



Ain Shams University  
Faculty of Engineering

# User Manual

---

Name	Mahmoud Hamdy Abd El-Gawwad
Section	3
Name	Doaa Kamel Abdelmoneim Ali
Section	2

Cairo, 2018

## Introduction

- Project is implemented in C++ under Windows OS using Visual Studio IDE

## How to run?

- Make sure that our test file named Tiny\_program.txt exists in the same directory as the .exe file.
- Program starts with operating on the TINY code previously provided in the lecture slides.
- If you want to use another test for a tiny program, edit the existing provided test file.

# Snapshots

## Output:

C:\WINDOWS\system32\cmd.exe

```
Code is scanned Successfully and output file is generated
Press any key to continue . . .
```

output.txt - Notepad

File Edit Format View Help

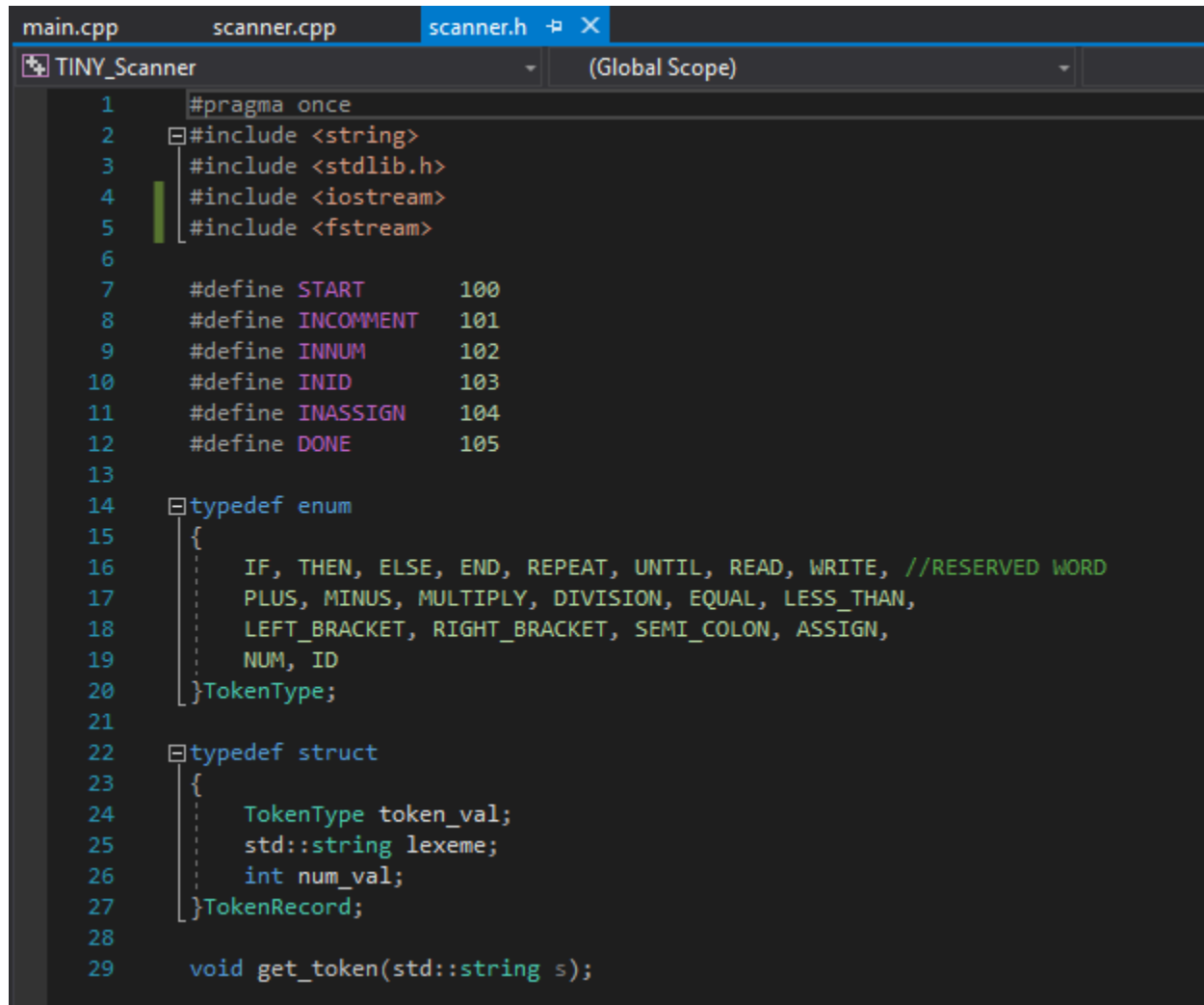
```
read, reserved word
x, Identifier
;, Symbol
if, reserved word
0, Number
<, Symbol
x, Identifier
then, reserved word
fact, Identifier
:=, Symbol
1, Number
;, Symbol
repeat, reserved word
fact, Identifier
:=, Symbol
fact, Identifier
*, Symbol
x, Identifier
;, Symbol
x, Identifier
:=, Symbol
x, Identifier
-, Symbol
1, Number
until, reserved word
x, Identifier
=, Symbol
0, Number
;, Symbol
write, reserved word
fact, Identifier
end, reserved word
|
```

Code:

main.cpp

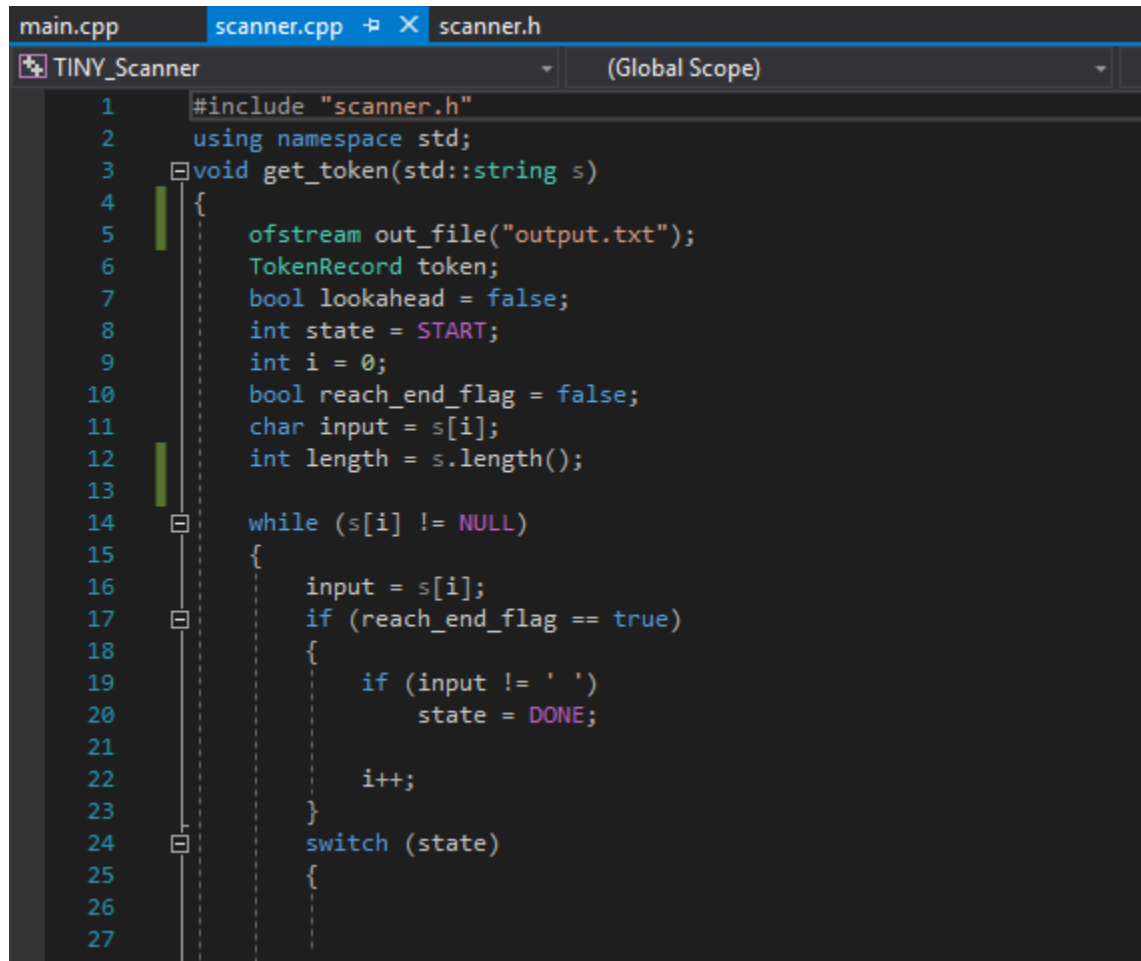
```
1  #include <iostream>
2  #include<string.h>
3  #include <fstream>
4  #include "scanner.h"
5  using namespace std;
6  int main(void)
7  {
8      string x;
9      string a;
10     ifstream my_prog;
11     my_prog.open("Tiny_program.txt");
12     if (my_prog.is_open())
13     {
14         while (!my_prog.eof())
15         {
16             getline(my_prog, a);
17             x += a;
18         }
19         if (x[x.length() - 1] != '\\0')
20             x += '\\0';
21     }
22     get_token(x);
23     system("pause");
24 }
```

scanner.h



```
1  #pragma once
2  #include <string>
3  #include <stdlib.h>
4  #include <iostream>
5  #include <fstream>
6
7  #define START      100
8  #define INCOMMENT  101
9  #define INNUM      102
10 #define INID       103
11 #define INASSIGN   104
12 #define DONE      105
13
14 typedef enum
15 {
16     IF, THEN, ELSE, END, REPEAT, UNTIL, READ, WRITE, //RESERVED WORD
17     PLUS, MINUS, MULTIPLY, DIVISION, EQUAL, LESS_THAN,
18     LEFT_BRACKET, RIGHT_BRACKET, SEMI_COLON, ASSIGN,
19     NUM, ID
20 }TokenType;
21
22 typedef struct
23 {
24     TokenType token_val;
25     std::string lexeme;
26     int num_val;
27 }TokenRecord;
28
29 void get_token(std::string s);
```

scanner.cpp



The screenshot shows a code editor with three tabs: main.cpp, scanner.cpp (active), and scanner.h. The active tab displays the implementation of the get\_token function in scanner.cpp. The code is as follows:

```
1  #include "scanner.h"
2  using namespace std;
3  void get_token(std::string s)
4  {
5      ofstream out_file("output.txt");
6      TokenRecord token;
7      bool lookahead = false;
8      int state = START;
9      int i = 0;
10     bool reach_end_flag = false;
11     char input = s[i];
12     int length = s.length();
13
14     while (s[i] != NULL)
15     {
16         input = s[i];
17         if (reach_end_flag == true)
18         {
19             if (input != ' ')
20                 state = DONE;
21
22             i++;
23         }
24         switch (state)
25         {
26
27             START
```

```
main.cpp scanner.cpp X scanner.h
TINY_Scanner (Global Scope)
27
28     case START:
29         if (input == ' ')
30             state = START;
31         else if (input == '{')
32             state = INCOMMENT;
33         else if (int(input) >= int('0') && int(input) <= int('9'))
34         {
35             token.token_val = NUM;
36             token.lexeme = input;
37             token.num_val = int(input);
38             state = INNUM;
39         }
40         else if (int(input) >= int('A') && int(input) <= int('z'))
41         {
42             token.token_val = ID;
43             token.lexeme = input;
44             state = INID;
45         }
46         else if (input == ':')
47         {
48             token.token_val = ASSIGN;
49             token.lexeme = input;
50             state = INASSIGN;
51         }
```

```
main.cpp scanner.cpp X scanner.h
TINY_Scanner (Global Scope)
52 else
53 {
54     state = DONE;
55     token.lexeme = input;
56
57     if (input == '+')
58         token.token_val = PLUS;
59
60     else if (input == '-')
61         token.token_val = MINUS;
62
63     else if (input == '*')
64         token.token_val = MULTIPLY;
65
66     else if (input == '/')
67         token.token_val = DIVISION;
68
69     else if (input == '=')
70         token.token_val = EQUAL;
71
72     else if (input == '<')
73         token.token_val = LESS_THAN;
74
75     else if (input == '(')
76         token.token_val = LEFT_BRACKET;
77
78     else if (input == ')')
79         token.token_val = RIGHT_BRACKET;
80
```

```
main.cpp scanner.cpp X scanner.h
TINY_Scanner (Global Scope)
80
81     else if (input == ';')
82         token.token_val = SEMI_COLON;
83     }
84     break;
85
86     case INCOMMENT:
87         if (input == '}')
88             state = START;
89
90     else
91         state = INCOMMENT;
92     break;
93
```



```
main.cpp scanner.cpp X scanner.h
TINY_Scanner (Global Scope)
93
94 case INNUM:
95     if (int(input) >= int('0') && int(input) <= int('9'))
96     {
97         token.lexeme += input;
98         token.num_val = stoi(token.lexeme);
99     }
100     else
101     {
102         lookahead = true;
103         state = DONE;
104     }
105     break;
106
107 case INID:
108     if (int(input) >= int('A') && int(input) <= int('z'))
109     {
110         token.lexeme += input;
111         state = INID;
112     }
113     else
114     {
115         lookahead = true;
116         state = DONE;
117     }
118
119     break;
```

```
main.cpp scanner.cpp X scanner.h
TINY_Scanner (Global Scope)
120
121 case INASSIGN:
122     if (input == '=')
123     {
124         token.lexeme += input;
125         state = DONE;
126     }
127     else
128     {
129         lookahead = true;
130         state = DONE;
131     }
132     break;
133
```

```
main.cpp scanner.cpp X scanner.h
TINY_Scanner (Global Scope)
134 case DONE:
135     state = START;
136     out_file << token.lexeme << ", ";
137     switch (token.token_val)
138     {
139     case ID:
140         if (token.lexeme == "if" || token.lexeme == "else" || token.lexeme == "then" ||
141             token.lexeme == "end" || token.lexeme == "repeat" || token.lexeme == "until" ||
142             token.lexeme == "read" || token.lexeme == "write")
143         {
144             out_file << "reserved word";
145         }
146         else
147         {
148             out_file << "Identifier";
149         }
150         break;
151
152     case NUM:
153         out_file << "Number";
154         break;
155
156     default:
157         out_file << "Symbol";
158         break;
159
160     }
161
162     out_file << endl;
```

```
main.cpp scanner.cpp X scanner.h
TINY_Scanner (Global Scope)
163
164     if (lookahead == true)
165     {
166         i--;
167         lookahead = false;
168     }
169
170     break;
171 }
172
173 if (i == (length - 1))
174 {
175     reach_end_flag = true;
176 }
177 if (state != DONE && i != (length - 1) && i != length)
178     i++;
179
180 }
181 cout << "Code is scanned Successfully and output file is generated" << endl;
182 }
```