Lab 1 Lexer

Sorin Macaluso

February 12, 2024

1 Test Cases

These are the test cases that I used for the program these are taken from Lexer without spaces and Gabriel Arnell Hall of Fame project. I have added changes to them to have unrecognized tokens and key words, as well a numbers in strings, pretty much anything that is not a lower case letter or space (since lowercase letters and space's are the only things allowed inside quotes).

I attempted to make it so that the tokens would match. Could not get all of them without ruining the document.

1.1 Test Case 1

```
/*Lexer without spaces*/
/*Long Test Case - Everything Except Boolean Declaration */
{
    /* Int Declaration */
    int a
    int b
    a = 0
    b=0
    /* While Loop */
    while (a != 3) {
        print(a)
        while (b != 3) {
            print(b)
            b = 1 + b
            if (b == 2) {
                /* Print Statement */
                print({"there is no spoon" /* This will do nothing */ )
            }
        }
        b = 0
        a = 1+a
    }
}
$

Output:
Begin_Block [ { ] Found at line 2 : 1
TYPE [ int ] Found at line 4 : 11
ID [ a ] Found at line 4 : 13
TYPE [ int ] Found at line 5 : 11
```

```
ID [ b ] Found at line 5 : 13
ID [ a ] Found at line 6 : 9
AssignmentStatement [ = ] Found at line 6 : 11
DIGIT [ 0 ] Found at line 6 : 13
ID [ b ] Found at line 7 : 9
AssignmentStatement [ = ] Found at line 7 : 10
DIGIT [ 0 ] Found at line 7 : 11
While_Statment [ while ] Found at line 9 : 13
Open_Expression [ ( ] Found at line 9 : 15
ID [ a ] Found at line 9 : 16
Not_Equal [ != ] Found at line 9 : 19
DIGIT [ 3 ] Found at line 9 : 21
Close_Expression [ ) ] Found at line 9 : 22
Begin_Block [ { ] Found at line 9 : 24
Print_Statment [ print ] Found at line 10 : 17
Open_Expression [ ( ] Found at line 10 : 18
ID [ a ] Found at line 10 : 19
Close_Expression [ ) ] Found at line 10 : 20
While_Statment [ while ] Found at line 11 : 17
Open_Expression [ ( ] Found at line 11 : 19
ID [ b ] Found at line 11 : 20
Not_Equal [ != ] Found at line 11 : 23
DIGIT [ 3 ] Found at line 11 : 25
Close_Expression [ ) ] Found at line 11 : 26
Begin_Block [ { ] Found at line 11 : 28
Print_Statment [ print ] Found at line 12 : 21
Open_Expression [ ( ] Found at line 12 : 22
ID [ b ] Found at line 12 : 23
Close_Expression [ ) ] Found at line 12 : 24
ID [ b ] Found at line 13 : 17
AssignmentStatement [ = ] Found at line 13 : 19
DIGIT [ 1 ] Found at line 13 : 21
InTop [ + ] Found at line 13 : 23
ID [ b ] Found at line 13 : 25
If_Statment [ if ] Found at line 14 : 18
Open_Expression [ ( ] Found at line 14 : 20
ID [ b ] Found at line 14 : 21
Equal [ == ] Found at line 14 : 24
DIGIT [ 2 ] Found at line 14 : 26
Close_Expression [ ) ] Found at line 14 : 27
Begin_Block [ { ] Found at line 14 : 29
Print_Statment [ print ] Found at line 16 : 25
Open_Expression [ ( ] Found at line 16 : 26
QUOTE [ " ] Found at line 16 : 27
CHAR [ t ] Found at line 16 : 28
CHAR [ h ] Found at line 16 : 29
CHAR [ e ] Found at line 16 : 30
CHAR [ r ] Found at line 16 : 31
CHAR [ e ] Found at line 16 : 32
SPACE [   ] Found at line 16 : 33
CHAR [ i ] Found at line 16 : 34
CHAR [ s ] Found at line 16 : 35
SPACE [   ] Found at line 16 : 36
CHAR [ n ] Found at line 16 : 37
CHAR [ o ] Found at line 16 : 38
SPACE [   ] Found at line 16 : 39
CHAR [ s ] Found at line 16 : 40
CHAR [ p ] Found at line 16 : 41
CHAR [ o ] Found at line 16 : 42
CHAR [ o ] Found at line 16 : 43
CHAR [ n ] Found at line 16 : 44
QUOTE [ " ] Found at line 16 : 45
Close_Expression [ ) ] Found at line 16 : 74
End_Block [ } ] Found at line 17 : 17
End_Block [ } ] Found at line 18 : 13
ID [ b ] Found at line 19 : 13
AssignmentStatement [ = ] Found at line 19 : 15
DIGIT [ 0 ] Found at line 19 : 17
ID [ a ] Found at line 20 : 13
AssignmentStatement [ = ] Found at line 20 : 15
DIGIT [ 1 ] Found at line 20 : 17
InTop [ + ] Found at line 20 : 18
ID [ a ] Found at line 20 : 19
End_Block [ } ] Found at line 21 : 9
End_Block [ } ] Found at line 22 : 5
END_OF_PROGRAM [ $ ] Found at line 23 : 1
Number of Errors is 0 :(
End of program 1
```

1.2 Test Case 2

```
/*Lexer "without any spaces"*/
/*LongTestCase-EverythingExceptBooleanDeclaration*/{/*IntDeclaration*/intaintba=0b=0/
*WhileLoop*/while(a!=3){print(a)while(b!=3){print(b)b=1+bif(b==2)
{/*PrintStatement*/print"there is no 1spoon"/*Thiswilldonothing*/)}}b=-0a=1+a}}$

Output:
Begin_Block [ { ] Found at line 25 : 52
TYPE [ int ] Found at line 25 : 73
ID [ a ] Found at line 25 : 74
TYPE [ int ] Found at line 25 : 77
ID [ b ] Found at line 25 : 78
ID [ a ] Found at line 25 : 79
AssignmentStatement [ = ] Found at line 25 : 80
DIGIT [ 0 ] Found at line 25 : 81
ID [ b ] Found at line 25 : 82
AssignmentStatement [ = ] Found at line 25 : 83
DIGIT [ 0 ] Found at line 25 : 84
While_Statment [ while ] Found at line 25 : 102
Open_Expression [ ( ] Found at line 25 : 103
ID [ a ] Found at line 25 : 104
Not_Equal [ != ] Found at line 25 : 106
DIGIT [ 3 ] Found at line 25 : 107
Close_Expression [ ) ] Found at line 25 : 108
Begin_Block [ { ] Found at line 25 : 109
Print_Statment [ print ] Found at line 25 : 114
Open_Expression [ ( ] Found at line 25 : 115
ID [ a ] Found at line 25 : 116
Close_Expression [ ) ] Found at line 25 : 117
While_Statment [ while ] Found at line 25 : 122
Open_Expression [ ( ] Found at line 25 : 123
ID [ b ] Found at line 25 : 124
Not_Equal [ != ] Found at line 25 : 126
DIGIT [ 3 ] Found at line 25 : 127
Close_Expression [ ) ] Found at line 25 : 128
Begin_Block [ { ] Found at line 25 : 129
Print_Statment [ print ] Found at line 25 : 134
Open_Expression [ ( ] Found at line 25 : 135
ID [ b ] Found at line 25 : 136
Close_Expression [ ) ] Found at line 25 : 137
ID [ b ] Found at line 25 : 138
AssignmentStatement [ = ] Found at line 25 : 139
DIGIT [ 1 ] Found at line 25 : 140
InTop [ + ] Found at line 25 : 141
ID [ b ] Found at line 25 : 142
If_Statment [ if ] Found at line 25 : 144
Open_Expression [ ( ] Found at line 25 : 145
ID [ b ] Found at line 25 : 146
Equal [ == ] Found at line 25 : 148
DIGIT [ 2 ] Found at line 25 : 149
Close_Expression [ ) ] Found at line 25 : 150
Begin_Block [ { ] Found at line 25 : 151
Print_Statment [ print ] Found at line 25 : 174
Open_Expression [ ( ] Found at line 25 : 175
QUOTE [ " ] Found at line 25 : 176
CHAR [ t ] Found at line 25 : 177
CHAR [ h ] Found at line 25 : 178
CHAR [ e ] Found at line 25 : 179
CHAR [ r ] Found at line 25 : 180
CHAR [ e ] Found at line 25 : 181
SPACE [   ] Found at line 25 : 182
CHAR [ i ] Found at line 25 : 183
CHAR [ s ] Found at line 25 : 184
SPACE [   ] Found at line 25 : 185
CHAR [ n ] Found at line 25 : 186
CHAR [ o ] Found at line 25 : 187
SPACE [   ] Found at line 25 : 188
Error: int not allowed in string [ 1 ] Found at line 25 : 189
CHAR [ s ] Found at line 25 : 190
CHAR [ p ] Found at line 25 : 191
CHAR [ o ] Found at line 25 : 192
CHAR [ o ] Found at line 25 : 193
CHAR [ n ] Found at line 25 : 194
QUOTE [ " ] Found at line 25 : 195
Close_Expression [ ) ] Found at line 25 : 217
End_Block [ } ] Found at line 25 : 218
End_Block [ } ] Found at line 25 : 219
ID [ b ] Found at line 25 : 220
AssignmentStatement [ = ] Found at line 25 : 221
Error: non recognized symbol [ - ] Found at line 25 : 222
DIGIT [ 0 ] Found at line 25 : 223
```

```
ID [ a ] Found at line 25 : 224
AssignmentStatement [ = ] Found at line 25 : 225
DIGIT [ 1 ] Found at line 25 : 226
InTop [ + ] Found at line 25 : 227
ID [ a ] Found at line 25 : 228
End_Block [ } ] Found at line 25 : 229
End_Block [ } ] Found at line 25 : 230
END_OF_PROGRAM [ $ ] Found at line 25 : 231
Number of Errors is 2 :(
Lexer failed :(
End of program 2
```

1.3 Test Case 3

```
/*keywords in strings*/
{/* Comment!!!*/ () print=whileif"teststring" intstringbooleanfalse true
==!=+ a 0123456789}$

Output:
Begin_Block [ { ] Found at line 27 : 1
Open_Expression [ ( ] Found at line 27 : 18
Close_Expression [ ) ] Found at line 27 : 19
Print_Statment [ print ] Found at line 27 : 25
AssignmentStatement [ = ] Found at line 27 : 26
While_Statment [ while ] Found at line 27 : 31
If_Statment [ if ] Found at line 27 : 33
QUOTE [ " ] Found at line 27 : 34
CHAR [ t ] Found at line 27 : 35
CHAR [ e ] Found at line 27 : 36
CHAR [ s ] Found at line 27 : 37
CHAR [ t ] Found at line 27 : 38
CHAR [ s ] Found at line 27 : 44
CHAR [ t ] Found at line 27 : 44
CHAR [ r ] Found at line 27 : 44
CHAR [ i ] Found at line 27 : 44
CHAR [ n ] Found at line 27 : 44
CHAR [ g ] Found at line 27 : 44
QUOTE [ " ] Found at line 27 : 45
TYPE [ int ] Found at line 27 : 49
TYPE [ string ] Found at line 27 : 55
TYPE [ boolean ] Found at line 27 : 62
Boolean_Value [ false ] Found at line 27 : 67
Boolean_Value [ true ] Found at line 27 : 72
Equal [ == ] Found at line 27 : 75
Not_Equal [ != ] Found at line 27 : 77
InTop [ + ] Found at line 27 : 78
ID [ a ] Found at line 27 : 80
DIGIT [ 0 ] Found at line 27 : 91
DIGIT [ 1 ] Found at line 27 : 91
DIGIT [ 2 ] Found at line 27 : 91
DIGIT [ 3 ] Found at line 27 : 91
DIGIT [ 4 ] Found at line 27 : 91
DIGIT [ 5 ] Found at line 27 : 91
DIGIT [ 6 ] Found at line 27 : 91
DIGIT [ 7 ] Found at line 27 : 91
DIGIT [ 8 ] Found at line 27 : 91
DIGIT [ 9 ] Found at line 27 : 91
End_Block [ } ] Found at line 27 : 92
END_OF_PROGRAM [ $ ] Found at line 27 : 93
Number of Errors is 0 :(
End of program 3
Number of Errors is 0 :(
End of program 3
```

1.4 Test Case 4

```
/*Testing for boolean statments*/
{
if true {
    print("if statement worked")
}
if false{
    print("if statement failed")
}
}$

Output:
Begin_Block [ { ] Found at line 30 : 1
If_Statment [ if ] Found at line 31 : 2
Boolean_Value [ true ] Found at line 31 : 7
Begin_Block [ { ] Found at line 31 : 9
Print_Statment [ print ] Found at line 32 : 9
Open_Expression [ ( ] Found at line 32 : 10
QUOTE [ " ] Found at line 32 : 11
CHAR [ i ] Found at line 32 : 13
CHAR [ f ] Found at line 32 : 13
SPACE [   ] Found at line 32 : 14
CHAR [ s ] Found at line 32 : 15
CHAR [ t ] Found at line 32 : 16
CHAR [ a ] Found at line 32 : 17
CHAR [ t ] Found at line 32 : 18
CHAR [ e ] Found at line 32 : 19
CHAR [ m ] Found at line 32 : 20
CHAR [ e ] Found at line 32 : 21
CHAR [ n ] Found at line 32 : 22
CHAR [ t ] Found at line 32 : 23
SPACE [   ] Found at line 32 : 24
CHAR [ w ] Found at line 32 : 25
CHAR [ o ] Found at line 32 : 26
CHAR [ r ] Found at line 32 : 27
CHAR [ k ] Found at line 32 : 28
CHAR [ e ] Found at line 32 : 29
CHAR [ d ] Found at line 32 : 30
QUOTE [ " ] Found at line 32 : 31
Close_Expression [ ) ] Found at line 32 : 32
End_Block [ } ] Found at line 33 : 1
```

```
If_Statment [ if ] Found at line 34 : 2
Boolean_Value [ false ] Found at line 34 : 8
Begin_Block [ { ] Found at line 34 : 9
Print_Statment [ print ] Found at line 35 : 9
Open_Expression [ ( ] Found at line 35 : 10
QUOTE [ " ] Found at line 35 : 11
CHAR [ i ] Found at line 35 : 13
CHAR [ f ] Found at line 35 : 13
SPACE [   ] Found at line 35 : 14
CHAR [ s ] Found at line 35 : 15
CHAR [ t ] Found at line 35 : 16
CHAR [ a ] Found at line 35 : 17
CHAR [ t ] Found at line 35 : 18
CHAR [ e ] Found at line 35 : 19
CHAR [ m ] Found at line 35 : 20
CHAR [ e ] Found at line 35 : 21
CHAR [ n ] Found at line 35 : 22
CHAR [ t ] Found at line 35 : 23
SPACE [   ] Found at line 35 : 24
CHAR [ f ] Found at line 35 : 25
CHAR [ a ] Found at line 35 : 26
CHAR [ i ] Found at line 35 : 27
CHAR [ l ] Found at line 35 : 28
CHAR [ e ] Found at line 35 : 29
CHAR [ d ] Found at line 35 : 30
QUOTE [ " ] Found at line 35 : 31
Close_Expression [ ) ] Found at line 35 : 32
End_Block [ } ] Found at line 36 : 1
End_Block [ } ] Found at line 37 : 1
END_OF_PROGRAM [ $ ] Found at line 37 : 2
Number of Errors is 0 :(
End of program 4
```

1.5 Test Case 5

```

/*This is suppose to crash at the top since there is a space before the first line */
 {
       int a
       a = 3
       int b
       b = 4
       a = b
       print(a)
       if (a == b) {
          print(a)
       }
     }$

Output:
Begin_Block [ { ] Found at line 40 : 2
TYPE [ int ] Found at line 41 : 10
ID [ a ] Found at line 41 : 12
ID [ a ] Found at line 42 : 8
AssignmentStatement [ = ] Found at line 42 : 10
DIGIT [ 3 ] Found at line 42 : 12
TYPE [ int ] Found at line 43 : 10
ID [ b ] Found at line 43 : 12
ID [ b ] Found at line 44 : 8
AssignmentStatement [ = ] Found at line 44 : 10
DIGIT [ 4 ] Found at line 44 : 12
ID [ a ] Found at line 45 : 8
AssignmentStatement [ = ] Found at line 45 : 10
ID [ b ] Found at line 45 : 12
Print_Statment [ print ] Found at line 46 : 12
Open_Expression [ ( ] Found at line 46 : 13
ID [ a ] Found at line 46 : 14
Close_Expression [ ) ] Found at line 46 : 15
If_Statment [ if ] Found at line 47 : 9
Open_Expression [ ( ] Found at line 47 : 11
ID [ a ] Found at line 47 : 12
Equal [ == ] Found at line 47 : 15
ID [ b ] Found at line 47 : 17
Close_Expression [ ) ] Found at line 47 : 18
Begin_Block [ { ] Found at line 47 : 20
Print_Statment [ print ] Found at line 48 : 15
Open_Expression [ ( ] Found at line 48 : 16
ID [ a ] Found at line 48 : 17
Close_Expression [ ) ] Found at line 48 : 18
End_Block [ } ] Found at line 49 : 8
End_Block [ } ] Found at line 50 : 6
END_OF_PROGRAM [ $ ] Found at line 50 : 7
Number of Errors is 0 :(
End of program 5
```

1.6 Test Case 6

```
/*General test case for stuff*/
{
        int a
        boolean b
        string c
       a = 9
        b = true
        {
           print(a)
           print(b)
           b = false
           c = "hello world"
           int b
           b = 0
           {
              print(c)
              a = 1 + 2 + a
              {
                 print(b)
              }
           }
           b = a
           print(b)
        }
        print(b)




}$

Begin_Block [ { ] Found at line 52 : 1
TYPE [ int ] Found at line 53 : 11
ID [ a ] Found at line 53 : 13
TYPE [ boolean ] Found at line 54 : 15
ID [ b ] Found at line 54 : 17
TYPE [ string ] Found at line 55 : 14
ID [ c ] Found at line 55 : 16
ID [ a ] Found at line 56 : 8
AssignmentStatement [ = ] Found at line 56 : 10
DIGIT [ 9 ] Found at line 56 : 12
ID [ b ] Found at line 57 : 9
AssignmentStatement [ = ] Found at line 57 : 11
Boolean_Value [ true ] Found at line 57 : 16
Begin_Block [ { ] Found at line 58 : 9
Print_Statment [ print ] Found at line 59 : 16
Open_Expression [ ( ] Found at line 59 : 17
ID [ a ] Found at line 59 : 18
Close_Expression [ ) ] Found at line 59 : 19
Print_Statment [ print ] Found at line 60 : 16
Open_Expression [ ( ] Found at line 60 : 17
ID [ b ] Found at line 60 : 18
Close_Expression [ ) ] Found at line 60 : 19
ID [ b ] Found at line 61 : 12
AssignmentStatement [ = ] Found at line 61 : 14
Boolean_Value [ false ] Found at line 61 : 20
ID [ c ] Found at line 62 : 12
AssignmentStatement [ = ] Found at line 62 : 14
QUOTE [ " ] Found at line 62 : 16
CHAR [ h ] Found at line 62 : 17
CHAR [ e ] Found at line 62 : 18
CHAR [ l ] Found at line 62 : 19
CHAR [ l ] Found at line 62 : 20
CHAR [ o ] Found at line 62 : 21
SPACE [   ] Found at line 62 : 22
CHAR [ w ] Found at line 62 : 23
CHAR [ o ] Found at line 62 : 24
CHAR [ r ] Found at line 62 : 25
CHAR [ l ] Found at line 62 : 26
CHAR [ d ] Found at line 62 : 27
QUOTE [ " ] Found at line 62 : 28
TYPE [ int ] Found at line 63 : 14
ID [ b ] Found at line 63 : 16
ID [ b ] Found at line 64 : 12
AssignmentStatement [ = ] Found at line 64 : 14
DIGIT [ 0 ] Found at line 64 : 16
Begin_Block [ { ] Found at line 65 : 12
Print_Statment [ print ] Found at line 66 : 19
Open_Expression [ ( ] Found at line 66 : 20
ID [ c ] Found at line 66 : 21
Close_Expression [ ) ] Found at line 66 : 22
ID [ a ] Found at line 67 : 15
AssignmentStatement [ = ] Found at line 67 : 17
DIGIT [ 1 ] Found at line 67 : 19
InTop [ + ] Found at line 67 : 21
DIGIT [ 2 ] Found at line 67 : 23
InTop [ + ] Found at line 67 : 25
ID [ a ] Found at line 67 : 27
Begin_Block [ { ] Found at line 68 : 15
Print_Statment [ print ] Found at line 69 : 22
Open_Expression [ ( ] Found at line 69 : 23
ID [ b ] Found at line 69 : 24
Close_Expression [ ) ] Found at line 69 : 25
End_Block [ } ] Found at line 70 : 15
End_Block [ } ] Found at line 71 : 12
ID [ b ] Found at line 72 : 12
AssignmentStatement [ = ] Found at line 72 : 14
ID [ a ] Found at line 72 : 16
Print_Statment [ print ] Found at line 73 : 16
Open_Expression [ ( ] Found at line 73 : 17
ID [ b ] Found at line 73 : 18
Close_Expression [ ) ] Found at line 73 : 19
End_Block [ } ] Found at line 74 : 9
Print_Statment [ print ] Found at line 75 : 13
Open_Expression [ ( ] Found at line 75 : 14
ID [ b ] Found at line 75 : 15
Close_Expression [ ) ] Found at line 75 : 16
End_Block [ } ] Found at line 80 : 1
END_OF_PROGRAM [ $ ] Found at line 80 : 2
Number of Errors is 0 :(
End of program 6
```

1.7 Test Case 7

```
/*Testing for no new lines in strings*/
"a
b"$

QUOTE [ " ] Found at line 83 : 1
CHAR [ a ] Found at line 83 : 2
Error: new line not allowed in string [  ] Found at line 83 : 2
CHAR [ b ] Found at line 84 : 1
QUOTE [ " ] Found at line 84 : 2
END_OF_PROGRAM [ $ ] Found at line 84 : 3
Number of Errors is 1 :(
Lexer failed :(
End of program 7
```

2 References

Java Regular Expression
Enum
Switch cases (Was more of an idea)
Double Quote For Regular Expressions
Groups in Regular Expressions
More Group stuff
Index for Regular Expressions
Check for end of line

I used these all to help me understand how to use Regular Expression and get the data that I need in order to make the tokens. Enum and Cases where not used but more of an ideas that I was thing about using but never did end up using them