```
%option noyywrap
%{
#include<stdio.h>
#include<string.h>
struct SymbolTable
{
char symbol[10];
char type[10];
}SymbolTable[10];
int count = 0;
char data[10];
char type[10];
void insert();
void display();
%}
letter[a-zA-Z]
digit[0-9]
num({digit}{digit}*)
KEYWORDS "class"|"static"
datatype(int|char|float|void)
CONDITIONAL "if"|"else"|"else if"|"switch"|"case"
SC ";"
array({id}(\[))
id({letter}({letter}|{digit})*)
ARITH_OP "+"|"-"|"/"|"%"|"*";
LOGICAL_OP "&&"|"||"|"!"|"!="
REL_OP "<"|">"|"<="|">="|"=="
UNARY "++"|"--"
%%

{datatype} {printf("%s is an datatype\n",yytext);}
{CONDITIONAL} { printf("%s\t==> CONDITIONAL\n",yytext);}
{num} {printf("%s is a number\n",yytext);}
{array} {printf("%s is an array\n",yytext);insert(yytext,"array");}
{KEYWORDS} {printf("%s\t==> KEYWORDS\n",yytext);}
{id} {printf("%s is an identifier\n",yytext);insert(yytext,"id");}
{UNARY} {printf("%s\t==> UNARY OP\n",yytext);}
{ARITH_OP} {printf("%s\t==> ARITHMETIC OPERATOR\n",yytext);}
{LOGICAL_OP} {printf("%s\t==> LOGICAL OP\n",yytext);}
{SC} {printf("%s\t==> DELIMITER\n",yytext);}
"=" {printf("%s\t==> ASSIGNMENT OP\n",yytext);}
"{" {printf("%s\t==> BLOCK BEGIN\n",yytext);}
"}" {printf("%s\t==> BLOCK END\n",yytext);}
"(" {printf("%s\t==> PARANTHESIS BEGIN\n",yytext);}
")" {printf("%s\t==> PARENTHESIS END\n",yytext);}
%%
int main()
{
yylex();
display();
```

```
return 0;
}
void insert(char data[10],char type[10])
{strcpy(SymbolTable[count].symbol,data);
strcpy(SymbolTable[count].type,type);
++count;
}
void display()
{
int i;
printf("****Symbol Table***");
for(int i=0;i<count;i++)
{
printf("\n%s\t%s",SymbolTable[i].symbol,SymbolTable[i].type);
}
}
```