ASSIGNMENT: 01

Name: Abhijeet Biswas
SRN: 201900400
Roll No: 05
Div: B

Question:

Design a Lexical analyser for the subset of Java Language. Read input from the file. Also create symbol table. Detect any one lexical error. Output in 4 columns Line No, Lexeme, Token and Token Value. Upload single file containing input, output and source code.

Input (Java Program):

```
public class prog {
  public static void main ( String [ ] args ) {
    int num1 = 5 , num2 = 15 , add ;
    add = num1 + num2 ;
    System.out.println ( " Sum is " + add ) ;
  }
}
```

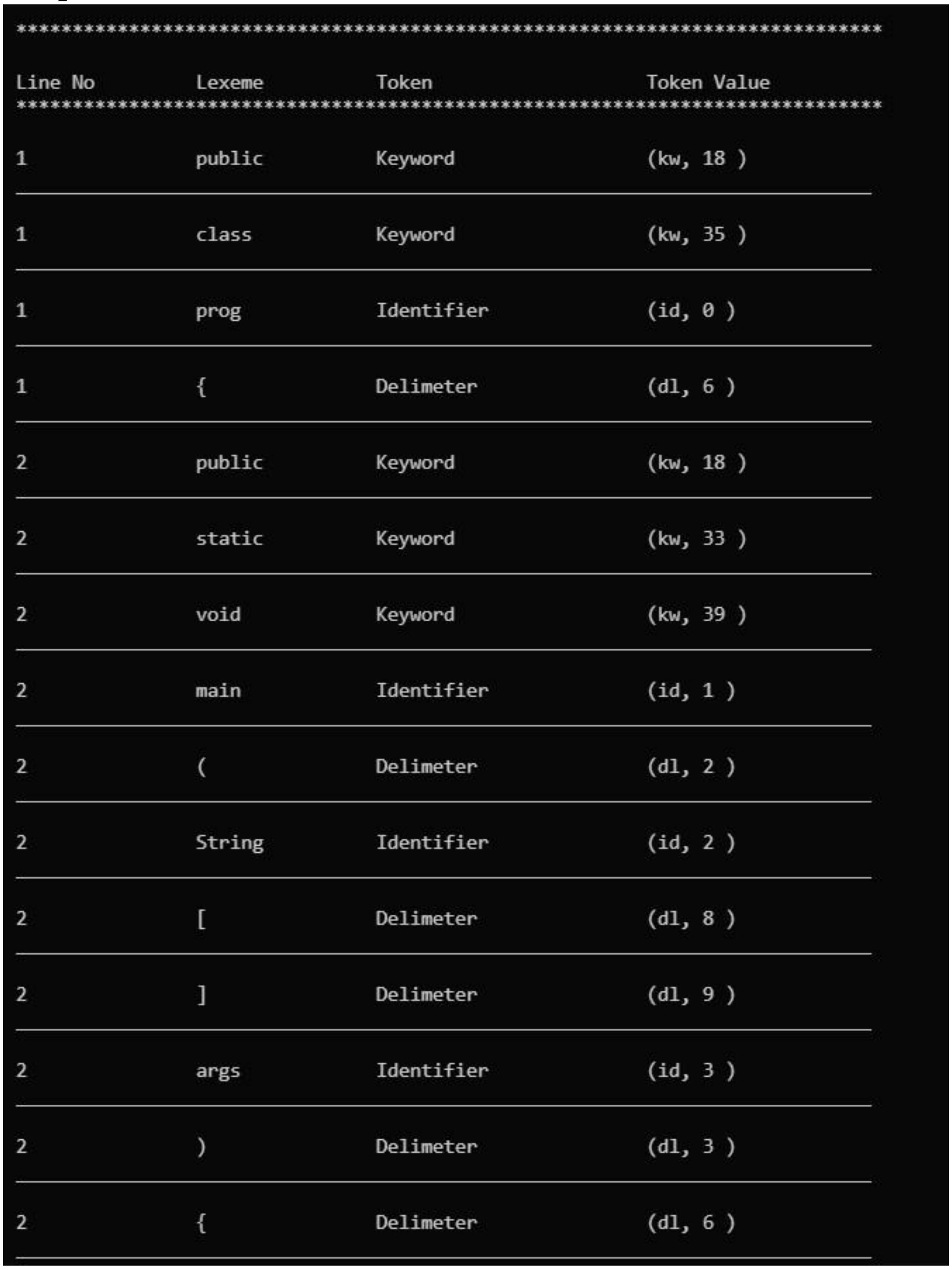
Code:

```
import re
Keyword =
["abstract","do","if","package","synchronized","boolean","double","implements
","private","this","break", "else","import","protected",
"throw","byte","extends","instanceof","public","throws","case","false","int","ret
urn","transient","catch","final","interface","short",

"true","char","finally","long","static","try","class","float","native","strictfp","void
","const","for","new","super","volatile","continue",
        "goto","null","switch","while","default","assert","string"]
Operators = ["+", "-", "*", "/", "%","<", "<=", ">", ">=", "==","!=", "<<", ">>",
">>>", "=", "+=","-=", "*=", "/=", "&", "^","|", "&&", "||",
        "?:", "!", "^=","|=", "<<=", ">>=", ">>>=","++", "--"]
Delimiters = [",", ";", "(", ")", "\\", "/", "{", "}", "[", "]",'"']
seperators=["."]

Symbol = [ ]
l = 0
a= open('prog.java', 'r')
content = a.readlines()
data=[]
r="^([a-zA-Z_$][a-zA-Z\\d_$]*)$"
s=".([^.]+)."
print("\n*************************************************************
*******************\n")
print("Line No\t\tLexeme\t\tToken\t\t\tToken
Value\n*************************************************************
******************\n")
for line in content:
   l += 1
   line = line.strip()
   data = line.split(' ')
```

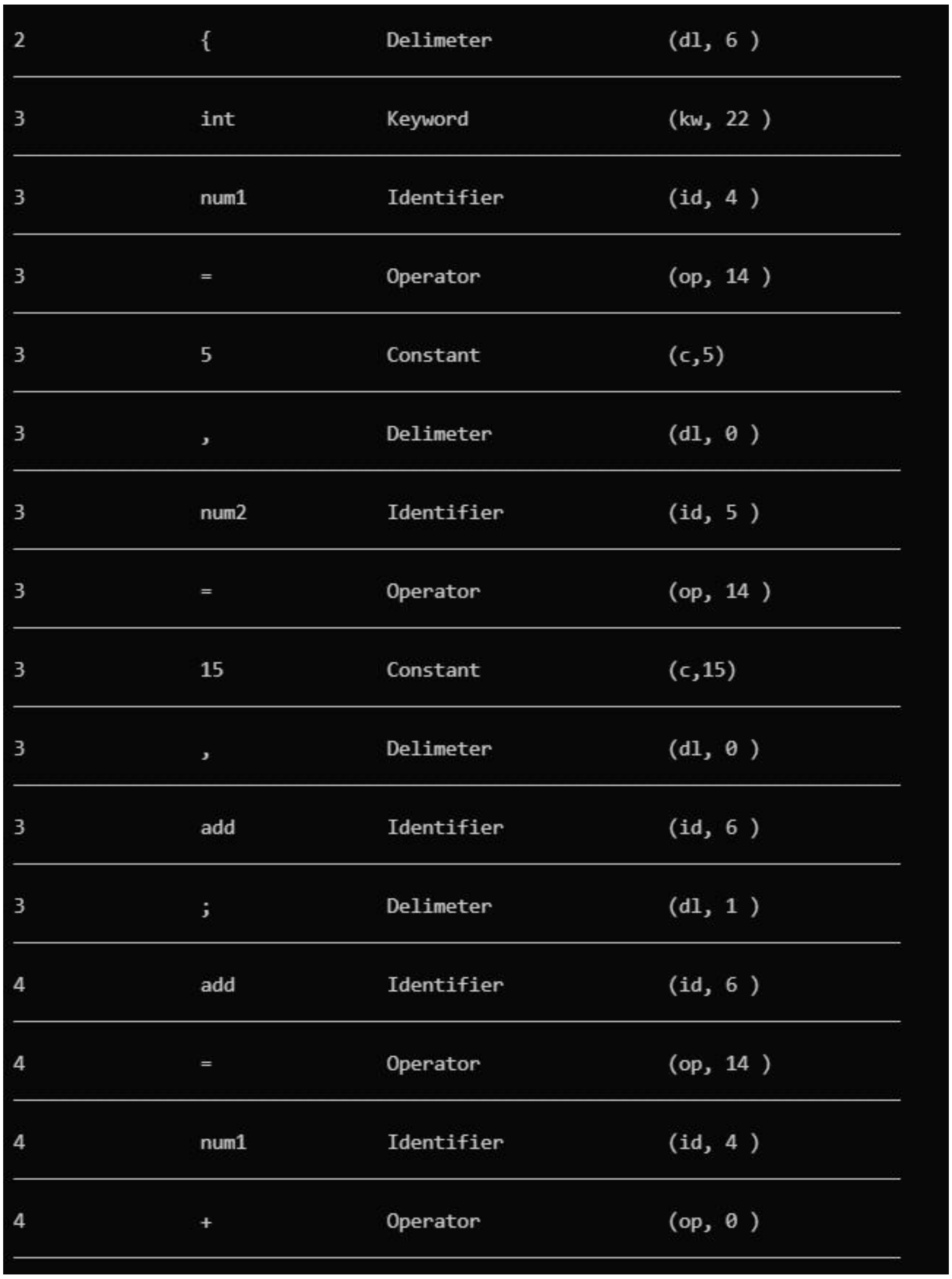
```
  try:
    for i in range(0, 15):
      if data[i] in Delimiters:

           indk=Delimiters.index(data[i])

print(l,"\t\t"+data[i]+"\t\tDelimeter\t\t(dl,",indk,")\n________________________
___________________________________________________________\n")
      elif data[i] in Operators:

           indk=Operators.index(data[i])

print(l,"\t\t"+data[i]+"\t\tOperator\t\t(op,",indk,")\n________________________
___________________________________________________________\n")
      elif data[i].isnumeric():

print(l,"\t\t"+data[i]+"\t\tConstant\t\t(c,"+data[i]+")\n______________________
____________________________________________________________\n")
      elif data[i] in Keyword:
           indk = Keyword.index(data[i])

print(l,"\t\t"+data[i]+"\t\tKeyword\t\t\t(kw,",indk,")\n______________________
____________________________________________________________\n")
      elif (re.search(r,data[i])) :
            if data[i] not in Symbol:

              Symbol.append(data[i])

              indk = Symbol.index(data[i])


print(l,"\t\t"+data[i]+"\t\tIdentifier\t\t(id,",indk,")\n_________________________
__________________________________________________________\n")
            elif data[i] in Symbol:

              indk = Symbol.index(data[i])


print(l,"\t\t"+data[i]+"\t\tIdentifier\t\t(id,",indk,")\n_________________________
__________________________________________________________\n")
        elif (re.search(s,data[i])) :
            new=data[i].split(".")

            for wr in new:

             if wr not in Symbol:

              Symbol.append(wr)

              indk = Symbol.index(wr)
```

```
print(l,"\t\t"+wr+"\t\tidentifier\t\t(id,",indk,")\n_________________________________
_______________________________________________________________\n")

                elif wr in Symbol:


                  indk = Symbol.index(wr)



print(l,"\t\t"+wr+"\t\tidentifier\t\t(id,",indk,")\n_________________________________
_______________________________________________________________\n")




          else: print("error
at\t"+data[i]+"\n________________________________________________________________
________________________________\n")


   except:

       pass

print("\n\n**********************************************************
************\n\n")

print("\t\t\tSYMBOL
TABLE\n\n*******************************************************
***********\n")

print("\t\tSymbol\t\t\tIndex\n\t\t************************************
****\n")

for word in Symbol:

   i = Symbol.index(word);


print("\t\t"+word+"\t\t\t",i,"\n\t\t_______________________________________________
_____\n")
```

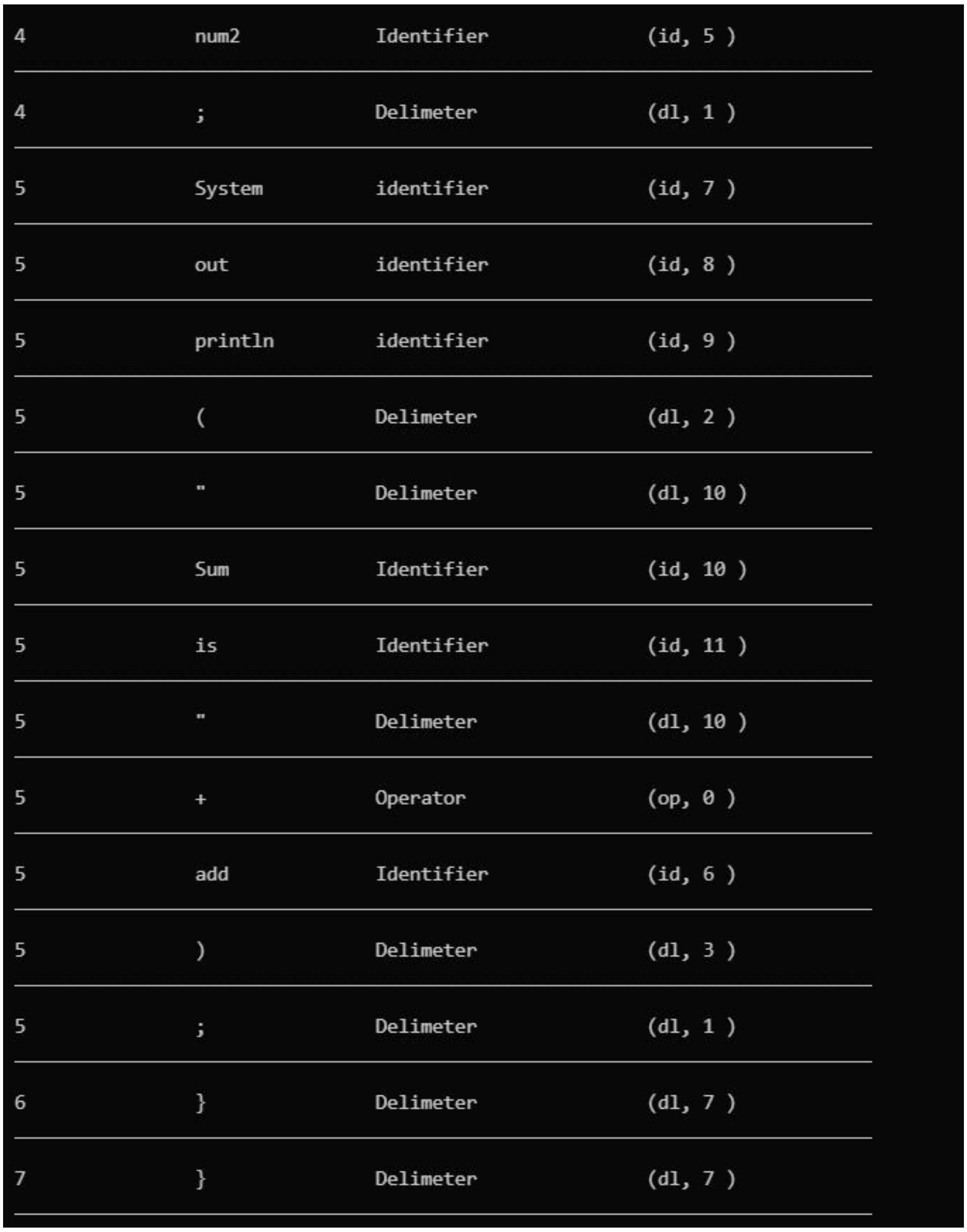
Output:



```
******************************************************************************

Line No         Lexeme          Token                   Token Value
******************************************************************************

1               public          Keyword                 (kw, 18 )
_____________________________________________________________________________

1               class           Keyword                 (kw, 35 )
_____________________________________________________________________________

1               prog            Identifier              (id, 0 )
_____________________________________________________________________________

1               {               Delimeter               (dl, 6 )
_____________________________________________________________________________

2               public          Keyword                 (kw, 18 )
_____________________________________________________________________________

2               static          Keyword                 (kw, 33 )
_____________________________________________________________________________

2               void            Keyword                 (kw, 39 )
_____________________________________________________________________________

2               main            Identifier              (id, 1 )
_____________________________________________________________________________

2               (               Delimeter               (dl, 2 )
_____________________________________________________________________________

2               String          Identifier              (id, 2 )
_____________________________________________________________________________

2               [               Delimeter               (dl, 8 )
_____________________________________________________________________________

2               ]               Delimeter               (dl, 9 )
_____________________________________________________________________________

2               args            Identifier              (id, 3 )
_____________________________________________________________________________

2               )               Delimeter               (dl, 3 )
_____________________________________________________________________________

2               {               Delimeter               (dl, 6 )
_____________________________________________________________________________
```

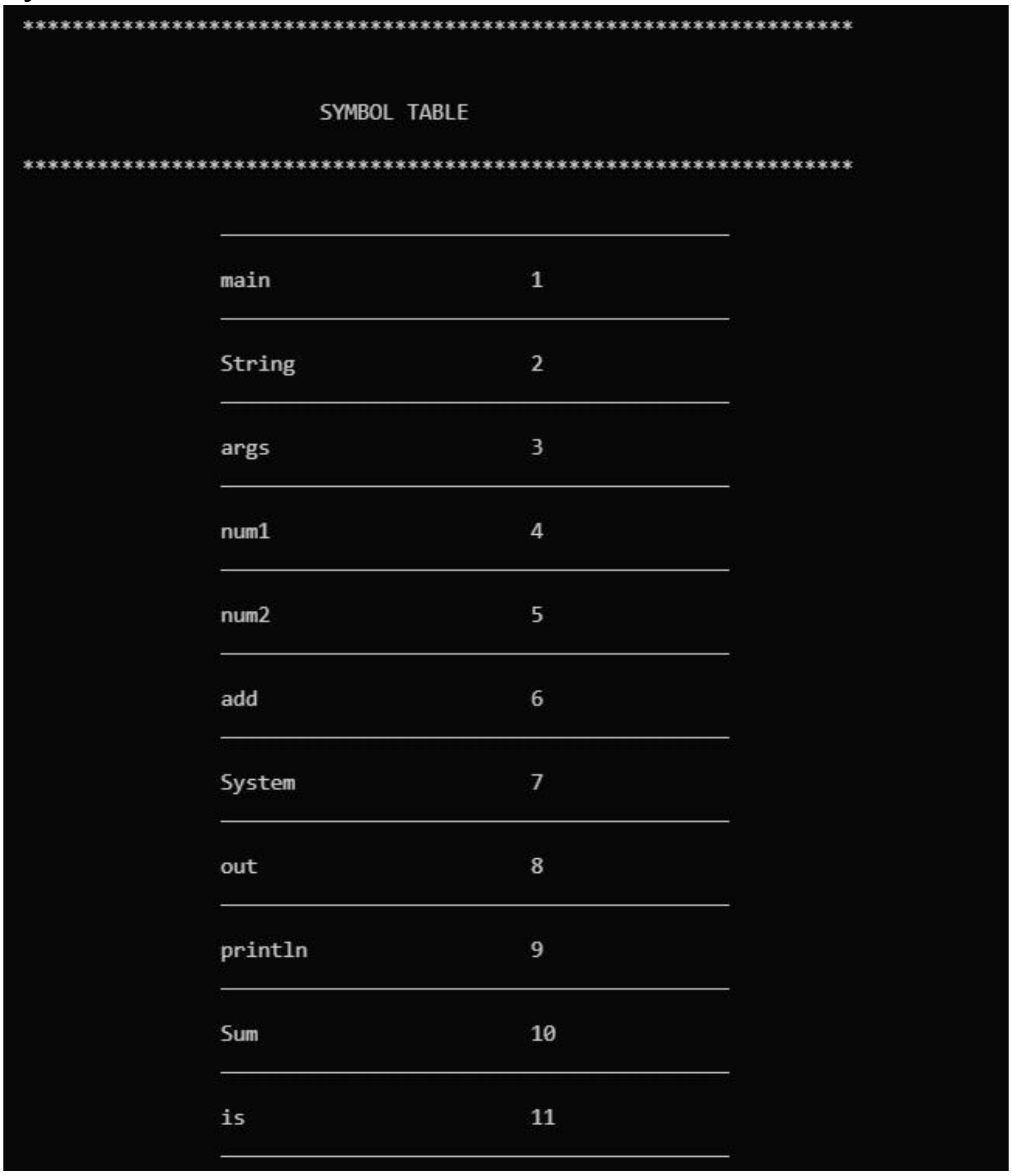


```
2        {        Delimeter     (dl, 6 )
3        int      Keyword       (kw, 22 )
3        num1     Identifier    (id, 4 )
3        =        Operator      (op, 14 )
3        5        Constant      (c,5)
3        ,        Delimeter     (dl, 0 )
3        num2     Identifier    (id, 5 )
3        =        Operator      (op, 14 )
3        15       Constant      (c,15)
3        ,        Delimeter     (dl, 0 )
3        add      Identifier    (id, 6 )
3        ;        Delimeter     (dl, 1 )
4        add      Identifier    (id, 6 )
4        =        Operator      (op, 14 )
4        num1     Identifier    (id, 4 )
4        +        Operator      (op, 0 )
```



```
4              num2           Identifier            (id, 5 )

4              ;              Delimeter             (dl, 1 )

5              System         identifier            (id, 7 )

5              out            identifier            (id, 8 )

5              println        identifier            (id, 9 )

5              (              Delimeter             (dl, 2 )

5              "              Delimeter             (dl, 10 )

5              Sum            Identifier            (id, 10 )

5              is             Identifier            (id, 11 )

5              "              Delimeter             (dl, 10 )

5              +              Operator              (op, 0 )

5              add            Identifier            (id, 6 )

5              )              Delimeter             (dl, 3 )

5              ;              Delimeter             (dl, 1 )

6              }              Delimeter             (dl, 7 )

7              }              Delimeter             (dl, 7 )
```

Symbol Table:



```
*****************************************************************


                      SYMBOL TABLE

*****************************************************************


               _______________________________________

               main                       1
               _______________________________________

               String                     2
               _______________________________________

               args                       3
               _______________________________________

               num1                       4
               _______________________________________

               num2                       5
               _______________________________________

               add                        6
               _______________________________________

               System                     7
               _______________________________________

               out                        8
               _______________________________________

               println                    9
               _______________________________________

               Sum                        10
               _______________________________________

               is                         11
               _______________________________________
```