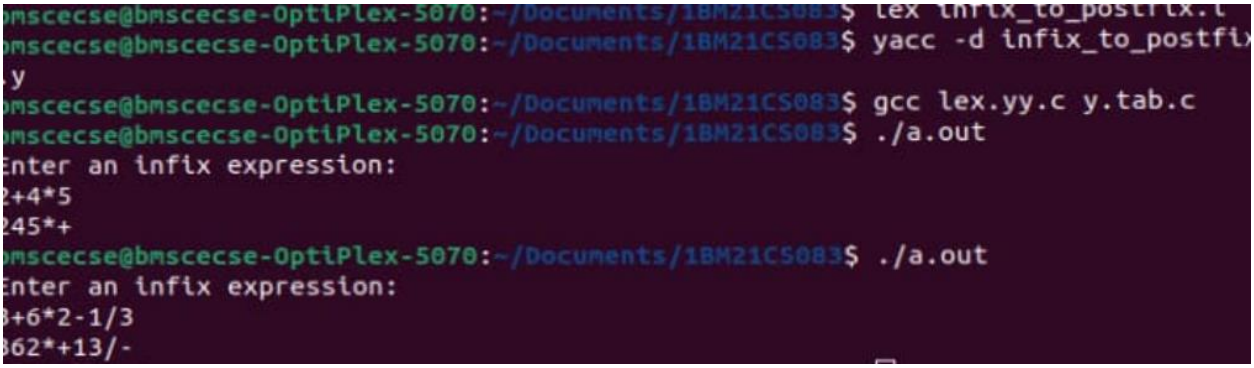
```
%{
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>
%}
%token digit

%%

S: E {printf("\n\n");};
```

```
E : E '+' T {printf ("+");};
| T
T : T '*' F {printf ("*");};
| F ;
F : '(' E ')'
| digit {printf ("%d", $1);}
;
%%

int main()
{
    printf ("Enter infix expression");
    yyparse();
}
yyerror();
{
    printf ("Error");
}
```

Output :-

Enter infix expression 2+6*3+4

2 6 3 * + 4 +


bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex infix_to_postfix.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ yacc -d infix_to_postfix
.y
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ gcc lex.yy.c y.tab.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
Enter an infix expression:
2+4*5
245*+
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
Enter an infix expression:
8+6*2-1/3
862*+13/-

Bafna Gold
Date: Page:

Q#3 Use YACC to generate 3-Address Code for a given expression

→ Pl.l

```
d [0-9]+
a [a-zA-Z]+
%{
#include <stdio.h>
#include <stdlib.h>
#include "y.tab.h"
extern int yylval;
extern char iden[20];
%}
%%
{d} {yylval = atoi(yytext); return digit;}
{a} {strcpy(iden, yytext); yylval = 1; return id;}
[\t] {;}
\n return 0)
. return yytext[0];
%%
int yywrap()
{
}
```

Pl. y

```
%{
#include <math.h>
#include <ctype.h>
#include <stdio.h>
int var_cnt = 0;
char iden[20];
%}
```

```
%{ token idigit
%%
S: id '=' E { printf ("%s = t%d\n", iden, var_cnt-1); }
E: E '+' T { $$ = var_cnt; var_cnt++;
             printf("t%d = t%d + t%d ;\n", $$, $1, $3);
           }
 | E '-' T { $$ = var_cnt; var_cnt++; printf ("t%d =
             t %d - t%d ; \n", $$, $1, $3);
           }
 | T { $$ = $1; };

T: T '*' F { $$ = var_cnt; var_cnt++; printf ("t%d
             = t%d * t%d ; \n", $$, $1, $3); }

P : '(' E ')' { $$ = $2; }
 | digit { $$ = var_cnt; var_cnt++; printf ("t%d =
             %d ; \n", $$, $1); } ;
%% %%

int main()
{
   var_cnt = 0;
   printf ("Enter an Expression : \n");
   yyparse();
   return 0;
}
yyerror()
{
  printf ("error");
}
```

Bafna Gold
Date: Page:

Output: -

Enter an expression:

a = 2+3*6

t0 = 2;

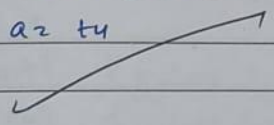
t1 = 3;

t2 = 6;

t3 = t1 * t2

t4 = t0 + t3;

a = t4



```
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex 3addcode.l
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ yacc -d 3addcode.y
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ gcc lex.yy.c y.tab.c
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
nter an expression:
=8+9-2
0 = 8;
1 = 9;
2 = t0 + t1;
3 = 2;
4 = t2 - t3;
=t4
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
nter an expression:
=2^3/23+5
 = 2;
 = 3;
 = t0 ^ t1;
 = 23;
 = t2 / t3;
 = 5;
 = t4 + t5;
t6
```