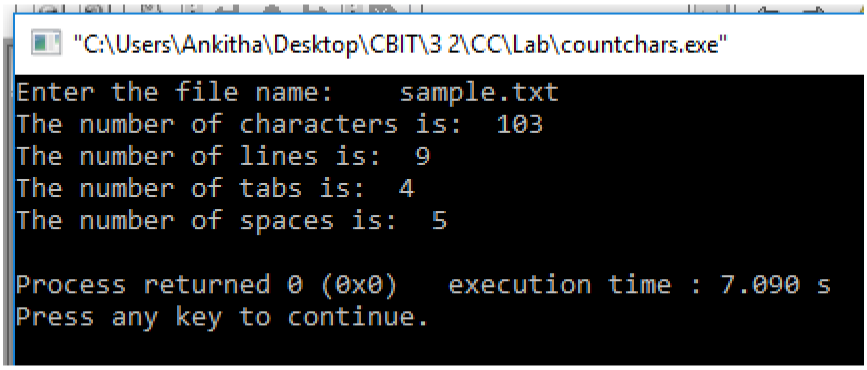
Program:

```
#include<iostream>
#include<fstream>
#include<cstring>
using namespace std;
int main(){
  int chars=0, tabs=0, spaces=0, lines=0;
  string filename,str;
  cout<<"Enter the file name:\t";
  cin>>filename;
  ifstream inFile;
  inFile.open(filename.c_str());
  while(getline(inFile,str)){
    for (int i=0; i<str.length(); i++){
      if(str[i]=='\t')      tabs++;
      else  if(str[i]==' ') spaces++;
      chars++;
    }
    lines++;
  }
  inFile.close();
  cout<<"The number of characters is:  "<<chars<<"\nThe number of lines is:  "<<lines<<endl;
  cout<<"The number of tabs is:  "<<tabs<<"\nThe number of spaces is:  "<<spaces<<endl;
  return 0;
}
```

Output:

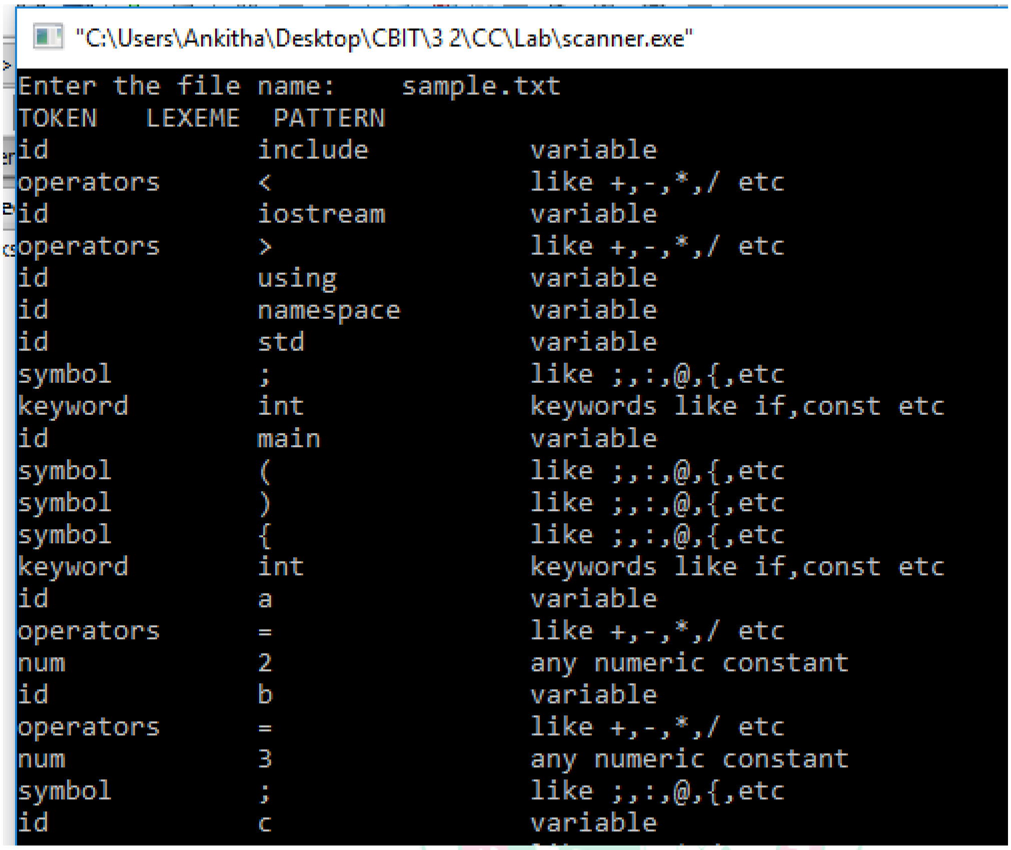


"C:\Users\Ankitha\Desktop\CBIT\3 2\CC\Lab\countchars.exe"

```
Enter the file name:    sample.txt
The number of characters is:  103
The number of lines is:  9
The number of tabs is:  4
The number of spaces is:  5

Process returned 0 (0x0)   execution time : 7.090 s
Press any key to continue.
```

Output for SCANNER:



"C:\Users\Ankitha\Desktop\CBIT\3 2\CC\Lab\scanner.exe"

```
Enter the file name:    sample.txt
TOKEN    LEXEME   PATTERN
id              include           variable
operators       <                 like +,-,*,/ etc
id              iostream          variable
operators       >                 like +,-,*,/ etc
id              using             variable
id              namespace         variable
id              std               variable
symbol          ;                 like ;,:,@,{,etc
keyword         int               keywords like if,const etc
id              main              variable
symbol          (                 like ;,:,@,{,etc
symbol          )                 like ;,:,@,{,etc
symbol          {                 like ;,:,@,{,etc
keyword         int               keywords like if,const etc
id              a                 variable
operators       =                 like +,-,*,/ etc
num             2                 any numeric constant
id              b                 variable
operators       =                 like +,-,*,/ etc
num             3                 any numeric constant
symbol          ;                 like ;,:,@,{,etc
id              c                 variable
```

Program:

```
#include<iostream>
#include<fstream>
#include<cstring>
#include<iomanip>
#include<sstream>
#include<unordered_set>
#include<vector>
using namespace std;
unordered_set<string> keywords = {"auto","break","case","char","const","continue","default","do",
        "double","else","enum","extern","float","for","if","int","long","register","return","short",
        "signed","sizeof","static","struct","switch","typedef","unsigned","void","while"};
unordered_set<string> operators = {"+","-","=","<",">","*","%","/"};
unordered_set<string> symbols = {";","{","}","(",")","]","@","!","["};

class output{
public:    string token,lexeme,pattern;
           output (string a,string b,string c){token=a;lexeme=b;pattern=c;} //constructor
};
int main () {
  string filename;
  cout<<"Enter the file name:\t";
  cin>>filename;
  ifstream inFile;
  inFile.open(filename.c_str());
  string str;
  vector<output> v;
  while (getline( inFile, str )){
        int len=str.length(),i=0;
        do {
                char c=str[i];
                string identifier="", num="", target="";
                stringstream ss;
                bool word=false, number=false;
                while((c>=65 && c<=90)||(c>=97 && c<=122)){
```

```
                identifier+=c; c=str[++i];

                if(i+1==len) break;

                word=true;

            }

            if(word){

                if(keywords.find(identifier)==keywords.end())
                    v.push_back(output("id",identifier,"variable"));

                else    v.push_back(output("keyword",identifier,"keywords like if,const etc"));
            }

            while(c>=48 && c<=57){

                num+=c;         c=str[++i];

                if(i+1==len) break;

                number=true;

            }

            if(number)      v.push_back(output("num",num,"any numeric constant"));

            ss<<c;          ss>>target;

            if( operators.find (target) != operators.end())
                            v.push_back(output("operators",target,"like +,-,*,/ etc"));

            if( symbols.find(target) != symbols.end() )
                                v.push_back(output("symbol",target,"like ;,:,@,{,etc"));

            if(c=='"'){

                do{

                    if(i+1==len) break;         c=str[++i];

                }while(c!='"');

            }

            i++;

        }while(i<len);

    }

    cout<<"TOKEN\tLEXEME\tPATTERN\n";

    vector<output>::iterator itr;

    for (itr= v.begin(); itr != v.end(); ++itr){

      output temp=*itr;

      cout <<left<<setw(15)<<temp.token <<left<<setw(15)<<temp.lexeme<<"\t"<<temp.pattern<<endl;

    }

    return 0;

}
```