**Token:**

In every Natural languages, there are some grammatical rules to write something. Like human language, C has some rules to write code properly. These rules are known as token.

C tokens are of six types-

1. **Keywords:**  
   A variable is a meaningful name of data storage location in computer memory. When using a variable you refer to memory address of computer.  
   Example: do,while,for
2. **Identifiers:**  
   The term identifier is usually used for variable names.  
   Example: main,A,AB
3. **Constants:**  
   Constants are expressions with a fixed value.  
   Example: 1,23,0
4. **Strings:**  
   Sequence of characters.  
   Example: “Ok”, “%d %d”
5. **Special symbols:**  
   Symbols other than the Alphabets and Digits and white-spaces.

Example: (), {},

1. **Operators:**  
   A symbol that represent a specific mathematical or non-mathematical action.  
   Example: +, /,-,\*

**Source Code:**

#include<bits/stdc++.h>

using namespace std;

template <class T> void print\_vector(T v)

{

int sz=v.size()-1;

for(int i=0;i<=sz;i++)cout<<v[i]<<" \n"[i==sz];

}

class Token

{

private:

map<string,bool>token;

map<char,bool>operators;

string code;

set<string>Operators,keywords,Identifiers,strings,constants;

set<char>specialSymbol;

void init();

void process();

bool isKeyWord(string str);

bool isOperator(char str);

bool isSpecialSymbol(char str);

public:

Token(string file\_name);

vector<string>getKeywords();

vector<string>getIdentifires();

vector<string>getStrings();

vector<string>getConstants();

vector<string>getOperators();

vector<char>getSpecialSymbol();

};

void Token::init()

{

FILE \*fp=fopen("keyword.txt","r");

while(!feof(fp))

{

char tk[100];

fscanf(fp,"%s",tk);

token[tk]=true;

}

fclose(fp);

fp=fopen("operator.txt","r");

while(!feof(fp))

{

char c[3];

fscanf(fp,"%s",c);

operators[c[0]]=true;

}

fclose(fp);

}

bool Token::isKeyWord(string str)

{

return token.find(str)!=token.end();

}

bool Token::isOperator(char str)

{

return operators.find(str)!=operators.end() and str!=' ';

}

void Token::process()

{

for(int i=0; i<code.size(); i++)

{

string str="";

if(isdigit(code[i]))

{

while(isdigit(code[i]))

{

str+=code[i];

i++;

}

constants.insert(str);

i--;

}

else if(code[i]=='\"')

{

str+='\"';

i++;

while(code[i]!='\"')

{

str+=code[i];

i++;

}

str+='\"';

strings.insert(str);

}

else if(isalpha(code[i]) || code[i]=='\_')

{

while(isalpha(code[i])||isdigit(code[i])|| code[i]=='\_')

{

str+=code[i];

i++;

}

if(isKeyWord(str))keywords.insert(str);

else Identifiers.insert(str);

i--;

}

else

{

if(isOperator(code[i]))

{

while(isOperator(code[i]))

{

str+=code[i];

i++;

}

Operators.insert(str);

i--;

}

else if(code[i]!=' ' and code[i]!='\n')specialSymbol.insert(code[i]);

}

}

}

Token::Token(string filename)

{

init();

FILE \*fp=fopen(filename.c\_str(),"r");

code="";

while(!feof(fp))

{

char c;

fscanf(fp,"%c",&c);

code+=c;

}

fclose(fp);

process();

}

vector<string>Token::getKeywords()

{

vector<string>ret;

for(auto x:keywords)ret.push\_back(x);

return ret;

}

vector<string>Token::getIdentifires()

{

vector<string>ret;

for(auto x:Identifiers)ret.push\_back(x);

return ret;

}

vector<string>Token::getConstants()

{

vector<string>ret;

for(auto x:constants)ret.push\_back(x);

return ret;

}

vector<string>Token::getStrings()

{

vector<string>ret;

for(auto x:strings)ret.push\_back(x);

return ret;

}

vector<string>Token::getOperators()

{

vector<string>ret;

for(auto x:Operators)ret.push\_back(x);

return ret;

}

vector<char>Token::getSpecialSymbol()

{

vector<char>ret;

for(auto x:specialSymbol)ret.push\_back(x);

return ret;

}

int main()

{

Token t("input.cpp");

printf("Keywords (%d): ",t.getKeywords().size());

print\_vector(t.getKeywords());

printf("Identifiers (%d): ",t.getIdentifires().size());

print\_vector(t.getIdentifires());

printf("Constants (%d): ",t.getConstants().size());

print\_vector(t.getConstants());

printf("Strings (%d): ",t.getStrings().size());

print\_vector(t.getStrings());

printf("Special symbols (%d): ",t.getSpecialSymbol().size());

print\_vector(t.getSpecialSymbol());

printf("Operators (%d): ",t.getOperators().size());

print\_vector(t.getOperators());

return 0;

}

**Sample Input:**

int main()

{

int A,B;

printf("Enter 1st Number: ");

scanf("%d",&A);

printf("Enter 2nd Number" );

scanf("%d",&B);

int result=A+B;

A++;

printf("result %d\n",result);

return 0;

}

**Output:**

