POORNIMA

Experiment – 1

Q WAP to identify weather given string is keyword or not.

```
#include<stdio.h>
#include<string.h>
Void main()
{
char arr[100][10] = {"if", "else", "elseif", "struct",
                     "break", "false", "true", "for",
                     "while"};
Char n[10];
int f=0;
printf("enter the keyword \n");
Scanf("%s", &n);
for (int i=0; i<10; i++)
{
  if (strcmp(arr[i],n)==0)
    {
      f=1;
    }}
  if (f==1)
  printf("keyword is present \n");
  else
  printf("keyword is not present \n");
}
```

output exp. 1

Enter the keyword
false
keyword is Present

O/P :-

Content of this file are
file is created

POORNIMA

Revision of file Handelling

Q W.AP in C to Read data from existing file.

```
#include<stdio.h>
Void main()
{
FILE *file;
Char a;
file = fopen("mk.txt","r+");
if (NULL == file)
{
printf("file can't be opened\n");
}
else printf("Content of this file are\n");
while (!FEof(file))
{
a = fgetc(file);
printf("%c",a);
}
fclose(file);
}
```

Sunita
20/Sept/2023

output :-

enter the filename filename.txt
else
total keyword 1

Lab - 2

POORNIMA

Experiment ②

Q → WAP in C to count total no. of keywords in a file.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int iskeyword (char word[])
{
char key[32][10] = {"for", "if", "else", "goto", "false",
                    "break"};
for (int i=0; i<32; i++) {
if (strcmp(key[i], word)==0) {
return 1;
}} return 0;
} int main() {
char filename[100];
char word[20];
int count 0;
printf ("enter the filename");
scanf ("%s", filename);
FILE *f;
f = fopen (filename, "r");
if (f == NULL)
{ printf ("the file is empty\n");
return 0;
}
```

PROGRAM

```
while (fscanf(f, "%s", word) != EOF)
{
if (isKeyword(word))
{
count++;
printf("%s\n", word);
}
}
fclose(f);
printf("total keyword %d", count);
return 0;
}
```

O/P :-

enter the filename
filename.txt
-
-
*
%
+
+
+
*
total operators 8

POORNIMA

Lab - 3
Experiment - ①

// WAP in C to count total no. of operators in a file.

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int isKeyword(char word[])
{
char key[32][10]={"+","-","%","*","++","--"};
for(int i=0; i<10; i++)
{
if(strcmp(key[i],word)==0)
{
return 1;
}}
return 0;
}
int main()
{
char filename[200];
char word[20];
int count=0;
printf("enter the filename\n");
scanf("%s",filename);
FILE *f;
f=fopen(filename,"r");
if(f==NULL)
{
```

POORNIMA

```
printf("file is empty\n");
return 0;
}
while (fscanf(f, "%c", word) != EOF)
{
count++;
printf("%s\n", word);
}
}
fclose(f);
printf("total operators %d", count);
return 0;
}
```

Tuesday
3/Oct/2023

O/P :-

enter char to find : a
total number of a is 1

POORNIMA

Lab-4
Experiment 1

Q WAP in C to count total occurrence of given character in a file

```
#include<stdio.h>
int main()
{
FILE *file;
int count=0;
char ch;
file=fopen("nk.txt","r");
if (file==NULL)
{ printf("file can not opened\n");
}
printf("Enter char to find:-");
scanf("%c",&ch);
while(!feof(file))
{
if (ch==getc(file))
{ count++;
}
}
printf("total number of %c is %d\n",ch,count);
return 0;
}
```

10/10/2023.

o/p :-

Expression terminated by $: a+b+c=d.$

Given Expression : a+b+c=d

Symbol table

Symbol	addr	type
a	8390704	identifier
+	8390718	operator
b	8390815	identifier
+	8390625	operator
c	8390821	identifier
=	8390706	operator
d	8390717	identifier

POORNIMA

Lab 5

Q WAP in C to insert and display the entries in the Symbol table.

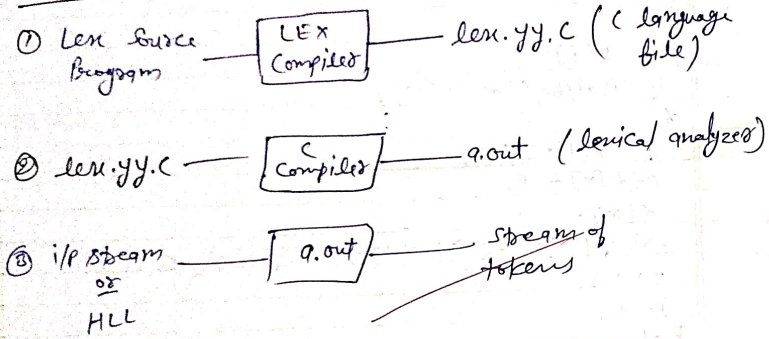
```
#include<stdio.h>
#include<ctype.h>
#include<stdlib.h>
#include<string.h>
#include<math.h>
Void main()
{ int i=0, j=0, x=0, n;
  Void *p, *add[5];
  Char *ch, Sch, b[15], d[15], c;
  printf("Expression terminated by $:");
  while((c=getchar())!='$')
  {
    b[i]=c;
    i++;
  }
  n= i-1;
  Printf("Given Expression:");
  i=0;
  while(i<=n)
  {
  Printf("%c", b[i]);
  i++;
  }
  Printf("\n Symbol table \n");
```

POORNIMA

```
printf("Symbol\t addr\t type");
while (j<=n)
{
  c = b[j];
  if (isalpha(toascii(c)))
  {
    p= malloc(c);
    add[x] = p;
    d[x] = c;
    printf("\n%c \t %d \t identifier\n", c, p);
    x++;
    j++;
  } else
  { ch=c;
  if(ch=='+'|| ch=='-'|| ch=='*'|| ch=='=')
  {  p= malloc(ch);
     add[x] = p;
     d[x] = ch;
     printf("\n%c \t %d \t operator\n", ch, p);
     x++;
     j++;
  }}}
```

Output
17/Oct/2023

Block Diagram :-



^a → Starting with a
$a → ending with a

Execution
lex: file.l
gcc lex.yy.c
./a.out

POORNIMA

Lab-6
Experiment 1

① Lex :- It is a tool of S/w which automatically generates a lexical analyzer.

→ A lex program consist of 3 parts & these parts are seperated by %%. (delimiters)

① Declaration
%%. (delimeters)

② Translation Rules → Pattern
%%. → Action

③ Auxiliary Procedures / functions
yywrap ()
yylex ()

※ yywrap() :- This function is called by lex tool when i/p is exhausted, It always return 1, when execution is complete

※ yylex() :- This function read the input stream and generate tokens acc. to the expressions.

Page No.....

o/p :-

Enter the i/p
Hi
Bise
Hiii
ERROR!!

o/p

Enter the i/p
Hello
mixed
hello
lowercase
HELLO
uppercase
99
number

POORNIMA

① Example :- (Basic Example)

```
%{
#include<stdio.h>
%}
%%
"Hi" {printf("Bise");}
.* {printf("Errors!");}
%%
void main()
{printf("Enter the i/p \n");
yylex();
}
int yywrap(){
return 1;
}
```

② Example

```
%{
#include<stdio.h>
%}
%%
[a-z]+ {printf("lowercase");}
[A-Z]+ {printf("Uppercase");}
[a-z A-Z]+ {printf("mixed");}
[0-9]+ {printf("number");}
.* {printf("ERROR!");}
%%
```

o/p:-

Enter the i/p
5
odd
6
even
22
even
43
odd.

POORNIMA

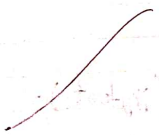
```
Void main(){
printf("Enter the i/p \n");
 yylex();
}
int yywrap(){
   return 1;
}
```

(3) Example :-

```
%{
#include<stdio.h>
%}
%%
[0-9]+{x= atoi(yytext);
      if(x%2==0)
            printf("Even\n");
        else
            printf("Even\n");
        else
            printf("odd"\n"); }
    .* {printf("ERROR !"); }
%%
Void main()
printf("enter the i/p \n");
  yylex();
}
  int yywrap(){
  return 1;
}
```

o/p :-

Enter the input
apple
Start with vowel
Hii
Wrong Input

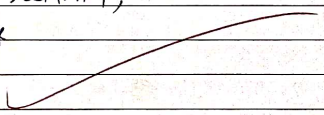


POORNIMA

Lab – 7

// Check weather a name start with vowel or not. :- a

```
%{
#include <stdio.h>
%}
%%
^[aeiou AEIOU][A-Z]* { printf("start with vowel\n"); }
.* { printf("wrong input"); }
%%
int main()
{ printf("Enter the input");
  yylex();
} int yywrap()
{
  return 1;
}
```



o/p :- Enter a mobile Number
48499
wrong input
9460285111
valid
0123456789
wrong input:

Lab – 8

POORNIMA

Experiment – 1

```
// check weather a phone number is valid or not
#include<stdio.h>
%{
%}
^[0] { printf("invalid\n"); }
^[1-9]{10} { printf("valid"); }
.* { printf(" wrong Input "); }
%%
int main()
{
printf("Enter a mobile Number\n");
yylex();
}
int yywrap()
{
return 1;
}
```

Expe – (2)

```
// check weather it is valid identifier or not
%{
#include<stdio.h>
%}
%%
[ ] * { printf("invalid\n"); }
^[0-9][a-zA-Z0-9]* { printf("invalid\n"); }
^[a-zA-Z][a-zA-Z0-9]* { printf("valid"); }
* { printf("wrong input"); }
%%
```

o/p :-

Enter a input

invalid
_abc
valid
a
valid

POORNIMA

```
Void main()
{
printf("Enter a input");
 yylex();
{
 int yywrap()
{
 return 1;
}
```

o/p :-

Enter a input
https://www.abc.com
Valid
https://www.abc.com?/.
Valid
https//www.abc.com
Wrong i/p.

POORNIMA

Lab - 9

Expt - 1

// write a lex program to check weather the given text is a valid input.

```
%{
#include<stdio.h>
%}
%%
((http)|(https))"://"(www.)[A-Z]*[\.][a-z]{2,3}[A-Z0-9%/=]* {printf("valid\n");}
.* {printf("wrong input\n");}
%%
void main()
{ printf("Enter a input\n");
  yylex();
}
int yywrap()
{
 return 1;
}
```

O/P :-

Enter a input
12 : 45 : 59
Valid
13 : 00 : 23
wrong input
04 : 30 : 60
valid input

POORNIMA Exp-(2)

11 write a lex program to check weather a given time is valid.

```
%{
#include<stdio.h>
%}
%%
([0][1-9]|[1][0-2]):[0-5][0-9]:
[0-5][0-9]{printf("valid\n");}
.*{printf("Wrong input\n");}
%%
void main()
{
printf("Enter a input\n");
yylex();
}
int yywrap()
{
return 1;
}
```

Switch Decision

Page No....