



School of Computer Science and Engineering

Fall Semester 2025-26

ASSESSMENT 2

SLOT: L55+L56

Moksh Pun
23BCE0349

Program 1 – Identify Tokens (keyword, datatype, identifier, operators, constants, special symbols)

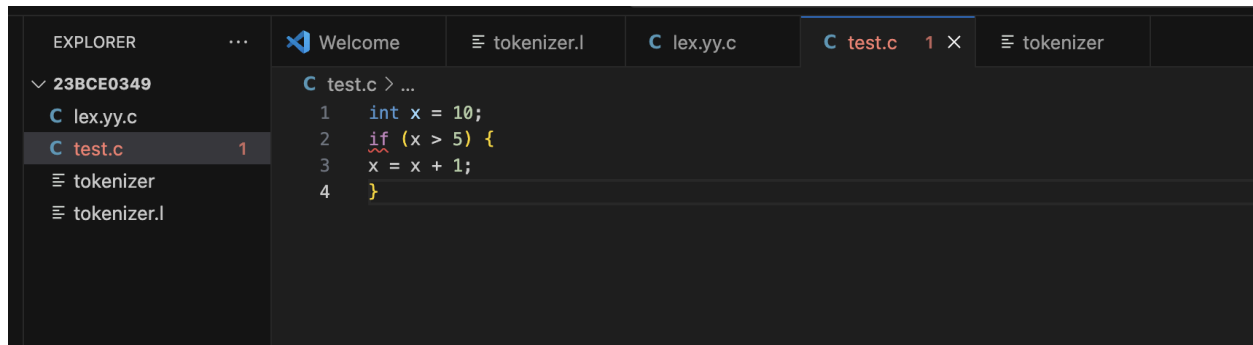
```
%{
#include <stdio.h>
#include <string.h>
int keyword_count = 0, datatype_count = 0, id_count = 0;
int operator_count = 0, constant_count = 0, special_count = 0;
}%
%option noyywrap
DIGIT [0-9]+
ID [a-zA-Z_][a-zA-Z0-9_]*
KEYWORD (if|else|for|while|return|break|continue|switch|case|default|do)
DATATYPE (int|float|char|double|long|short|void)
OPERATOR (\+|\-|\*|\/|=|==|!=|>|<|>=|<=|\+|\-|\-|\-)
SPECIAL (\(|\)|\{|\}|\[|\]|;|,|)
%%
{KEYWORD} { printf("Keyword: %s\n", yytext); keyword_count++; }
{DATATYPE} { printf("Datatype: %s\n", yytext); datatype_count++; }
{ID} { printf("Identifier: %s\n", yytext); id_count++; }
{DIGIT} { printf("Constant: %s\n", yytext); constant_count++; }
{OPERATOR} { printf("Operator: %s\n", yytext); operator_count++; }
{SPECIAL} { printf("Special Symbol: %s\n", yytext); special_count++; }
[\t\n] { /* ignore whitespace */ }
. { /* ignore others */ }
%%
int main(int argc, char *argv[]) {
```

```

if (argc < 2) {
printf("Usage: %s <filename>\n", argv[0]);
return 1;
}
FILE *fp = fopen(argv[1], "r");
if (!fp) {
printf("Could not open file.\n");
return 1;
}
yyin = fp;
yylex();
fclose(fp);
printf("\n--- Summary ---\n");
printf("Keywords: %d\nDatatypes: %d\nIdentifiers: %d\n", keyword_count,
datatype_count, id_count);
printf("Operators: %d\nConstants: %d\nSpecial Symbols: %d\n",
operator_count, constant_count, special_count);return 0;
}

```

FILE INPUT-



The screenshot shows the Visual Studio Code interface. On the left, the 'EXPLORER' sidebar displays a file tree with '23BCE0349' as the root. Under it, there are files 'lex.yy.c', 'test.c' (which is selected and has a red '1' next to it), 'tokenizer', and 'tokenizer.l'. The main editor area shows the content of 'test.c' with the following code:

```

1  int x = 10;
2  if (x > 5) {
3      x = x + 1;
4  }

```

Output:

```

Datatype: int
Identifier: x
Operator: =
Constant: 10
Special Symbol: ;
Keyword: if
Special Symbol: (
Identifier: x
Operator: >
Constant: 5
Special Symbol: )
Special Symbol: {
Identifier: x
Operator: =
Identifier: x
Operator: +
Constant: 1
Special Symbol: ;
Special Symbol: }

```

```

--- Summary ---
Keywords: 1
Datatypes: 1
Identifiers: 4
Operators: 4
Constants: 3
Special Symbols: 6

```

Name: Moksh Punn Register No: 23BCE0349

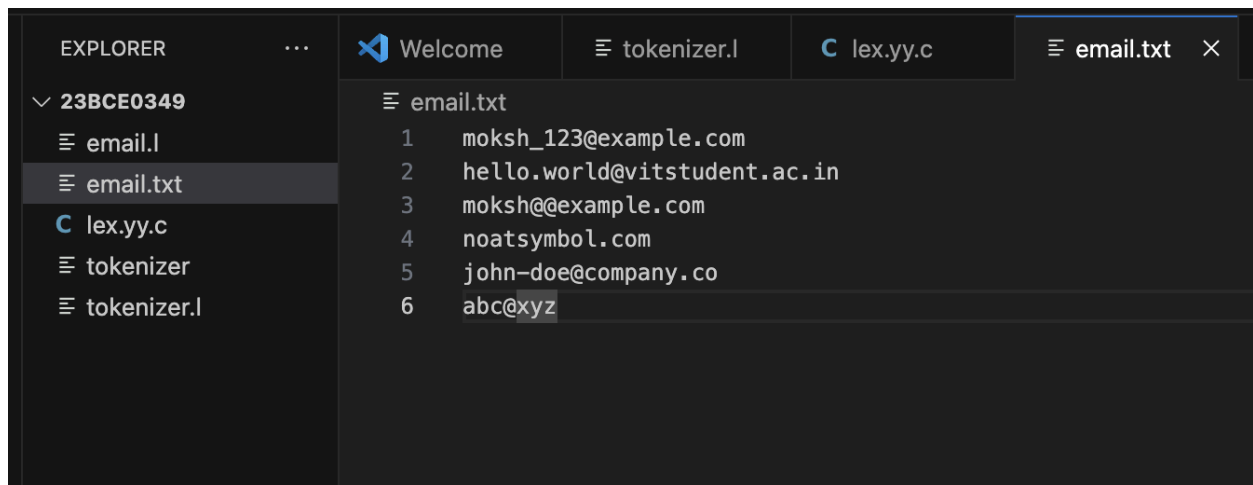
Program 2 – Validate Email Address

```

%{
#include <stdio.h>
#include <string.h>
}%
%option noyywrap
EMAIL [a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}
%%
{EMAIL} .* { printf("Valid Email ID: %s\n", yytext); }
{ printf("Invalid Email ID: %s\n", yytext); }%%
int main(int argc, char *argv[]) {
if (argc < 2) {
printf("Usage: %s <filename>\n", argv[0]);
return 1;
}
FILE *fp = fopen(argv[1], "r");
if (!fp) {
printf("Could not open file.\n");
return 1;
}
yyin = fp;
yylex();
fclose(fp);
return 0;
}

```

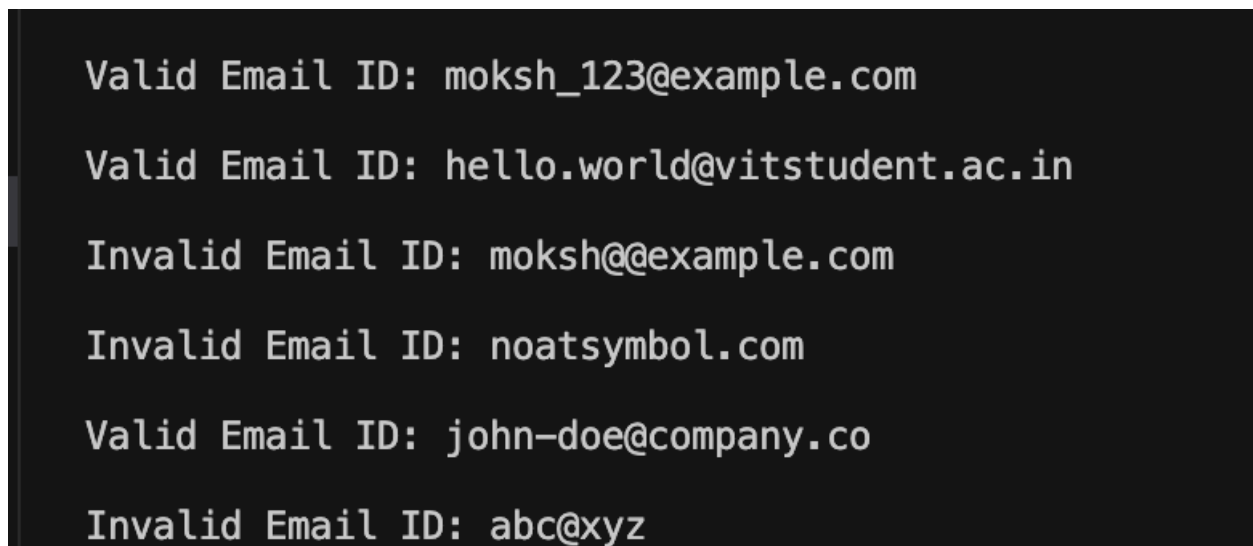
INPUT-



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a folder named '23BCE0349' containing files 'email.l', 'email.txt', 'lex.yy.c', 'tokenizer', and 'tokenizer.l'. The file 'email.txt' is selected. The code editor shows the contents of 'email.txt' with line numbers 1 through 6. The email addresses are: 1. moksh_123@example.com, 2. hello.world@vitstudent.ac.in, 3. moksh@@example.com, 4. noatsymbol.com, 5. john-doe@company.co, 6. abc@xyz.

```
1  moksh_123@example.com
2  hello.world@vitstudent.ac.in
3  moksh@@example.com
4  noatsymbol.com
5  john-doe@company.co
6  abc@xyz
```

OUTPUT-



The screenshot shows a terminal window with the following output:

```
Valid Email ID: moksh_123@example.com
Valid Email ID: hello.world@vitstudent.ac.in
Invalid Email ID: moksh@@example.com
Invalid Email ID: noatsymbol.com
Valid Email ID: john-doe@company.co
Invalid Email ID: abc@xyz
```

Program 3 – Count Characters, Words, Lines, and Whitespaces from a File

```
%{
#include <stdio.h>
int char_count = 0, word_count = 0, line_count = 0, space_count = 0;
%}
%option noyywrap
WORD [^ \t\n]+
SPACE [ \t]
NEWLINE \n
%%
{WORD} { word_count++; char_count += yyleng; }
{SPACE} { space_count++; char_count++; }
{NEWLINE} { line_count++; char_count++; }
. { char_count++; }
%%
int main(int argc, char *argv[]) {
```

```

if (argc < 2) {
printf("Usage: %s <filename>\n", argv[0]);
return 1;
}
FILE *fp = fopen(argv[1], "r");
if (!fp) {printf("Could not open file.\n");
return 1;
}
yyin = fp;
yylex();
fclose(fp);
printf("\n--- File Statistics ---\n");
printf("Characters: %d\n", char_count);
printf("Words: %d\n", word_count);
printf("Lines: %d\n", line_count);
printf("Whitespaces: %d\n", space_count);
return 0;
}

```

Input-

```

23BCE0349
└─ count_stats
└─ count_stats.l
└─ email.l
└─ email.txt
└─ lex.yy.c
└─ tokenizer
└─ tokenizer.l
└─ yourfile.txt

```

```

yourfile.txt
1  Hello world
2  This is a test file.
3  It has three lines.
4

```

OUTPUT-

```
echo "Name: Moksh Punn Register No: 23BCE0349"

count_stats.l:22: warning, rule cannot be matched

--- File Statistics ---
Characters: 53
Words: 11
Lines: 3
Whitespaces: 8
Name: Moksh Punn Register No: 23BCE0349
```