CODE

```
%{
#include<stdio.h>
#include<string.h>
struct SymbolTable
{
   char symbol[10];
   char type[10];
}SymbolTable[10];

void insert();
int search();
void display();
char data[10];
char type[10];
int count=0;
int i=0;
%}

letter [a-zA-Z]
digit [0-9]
id ({letter}({letter}|{digit})*)
classname (({letter}|{digit})*)
quote \"
space [' ']
datatype (int|char|float|void)
keyword (class|throws|import|try|catch|finally|static|implements|main)
booleanvalues (true|false)
loops (while|for|do)
conditionalstmts (if|else|switch)
string ({quote}({letter}|{digit})+{quote})
array ({id}(\[))
usrfunction ({keyword}(\())
predefinedclass (Exception|String|Integer|Float|Character|ArithmaticException)
userdefinedclass (class{space}{classname})
predefinedfunction (main|println|Scanner|nextInt|next|nextLine)
package (import{space}{id}(\.{id})*(\.\*)?)
param ({string}|{digit}+|({digit}+\.{digit}+)|{id})
params (({param}(\,{param})*)?)
assignment
({id}\=({string}|{digit}+|({digit}+\.{digit}+)|new{space}(({datatype}(\[([0-9]{1,3})?\])*)|((({classname}|{
predefinedclass})\({params}\)))))
operators ((\+\+)|(\-\-)|(\>\=)|(\<\=)|(\=\=)|(\!\=)|(\+\=))
```

```
accessspecifiers (public|private|protected)
%%

{accessspecifiers}          {  printf("\n\n %s is access specifier", yytext);}
{package}                   {   printf("\n\n %s is a package",yytext);}
{operators}                 {  printf("\n\n %s is operator",yytext);}
{assignment}                  {  printf("\n\n %s is assignment",yytext);}
{predefinedclass}             {   printf("\n\n %s is a predefined class",yytext);}
{booleanvalues}                            {printf("\n\n %s is a boolean value",yytext);}
{conditionalstmts}          {printf("\n\n %s is a conditional stmt",yytext);}
{loops}                   {  printf("\n\n %s is a loop",yytext);}
{keyword}                   {  printf("\n\n %s is a keyword", yytext);}
{predefinedfunction}        {printf("\n\n %s is predefined function",yytext);}
{usrfunction}  {

            printf("\n\n %s is a user defined function",yytext);
            yytext[strlen(yytext)-1]='\0';
            strcpy(data,yytext);
            strcpy(type,"id");
            return 5;
         }

{array}     {
            printf("\n\n %s is an array",yytext);
            yytext[strlen(yytext)-1]='\0';
            strcpy(data,yytext);
            strcpy(type,"id");
            return 5;

         }
{string}                    { printf("\n\n %s is a string",yytext);}
{datatype}                    {printf("\n\n %s is a datatype",yytext);}
{id}                        {      printf("\n\n %s is an id",yytext);     }
%%

int main()
{
   int i;
   yyin=fopen("sample.java","r");
   while(i=yylex())
   {
      if(i==5)
      {
            insert();
            printf("\t%s",data);
        }
    }
    display();
    yywrap();
}

int yywrap()
{
return 1;
}
void insert()
{
    if(search(data)==0)
    {
        strcpy(SymbolTable[count].symbol,data);
        strcpy(SymbolTable[count].type,type);
        ++count;
    }
}
int search()
{
    int i;
    for(i=0;i<count;i++)
    {
        if(strcmp(SymbolTable[i].symbol,data)==0)
        {
            return 1;
        }
    }
    return 0;
}
void display()
{
    for(int i=0;i<count;i++)
    {
        printf("\n%s\t%s",SymbolTable[i].symbol,SymbolTable[i].type);
    }
}
```

OUTPUT

```
➜  Lab 5 git:(master) ✗ flex scanner.l
➜  Lab 5 git:(master) ✗ gcc lex.yy.c
➜  Lab 5 git:(master) ✗ ./a.out
a=1


 a=1 is assignment
true


 true is a boolean value
false


 false is a boolean value
==


 == is operator
for


 for is a loop
int


 int is a datatype
a[]


 a[ is an array a]
public static void main(String args[]){System.out.println("Hello");}


 public is access specifier

 static is a keyword

 void is a datatype
```

main(is a user defined function   main

String is a predefined class

args[is an array   args]){

System is an id.

out is an id.

println is predefined function(

"Hello" is a string);}