

```

1  /*Long Test Case - Everything Except Boolean Declaration */
2  {
3      /* Int Declaration */
4      int a
5      int b
6      a = 0
7      b=0
8      /* While Loop */
9      while (a != 3) {
10         print(a)
11         while (b != 3) {
12             print(b)
13             b = 1 + b
14             if (b == 2) {
15                 /* Print Statement */
16                 print("there is no spoon" /* This will do nothing */ )
17             }
18         }
19         b = 0
20         a = 1+a
21     }
22 }
23 $

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 2...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 4...
LEXER --> | T_ID [ a ] on line 4...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 5...
LEXER --> | T_ID [ b ] on line 5...
LEXER --> | T_ID [ a ] on line 6...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 6...
LEXER --> | T_DIGIT [ 0 ] on line 6...
LEXER --> | T_ID [ b ] on line 7...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 7...
LEXER --> | T_DIGIT [ 0 ] on line 7...
LEXER --> | T_WHILE [ while ] on line 9...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 9...
LEXER --> | T_ID [ a ] on line 9...
LEXER --> | T_INEQUALITY_OP [ != ] on line 9...
LEXER --> | T_DIGIT [ 3 ] on line 9...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 9...
LEXER --> | T_OPENING_BRACE [ { ] on line 9...
LEXER --> | T_PRINT [ print ] on line 10...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 10...
LEXER --> | T_ID [ a ] on line 10...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 10...
LEXER --> | T_WHILE [ while ] on line 11...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 11...
LEXER --> | T_ID [ b ] on line 11...
LEXER --> | T_INEQUALITY_OP [ != ] on line 11...
LEXER --> | T_DIGIT [ 3 ] on line 11...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 11...
LEXER --> | T_OPENING_BRACE [ { ] on line 11...
LEXER --> | T_PRINT [ print ] on line 12...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 12...
LEXER --> | T_ID [ b ] on line 12...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 12...
LEXER --> | T_ID [ b ] on line 13...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 13...
LEXER --> | T_DIGIT [ 1 ] on line 13...
LEXER --> | T_ADDITION_OP [ + ] on line 13...
LEXER --> | T_ID [ b ] on line 13...
LEXER --> | T_IF [ if ] on line 14...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 14...

```

```

LEXER --> | T_ID [ b ] on line 14...
LEXER --> | T_EQUALITY_OP [ == ] on line 14...
LEXER --> | T_DIGIT [ 2 ] on line 14...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 14...
LEXER --> | T_OPENING_BRACE [ { ] on line 14...
LEXER --> | T_PRINT [ print ] on line 16...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 16...
LEXER --> | T_QUOTE [ " ] on line 16...
LEXER --> | T_CHAR [ t ] on line 16...
LEXER --> | T_CHAR [ h ] on line 16...
LEXER --> | T_CHAR [ e ] on line 16...
LEXER --> | T_CHAR [ r ] on line 16...
LEXER --> | T_CHAR [ e ] on line 16...
LEXER --> | T_CHAR [ ] on line 16...
LEXER --> | T_CHAR [ i ] on line 16...
LEXER --> | T_CHAR [ s ] on line 16...
LEXER --> | T_CHAR [ ] on line 16...
LEXER --> | T_CHAR [ n ] on line 16...
LEXER --> | T_CHAR [ o ] on line 16...
LEXER --> | T_CHAR [ ] on line 16...
LEXER --> | T_CHAR [ s ] on line 16...
LEXER --> | T_CHAR [ p ] on line 16...
LEXER --> | T_CHAR [ o ] on line 16...
LEXER --> | T_CHAR [ o ] on line 16...
LEXER --> | T_CHAR [ n ] on line 16...
LEXER --> | T_QUOTE [ " ] on line 16...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 16...
LEXER --> | T_CLOSING_BRACE [ } ] on line 17...
LEXER --> | T_CLOSING_BRACE [ } ] on line 18...
LEXER --> | T_ID [ b ] on line 19...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 19...
LEXER --> | T_DIGIT [ 0 ] on line 19...
LEXER --> | T_ID [ a ] on line 20...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 20...
LEXER --> | T_DIGIT [ 1 ] on line 20...
LEXER --> | T_ADDITION_OP [ + ] on line 20...
LEXER --> | T_ID [ a ] on line 20...
LEXER --> | T_CLOSING_BRACE [ } ] on line 21...
LEXER --> | T_CLOSING_BRACE [ } ] on line 22...
LEXER --> | T_EOPS [ $ ] on line 23...

```

```

1 /*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 spoon"/*Thiswillldonothing*/)}}b=0a=1+a}}$

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```
1 /*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 spoon"/*Thiswillldonothing*/)}}b=0a=1+a}}$
```

Remove the comments?



```
1 {intaintba=0b=0while(a!=3){print(a)while(b!=3){print(b)b=1+bif(b==2)
{print("there is no spoon")}}b=0a=1+a}}$
```

Start lexing.



```
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...
```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

```

lastPosition      -1
currentPosition   0

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

Not a keyword.

Not an id.

Is a symbol. Note that and its positions. found **symbol** in positions 0-0

lastPosition 0

currentPosition 0

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {i}ntaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing. (Can we stop here?)

lastPosition 0

currentPosition 1

found **symbol** in positions 0-0

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      0
currentPosition   2
found symbol in positions 0-0

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

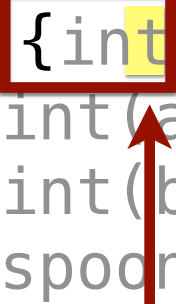
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
  {print(a)while(b!=3)
  {print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      0
currentPosition   3
found symbol in positions 0-0

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```



```

1 {int a int b a=0 b=0 while(a!=3)
  {print(a) while(b!=3)
  {print(b) b=1+b if(b==2){print("there is
no spoon")}} b=0 a=1+a}}$

```

Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing. (Can we stop yet?)

lastPosition 0

currentPosition 4

found **symbol** in positions 0-0

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {int a int b a=0 b=0 while(a!=3)
  {print(a) while(b!=3)
  {print(b) b=1+b if(b==2){print("there is
no spoon")}} b=0 a=1+a}}$

```

- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      0
currentPosition   5
found symbol in positions 0-0

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

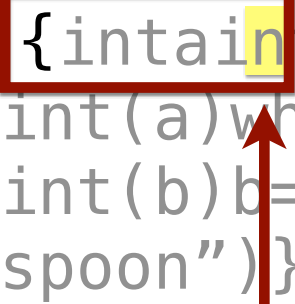
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing.

lastPosition 0

currentPosition 6

found **symbol** in positions 0-0

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
  {print(a)while(b!=3)
  {print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing. (Are we there yet?)

lastPosition 0

currentPosition 7

found **symbol** in positions 0-0

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      0
currentPosition   8
found symbol in positions 0-0

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      0
currentPosition   9
found symbol in positions 0-0

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

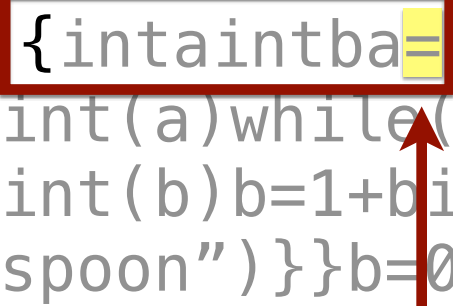
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



Not a keyword.

Not an id.

Is a symbol !!

Symbols, like white space (if present and outside of quotes) mean that we can stop moving ahead and see what we've got so far.

We've got a symbol, so consume the input, emit the token, and reset the pointers.

```

lastPosition      0
currentPosition   10
found symbol in positions 0-0

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...


```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```


lastPosition 1
currentPosition 1


LEXER --> | T_OPENING_BRACE [{] on line 1...
LEXER --> | T_VARIABLE_TYPE [int] on line 1...
LEXER --> | T_ID [a] on line 1...
LEXER --> | T_VARIABLE_TYPE [int] on line 1...
LEXER --> | T_ID [b] on line 1...
LEXER --> | T_ID [a] on line 1...
LEXER --> | T_ASSIGNMENT_OP [=] on line 1...
LEXER --> | T_DIGIT [0] on line 1...
LEXER --> | T_ID [b] on line 1...
LEXER --> | T_ASSIGNMENT_OP [=] on line 1...
LEXER --> | T_DIGIT [0] on line 1...
LEXER --> | T_WHILE [while] on line 1...
LEXER --> | T_OPENING_PARENTHESES [(] on line 1...
LEXER --> | T_ID [a] on line 1...
LEXER --> | T_INEQUALITY_OP [!=] on line 1...
LEXER --> | T_DIGIT [3] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [)] on line 1...
LEXER --> | T_OPENING_BRACE [{] on line 1...
LEXER --> | T_PRINT [print] on line 1...
LEXER --> | T_OPENING_PARENTHESES [(] on line 1...
LEXER --> | T_ID [a] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [)] on line 1...
LEXER --> | T_WHILE [while] on line 1...
LEXER --> | T_OPENING_PARENTHESES [(] on line 1...
LEXER --> | T_ID [b] on line 1...
LEXER --> | T_INEQUALITY_OP [!=] on line 1...
LEXER --> | T_DIGIT [3] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [)] on line 1...
LEXER --> | T_OPENING_BRACE [{] on line 1...
LEXER --> | T_PRINT [print] on line 1...
LEXER --> | T_OPENING_PARENTHESES [(] on line 1...
LEXER --> | T_ID [b] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [)] on line 1...
LEXER --> | T_ID [b] on line 1...
LEXER --> | T_ASSIGNMENT_OP [=] on line 1...
LEXER --> | T_DIGIT [1] on line 1...
LEXER --> | T_ADDITION_OP [+] on line 1...
LEXER --> | T_ID [b] on line 1...
LEXER --> | T_IF [if] on line 1...
LEXER --> | T_OPENING_PARENTHESES [(] on line 1...

LEXER --> | T_ID [b] on line 1...
LEXER --> | T_EQUALITY_OP [==] on line 1...
LEXER --> | T_DIGIT [2] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [)] on line 1...
LEXER --> | T_OPENING_BRACE [{] on line 1...
LEXER --> | T_PRINT [print] on line 1...
LEXER --> | T_OPENING_PARENTHESES [(] on line 1...
LEXER --> | T_QUOTE ["] on line 1...
LEXER --> | T_CHAR [t] on line 1...
LEXER --> | T_CHAR [h] on line 1...
LEXER --> | T_CHAR [e] on line 1...
LEXER --> | T_CHAR [r] on line 1...
LEXER --> | T_CHAR [e] on line 1...
LEXER --> | T_CHAR [] on line 1...
LEXER --> | T_CHAR [i] on line 1...
LEXER --> | T_CHAR [s] on line 1...
LEXER --> | T_CHAR [] on line 1...
LEXER --> | T_CHAR [n] on line 1...
LEXER --> | T_CHAR [o] on line 1...
LEXER --> | T_CHAR [] on line 1...
LEXER --> | T_CHAR [s] on line 1...
LEXER --> | T_CHAR [p] on line 1...
LEXER --> | T_CHAR [o] on line 1...
LEXER --> | T_CHAR [o] on line 1...
LEXER --> | T_CHAR [n] on line 1...
LEXER --> | T_QUOTE ["] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [)] on line 1...
LEXER --> | T_CLOSING_BRACE [}] on line 1...
LEXER --> | T_CLOSING_BRACE [}] on line 1...
LEXER --> | T_ID [b] on line 1...
LEXER --> | T_ASSIGNMENT_OP [=] on line 1...
LEXER --> | T_DIGIT [0] on line 1...
LEXER --> | T_ID [a] on line 1...
LEXER --> | T_ASSIGNMENT_OP [=] on line 1...
LEXER --> | T_DIGIT [1] on line 1...
LEXER --> | T_ADDITION_OP [+] on line 1...
LEXER --> | T_ID [a] on line 1...
LEXER --> | T_CLOSING_BRACE [}] on line 1...
LEXER --> | T_CLOSING_BRACE [}] on line 1...
LEXER --> | T_EOPS [\$] on line 1...


```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

Not a keyword.

Is an id.

(Actually, it is a char. But ids are single chars in our grammar, so we can take a shortcut here and match it as an id so long as we are not inside of quotes.)

```

lastPosition      1
currentPosition   1
found id in positions 1-1

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {in taintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing.

```

lastPosition      1
currentPosition   2
found id in positions 1-1

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```
1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$
```



Is a keyword.

lastPosition 1
currentPosition 3
found **int** in positions 1-3

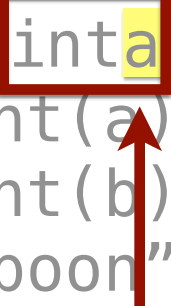
```
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
```

```
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...
```

```

1 {int a int b a=0 b=0 while(a!=3)
  {print(a) while(b!=3)
  {print(b) b=1+b if(b==2){print("there is
no spoon")}} b=0 a=1+a}}$

```



Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing.

lastPosition 1

currentPosition 4

found **int** in positions 1-3

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      1
currentPosition   5
found int in positions 1-3

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      1
currentPosition   6
found int in positions 1-3

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      1
currentPosition   7
found int in positions 1-3

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      1
currentPosition   8
found int in positions 1-3

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```



```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing.

lastPosition 1

currentPosition 9

found **int** in positions 1-3

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

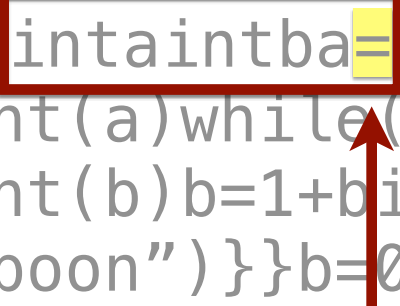
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



Not a keyword.

Not an id.

Is a symbol.

Symbols, like white space (if present and outside of quotes) mean that we can stop moving ahead and see what we've got so far.

We've got a symbol, so consume the input, emit the token, and reset the pointers.

```

lastPosition      1
currentPosition   10
found int in positions 1-3

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...


```

```

1 {int a int b a=0 b=0 while(a!=3)
  {print(a) while(b!=3)
   {print(b) b=1+b if(b==2){print("there is
no spoon")}} b=0 a=1+a}}$

```


lastPosition 4
currentPosition 4



```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {int a int b a=0 b=0 while(a!=3)
  {print(a) while(b!=3)
  {print(b) b=1+b if(b==2){print("there is
no spoon")}} b=0 a=1+a}}$

```



Not a keyword.

Is an id.

(Actually, it is a char. But ...)

lastPosition	4
currentPosition	4
found id in positions	4-4

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```


LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      4
currentPosition   5
found id in positions 4-4

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

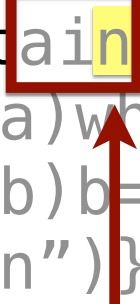
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      4
currentPosition   6
found id in positions 4-4

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

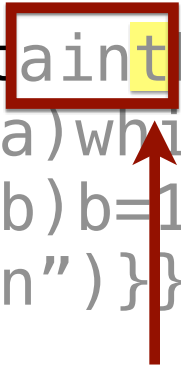
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



- Not a keyword.
- Not an id.
- Not a symbol.
- Not a digit.
- Not a char.
- Nothing.

```

lastPosition      4
currentPosition   7
found id in positions 4-4

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

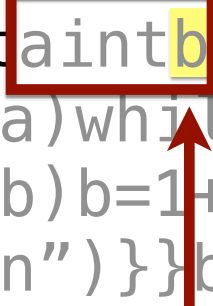
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {int a int b a=0 b=0 while(a!=3)
  {print(a) while(b!=3)
  {print(b) b=1+b if(b==2){print("there is
no spoon")}} b=0 a=1+a}}$

```



Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing.

lastPosition 4

currentPosition 8

found **id** in positions 4-4

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

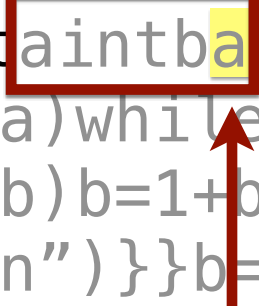
```



```

1 {int a int b a=0 b=0 while(a!=3)
  {print(a) while(b!=3)
  {print(b) b=1+b if(b==2){print("there is
no spoon")}} b=0 a=1+a}}$

```



Not a keyword.

Not an id.

Not a symbol.

Not a digit.

Not a char.

Nothing.

lastPosition 4

currentPosition 9

found **id** in positions 4-4

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

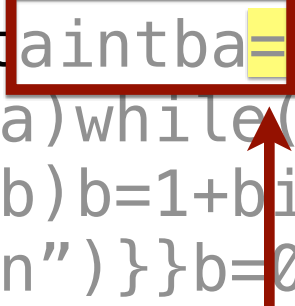
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {int a; int b = 0; while(a != 3)
  {print(a); while(b != 3)
   {print(b); b = 1 + b; if(b == 2) {print("there is
no spoon");}} b = 0; a = 1 + a;}}$

```



Not a keyword.

Not an id.

Is a symbol.

Consume input.

Emit token.

Reset pointers.

lastPosition 4

currentPosition 10

found **id** in positions 4-4

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...


```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)p=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```


lastPosition 5
currentPosition 5



```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```



```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



Not a keyword.

Is an id.

```

lastPosition      9
currentPosition   9
found id in positions 9-9

```

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```


LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



Not a keyword.

Not an id.

Is a symbol.

Consume input.

Emit token.

Reset pointers.

lastPosition 9

currentPosition 10

found **id** in positions 9-9

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



lastPosition	10
currentPosition	10



```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



Not a keyword.
Not an id.
Is a symbol.

lastPosition 10
currentPosition 10
found **symbol** in positions 10-10

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...


```



```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



Not a keyword.

Not an id.

Not a symbol.

Is a digit.

lastPosition 10

currentPosition 10

found **symbol** in positions 10-10

Remember how we wondered about stopping here? We can because there are no symbols with length > 2 in our grammar. This is either one of those symbols, or it's whatever we currently matched, or it's an error. In any case we don't need to look further.

Consume. Emit. Reset.

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```


LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```


lastPosition 11
currentPosition 11



```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...

```

```

LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```



Not a keyword.
 Not an id.
 Not a symbol.
 Is a digit.

lastPosition 11
 currentPosition 11
 found **digit** in positions 11-11

Digits are not part of anything else in our grammar so we can stop here.

Consume. Emit. Reset.

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```

```

1 {intaintba=0b=0while(a!=3)
{print(a)while(b!=3)
{print(b)b=1+bif(b==2){print("there is
no spoon")}}b=0a=1+a}}$

```

```

lastPosition      12
currentPosition   12

```

It's on and on and on on and on...
The lex don't stop until the break of dawn.

```

LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_VARIABLE_TYPE [ int ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_WHILE [ while ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_INEQUALITY_OP [ != ] on line 1...
LEXER --> | T_DIGIT [ 3 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_IF [ if ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_EQUALITY_OP [ == ] on line 1...
LEXER --> | T_DIGIT [ 2 ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_OPENING_BRACE [ { ] on line 1...
LEXER --> | T_PRINT [ print ] on line 1...
LEXER --> | T_OPENING_PARENTHESES [ ( ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CHAR [ t ] on line 1...
LEXER --> | T_CHAR [ h ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ r ] on line 1...
LEXER --> | T_CHAR [ e ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ i ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ ] on line 1...
LEXER --> | T_CHAR [ s ] on line 1...
LEXER --> | T_CHAR [ p ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ o ] on line 1...
LEXER --> | T_CHAR [ n ] on line 1...
LEXER --> | T_QUOTE [ " ] on line 1...
LEXER --> | T_CLOSING_PARENTHESES [ ) ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_ID [ b ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 0 ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_ASSIGNMENT_OP [ = ] on line 1...
LEXER --> | T_DIGIT [ 1 ] on line 1...
LEXER --> | T_ADDITION_OP [ + ] on line 1...
LEXER --> | T_ID [ a ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_CLOSING_BRACE [ } ] on line 1...
LEXER --> | T_EOPS [ $ ] on line 1...

```