Aayush Shah
19BCE245
17 August 2022

Compiler Construction

Practical 1

To implement lexical analyse to recognize all distinct token classes

- **Code :**

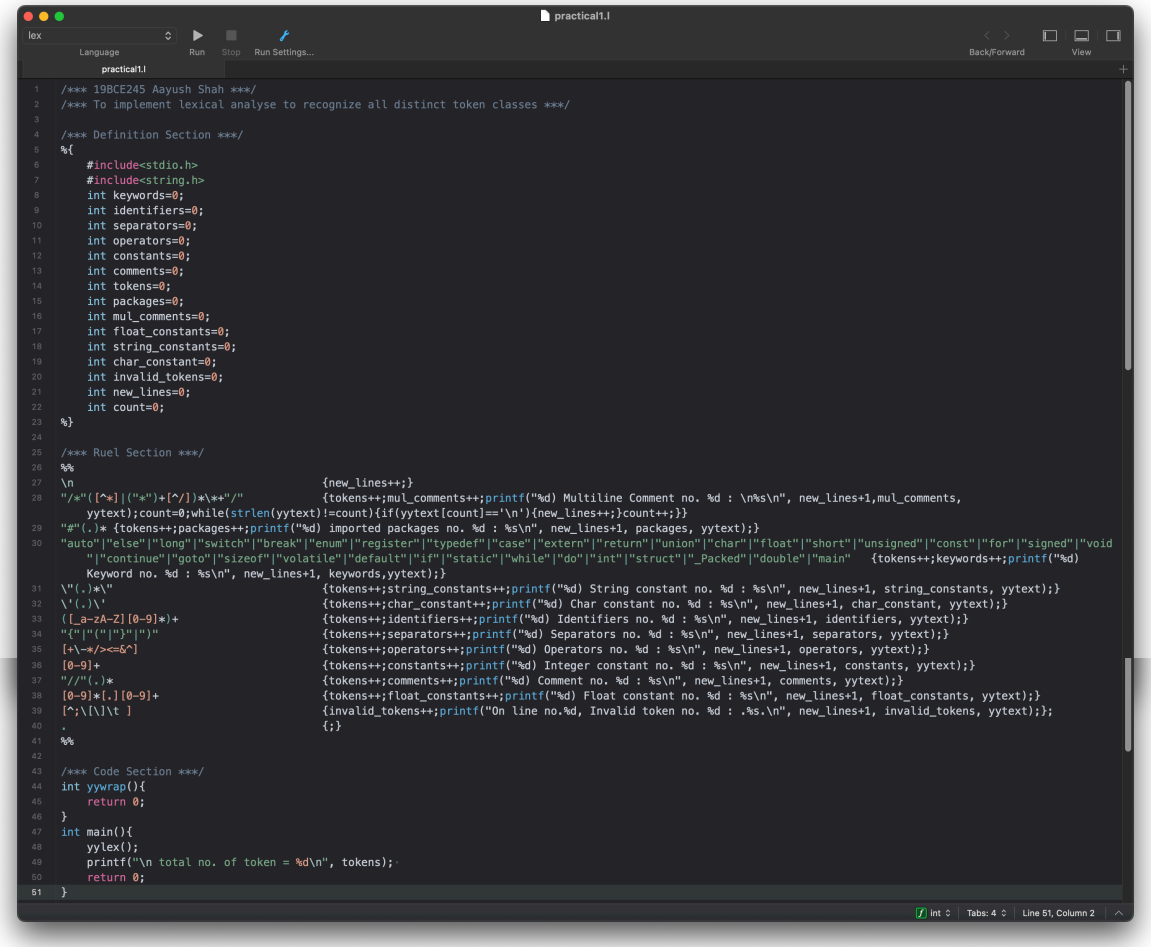
practical1.l

```
/*** 19BCE245 Aayush Shah ***/
/*** To implement lexical analyse to recognize all
distinct token classes ***/

/*** Definition Section ***/
%{
    #include<stdio.h>
    #include<string.h>
    int keywords=0;
    int identifiers=0;
    int separators=0;
    int operators=0;
    int constants=0;
    int comments=0;
    int tokens=0;
    int packages=0;
    int mul_comments=0;
    int float_constants=0;
    int string_constants=0;
    int char_constant=0;
    int invalid_tokens=0;
    int new_lines=0;
    int count=0;
%}
```

```
/*** Ruel Section ***/
%%
\n                              {new_lines++;}
"/*"([^*]|("*")+[^/])*\*+"/"     {tokens++;mul_comments+
+;printf("%d) Multiline Comment no. %d : \n%s\n",
new_lines+1,mul_comments,
yytext);count=0;while(strlen(yytext)!=count)
{if(yytext[count]=='\n'){new_lines++;}count++;}}
"#"(.)*   {tokens++;packages++;printf("%d) imported
packages no. %d : %s\n", new_lines+1, packages, yytext);}
"auto"|"else"|"long"|"switch"|"break"|"enum"|"register"|"
typedef"|"case"|"extern"|"return"|"union"|"char"|"float"|
"short"|"unsigned"|"const"|"for"|"signed"|"void"|"continu
e"|"goto"|"sizeof"|"volatile"|"default"|"if"|"static"|"wh
ile"|"do"|"int"|"struct"|"_Packed"|"double"|"main"
{tokens++;keywords++;printf("%d) Keyword no. %d : %s\n",
new_lines+1, keywords,yytext);}
\"(.)*\"            {tokens++;string_constants+
+;printf("%d) String constant no. %d : %s\n",
new_lines+1, string_constants, yytext);}
\'(.)\'             {tokens++;char_constant++;printf("%d)
Char constant no. %d : %s\n", new_lines+1, char_constant,
yytext);}
([_a-zA-Z][0-9]*)+          {tokens++;identifiers+
+;printf("%d) Identifiers no. %d : %s\n", new_lines+1,
identifiers, yytext);}
"{"|"("|"}"|")"                  {tokens++;separators+
+;printf("%d) Separators no. %d : %s\n", new_lines+1,
separators, yytext);}
[+\-*/><=&^]                     {tokens++;operators+
+;printf("%d) Operators no. %d : %s\n", new_lines+1,
operators, yytext);}
[0-9]+                           {tokens++;constants+
+;printf("%d) Integer constant no. %d : %s\n",
new_lines+1, constants, yytext);}
"//"(.)*                        {tokens++;comments+
+;printf("%d) Comment no. %d : %s\n", new_lines+1,
comments, yytext);}
[0-9]*[.][0-9]+                  {tokens+
+;float_constants++;printf("%d) Float constant no. %d :
%s\n", new_lines+1, float_constants, yytext);}
```

```
[^;\[\]\t ] {invalid_tokens++;printf("On line no.%d, Invalid token no. %d : .%s.\n", new_lines+1, invalid_tokens, yytext);};
. {;}
%%

/*** Code Section ***/
int yywrap(){
     return 0;
}
int main(){
     yylex();
     printf("\n total no. of token = %d\n", tokens);
     return 0;
}
```

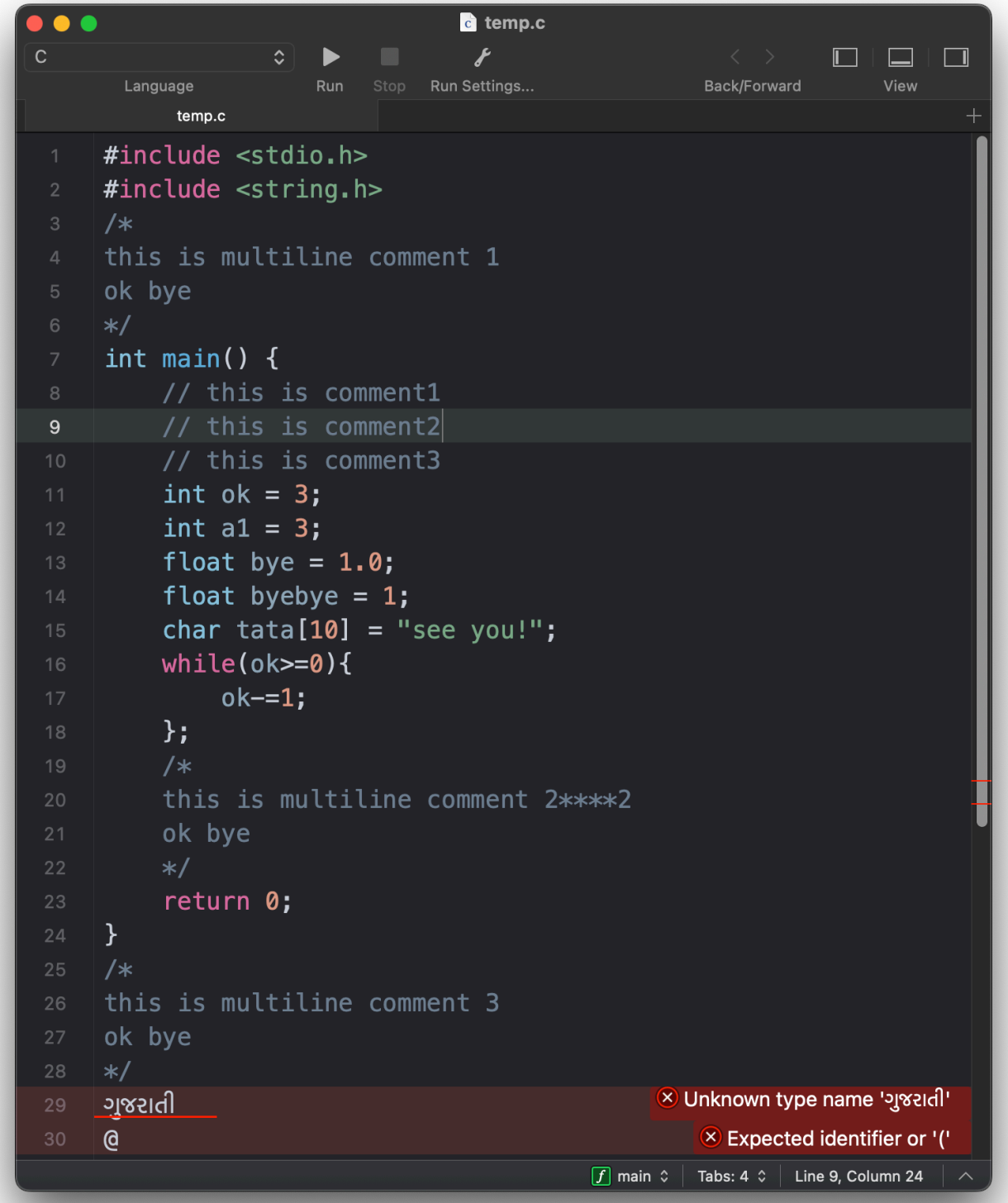
• **logic file code snapshot :**



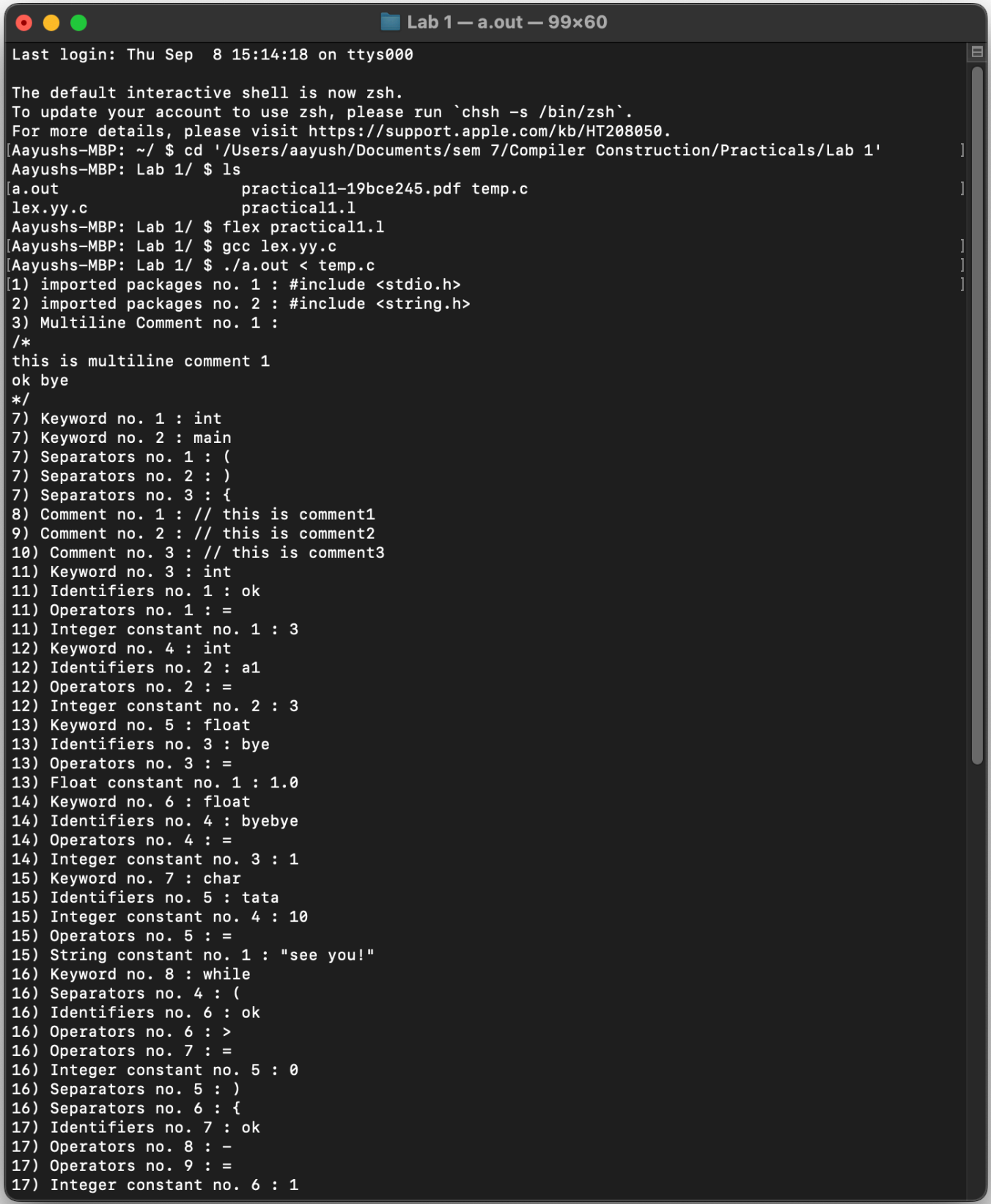
temp.c

```c
#include <stdio.h>
#include <string.h>
/*
this is multiline comment 1
ok bye
*/
int main() {
     // this is comment1
     // this is comment2
     // this is comment3
     int ok = 3;
     int a1 = 3;
     float bye = 1.0;
     float byebye = 1;
     char tata[10] = "see you!";
     while(ok>=0){
          ok-=1;
     };
     /*
     this is multiline comment 2****2
     ok bye
     */
     return 0;
}
/*
this is multiline comment 3
ok bye
*/
ગુજરાતી
@
```

• **Input file code snapshot :**



- **Output :**


Lab 1 — a.out — 99×60
Last login: Thu Sep 8 15:14:18 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Aayushs-MBP: ~/ $ cd '/Users/aayush/Documents/sem 7/Compiler Construction/Practicals/Lab 1'
Aayushs-MBP: Lab 1/ $ ls
a.out practical1-19bce245.pdf temp.c
lex.yy.c practical1.l
Aayushs-MBP: Lab 1/ $ flex practical1.l
Aayushs-MBP: Lab 1/ $ gcc lex.yy.c
Aayushs-MBP: Lab 1/ $ ./a.out < temp.c
1) imported packages no. 1 : #include <stdio.h>
2) imported packages no. 2 : #include <string.h>
3) Multiline Comment no. 1 :
/*
this is multiline comment 1
ok bye
*/
7) Keyword no. 1 : int
7) Keyword no. 2 : main
7) Separators no. 1 : (
7) Separators no. 2 :)
7) Separators no. 3 : {
8) Comment no. 1 : // this is comment1
9) Comment no. 2 : // this is comment2
10) Comment no. 3 : // this is comment3
11) Keyword no. 3 : int
11) Identifiers no. 1 : ok
11) Operators no. 1 : =
11) Integer constant no. 1 : 3
12) Keyword no. 4 : int
12) Identifiers no. 2 : a1
12) Operators no. 2 : =
12) Integer constant no. 2 : 3
13) Keyword no. 5 : float
13) Identifiers no. 3 : bye
13) Operators no. 3 : =
13) Float constant no. 1 : 1.0
14) Keyword no. 6 : float
14) Identifiers no. 4 : byebye
14) Operators no. 4 : =
14) Integer constant no. 3 : 1
15) Keyword no. 7 : char
15) Identifiers no. 5 : tata
15) Integer constant no. 4 : 10
15) Operators no. 5 : =
15) String constant no. 1 : "see you!"
16) Keyword no. 8 : while
16) Separators no. 4 : (
16) Identifiers no. 6 : ok
16) Operators no. 6 : >
16) Operators no. 7 : =
16) Integer constant no. 5 : 0
16) Separators no. 5 :)
16) Separators no. 6 : {
17) Identifiers no. 7 : ok
17) Operators no. 8 : -
17) Operators no. 9 : =
17) Integer constant no. 6 : 1

• **Full Output :**

```
1) imported packages no. 1 : #include <stdio.h>
2) imported packages no. 2 : #include <string.h>
3) Multiline Comment no. 1 :
/*
this is multiline comment 1
ok bye
*/
7) Keyword no. 1 : int
7) Keyword no. 2 : main
7) Separators no. 1 : (
7) Separators no. 2 : )
7) Separators no. 3 : {
8) Comment no. 1 : // this is comment1
9) Comment no. 2 : // this is comment2
10) Comment no. 3 : // this is comment3
11) Keyword no. 3 : int
11) Identifiers no. 1 : ok
11) Operators no. 1 : =
11) Integer constant no. 1 : 3
12) Keyword no. 4 : int
12) Identifiers no. 2 : a1
12) Operators no. 2 : =
12) Integer constant no. 2 : 3
13) Keyword no. 5 : float
13) Identifiers no. 3 : bye
13) Operators no. 3 : =
13) Float constant no. 1 : 1.0
14) Keyword no. 6 : float
14) Identifiers no. 4 : byebye
14) Operators no. 4 : =
14) Integer constant no. 3 : 1
15) Keyword no. 7 : char
15) Identifiers no. 5 : tata
15) Integer constant no. 4 : 10
15) Operators no. 5 : =
15) String constant no. 1 : "see you!"
16) Keyword no. 8 : while
16) Separators no. 4 : (
16) Identifiers no. 6 : ok
16) Operators no. 6 : >
16) Operators no. 7 : =
16) Integer constant no. 5 : 0
16) Separators no. 5 : )
16) Separators no. 6 : {
17) Identifiers no. 7 : ok
17) Operators no. 8 : -
17) Operators no. 9 : =
17) Integer constant no. 6 : 1
18) Separators no. 7 : }
19) Multiline Comment no. 2 :
```

```
/*
    this is multiline comment 2****2
    ok bye
    */
23) Keyword no. 9 : return
23) Integer constant no. 7 : 0
24) Separators no. 8 : }
25) Multiline Comment no. 3 :
/*
this is multiline comment 3
ok bye
*/
On line no.29, Invalid token no. 1 : .?.
On line no.29, Invalid token no. 2 : .?.
On line no.29, Invalid token no. 3 : .?.
On line no.29, Invalid token no. 4 : .?.
On line no.29, Invalid token no. 5 : .?.
On line no.29, Invalid token no. 6 : .?.
On line no.29, Invalid token no. 7 : .?.
On line no.29, Invalid token no. 8 : .?.
On line no.29, Invalid token no. 9 : .?.
On line no.29, Invalid token no. 10 : .?.
On line no.29, Invalid token no. 11 : .?.
On line no.29, Invalid token no. 12 : .?.
On line no.29, Invalid token no. 13 : .?.
On line no.29, Invalid token no. 14 : .?.
On line no.29, Invalid token no. 15 : .?.
On line no.29, Invalid token no. 16 : .?.
On line no.29, Invalid token no. 17 : .?.
On line no.29, Invalid token no. 18 : .?.
On line no.29, Invalid token no. 19 : .?.
On line no.29, Invalid token no. 20 : .?.
On line no.29, Invalid token no. 21 : .?.
On line no.30, Invalid token no. 22 : .@.
```