

**RAJASTHAN INSTITUTE OF ENGINEERING AND TECHNOLOGY, JAIPUR**

**Lab Record**

**Compiler Design**

**Submitted To: Submitted By:**

Mr. Mukesh Choudhary Vicky Kumar

Roll No. 18ERECS080

CSE (3rd year)



RAJASTHAN TECHNICAL UNIVERSITY, KOTA

**Syllabus**

**III Year-V Semester: B.Tech. Computer Science and Engineering**

**Compiler Design**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S.no** | **Name of Experiment** | **Date** | **Attenance(10)** | **Perfor-**  **mance**  **(10)** | **Rec**  **ord**  **(10)** | **Viva**  **(20)** | **Total**  **(50)** |
| 1. | To identify whether a given string is a keyword or not. |  |  |  |  |  |  |
| 2. | To count total no. of keywords in a file. |  |  |  |  |  |  |
| 3. | To count the total no. of operators in a file. |  |  |  |  |  |  |
| 4. | Count the total occurrence of each character in a given file. |  |  |  |  |  |  |
| 5. | To write a program for implementing Symbol Table. |  |  |  |  |  |  |

**EXPERIMENT NO. 1**

1.1 Aim :- To identify whether a given string is a keyword or not.

1.2 Resource – turbo c++

1.3 PROGRAM LOGIC:

1. Read the file from user
2. Check whether the file exist or not .
3. Check whether the input is keyword or string.
4. Print that result.

1.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program

1.5 PROGRAM:

#include <stdio.h>

#include <string.h>

int main() {

char keyword[32][10]={

"auto","double","int","struct","break","else","long",

"switch","case","enum","register","typedef","char",

"extern","return","union","const","float","short",

"unsigned","continue","for","signed","void","default",

"goto","sizeof","voltile","do","if","static","while"

} ;

char str[]="which";

int flag=0,i;

for(i = 0; i < 32; i++) {

if(strcmp(str,keyword[i])==0) {

flag=1;

}

}

if(flag==1)

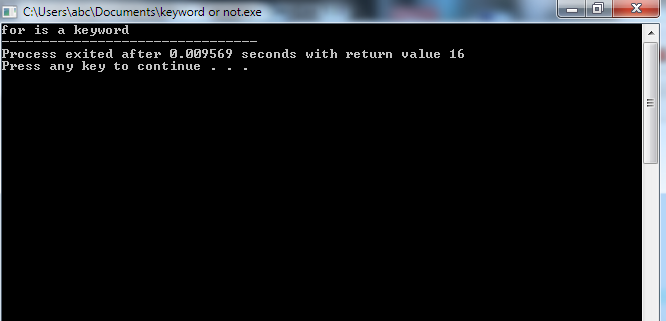
printf("%s is a keyword",str);

else

printf("%s is not a keyword",str);

}

Output :



**EXPERIMENT NO. 2**

1.1 Aim :- To count total no. of keywords in a file.

1.2 Resource – turbo c++

1.3 PROGRAM LOGIC:

1. Read the file from user
2. Check whether the file exist or not .
3. Count the total keyword present.
4. Print that result.

1.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program

1.5 PROGRAM:

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <conio.h>

#include<string.h>

int main()

{

char file[50] = "access modifier in class.cpp";

printf("Enter File name : ");

gets(file);

printf("\n");

FILE \* fp = fopen(file, "r");

FILE \* kout = fopen("keyword.txt", "w");

FILE \* iout = fopen("identifier.txt", "w");

if (fp == NULL)

{

printf("\n'%s' file not found...\n", file);

getch();

return 1;

}

int k, result, count, kcount;

char c, str[10];

kcount = count = 0;

char keywords[][10] = {"auto", "break", "case", "char", "const", "continue", "default","do", "double", "else", "enum", "extern", "float", "for", "goto", "if", "int","long", "register", "return", "short", "signed", "sizeof", "static", "struct","switch", "typedef", "union", "unsigned", "void", "volatile", "while" };

while((c = fgetc(fp)) != EOF)

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

{ ++count;

// printf("%d). %s\n", count, str);

for(k=0; k<32; k++)

{

result = strcmp(keywords[k], str);

if (result == 0)

{

++kcount;

printf(str);

printf("\n");

break;

}

}

if (result == 0)

fprintf(kout, "%s\n", str);

else

fprintf(iout, "%s\n", str);

strcpy(str,"");

}

else

{

if (isalpha(c) && c!=' ')

strncat(str, &c, 1);

}

}

fclose(fp);

fclose(kout);

fclose(iout);

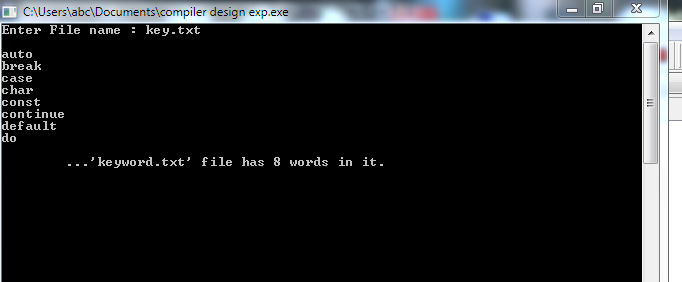
printf("\n\t...'%s' file has %d words in it.\n", "keyword.txt", kcount);

getch();

return 0;

}

Output :



**EXPERIMENT NO. 3**

1.1 Aim :- To count total no. of operators in a file.

1.2 Resource – turbo c++

1.3 PROGRAM LOGIC:

1. Read the file from user
2. Check whether the file exist or not .
3. Count the total operator present.
4. Print that result.

1.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program

1.5 PROGRAM:

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

int main()

{

char file[50];

printf("Enter File name : ");

gets(file);

printf("\n");

FILE \* fp = fopen(file, "r");

if (fp == NULL)

{

printf("\n'%s' file not found...\n", file);

getch();

return 1;

}

int k, result, count, kcount;

char c, str[10];

kcount = count = 0;

while((c = fgetc(fp)) != EOF)

{ if(c == '+' || c == '-' || c == '%' || c == '\*' || c == '/')

++count;

}

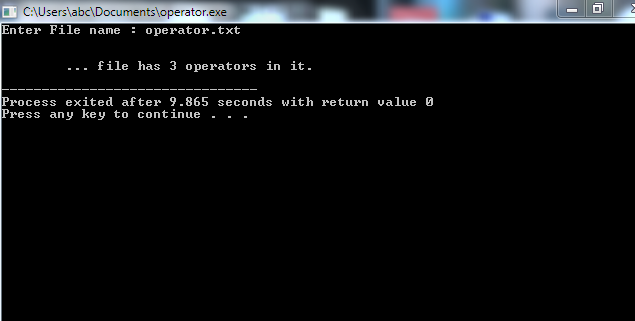
fclose(fp);

printf("\n\t... file has %d operators in it.\n", count);

return 0;

}

Output :



**EXPERIMENT NO. 4**

1.1 Aim :- Count total occurrence of each character in a given file.

1.2 Resource – Dev c++

1.3 PROGRAM LOGIC:

1. Read the file from user
2. Check whether the file exist or not .
3. Count the total character present.
4. Print that result.

1.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program

1.5 PROGRAM:

#include <stdio.h>

#define MAX\_FILE\_NAME 100

int main()

{

FILE\* fp;

int count = 0;

char filename[MAX\_FILE\_NAME];

// To store a character read from file

char c;

printf("Enter file name: ");

scanf("%s", filename);

fp = fopen(filename, "r");

if (fp == NULL) {

printf("Could not open file %s",

filename);

return 0;

}

// Extract characters from file

// and store in character c

for (c = getc(fp); c != EOF; c = getc(fp))

// Increment count for this character

count = count + 1;

fclose(fp);

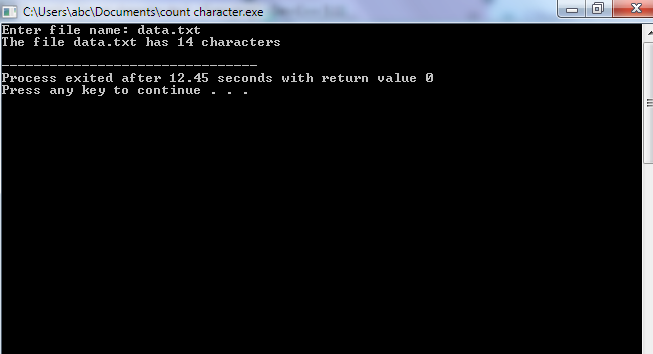
printf("The file %s has %d characters\n ",

filename, count);

return 0;

}

Output :



**EXPERIMENT NO. 5**

1.1 Aim :- To write a program for implementing Symbol Table.

1.2 Resource – turbo c++

1.3 PROGRAM LOGIC:

Step1: Start the program for performing insert, display, delete, search and modify option in symbol table  
Step2: Define the structure of the Symbol Table  
Step3: Enter the choice for performing the operations in the symbol Table  
Step4: If the entered choice is 1, search the symbol table for the symbol to be inserted. If the symbol is  
already present, it displays “Duplicate Symbol”. Else, insert the symbol and the corresponding address in  
the symbol table.  
Step5: If the entered choice is 2, the symbols present in the symbol table are displayed.  
Step6: If the entered choice is 3, the symbol to be deleted is searched in the symbol table.  
Step7: If it is not found in the symbol table it displays “Label Not found”. Else, the symbol is deleted.  
Step8: If the entered choice is 5, the symbol to be modified is searched in the symbol table.

1.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program

1.5 PROGRAM:

//Implementation of 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,srch,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");  
 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 :

