**Test Cases for Scanner assignment**

1. **Input:** *sample1.txt*

(print

(+ 5 bla))

**Output:**

line 1: left paranthesis: (

line 1: program entry: print

line 2: left paranthesis: (

line 2: arithmetic operation: +

line 2: arithmetic const: 5

line 2: variable/function name: bla

line 2: right paranthesis: )

line 2: right paranthesis: )

1. **Input**: *sample2.txt*

(print ("hello world"))

**Output:**

line 1: left paranthesis: (

line 1: program entry: print

line 1: left paranthesis: (

line 1: variable/function name: hello

line 1: variable/function name: world

line 1: right paranthesis: )

line 1: right paranthesis: )

1. **Input**: *sample3.txt*

int N ; comment

(define-fun fibonacci (N)

(if (not N)

(= N 50))

(+ (fibonacci (- N 1)) (fibonacci (- N 2)))))

(print(+ 5 bla) and (mod 6 N) or (get-bool(N) and (\* 5 6)))

**Output:**

line 1: variable/function types: int

line 1: variable/function name: N

line 2: left paranthesis: (

line 2: function definition keyword: define-fun

line 2: variable/function name: fibonacci

line 2: left paranthesis: (

line 2: variable/function name: N

line 2: right paranthesis: )

line 3: left paranthesis: (

line 3: conditional operator: if

line 3: left paranthesis: (

line 3: boolean operator: not

line 3: variable/function name: N

line 3: right paranthesis: )

line 4: left paranthesis: (

line 4: arithmetic comparison: =

line 4: variable/function name: N

line 4: arithmetic const: 50

line 4: right paranthesis: )

line 4: right paranthesis: )

line 5: left paranthesis: (

line 5: arithmetic operation: +

line 5: left paranthesis: (

line 5: variable/function name: fibonacci

line 5: left paranthesis: (

line 5: arithmetic operation: -

line 5: variable/function name: N

line 5: arithmetic const: 1

line 5: right paranthesis: )

line 5: right paranthesis: )

line 5: left paranthesis: (

line 5: variable/function name: fibonacci

line 5: left paranthesis: (

line 5: arithmetic operation: -

line 5: variable/function name: N

line 5: arithmetic const: 2

line 5: right paranthesis: )

line 5: right paranthesis: )

line 5: right paranthesis: )

line 5: right paranthesis: )

line 5: right paranthesis: )

line 6: left paranthesis: (

line 6: program entry: print

line 6: left paranthesis: (

line 6: arithmetic operation: +

line 6: arithmetic const: 5

line 6: variable/function name: bla

line 6: right paranthesis: )

line 6: boolean operator: and

line 6: left paranthesis: (

line 6: arithmetic operation: mod

line 6: arithmetic const: 6

line 6: variable/function name: N

line 6: right paranthesis: )

line 6: boolean operator: or

line 6: left paranthesis: (

line 6: predefined functions: get-bool

line 6: left paranthesis: (

line 6: variable/function name: N

line 6: right paranthesis: )

line 6: boolean operator: and

line 6: left paranthesis: (

line 6: arithmetic operation: \*

line 6: arithmetic const: 5

line 6: arithmetic const: 6

line 6: right paranthesis: )

line 6: right paranthesis: )

line 6: right paranthesis: )

1. **Input**: *sample4.txt*

int N ; comment

(define-fun fibonacci (N)

(if (not N)

(= N 50))

(+ (fibonacci (- N 1)) (fibonacci (- N 2)))))

:unrecognised\_token

(print(+ 5 bla) and (mod 6 N) or (get-bool(N) and (\* 5 6))

Output:

line 1: variable/function types: int

line 1: variable/function name: N

line 2: left paranthesis: (

line 2: function definition keyword: define-fun

line 2: variable/function name: fibonacci

line 2: left paranthesis: (

line 2: variable/function name: N

line 2: right paranthesis: )

line 3: left paranthesis: (

line 3: conditional operator: if

line 3: left paranthesis: (

line 3: boolean operator: not

line 3: variable/function name: N

line 3: right paranthesis: )

line 4: left paranthesis: (

line 4: arithmetic comparison: =

line 4: variable/function name: N

line 4: arithmetic const: 50

line 4: right paranthesis: )

line 4: right paranthesis: )

line 5: left paranthesis: (

line 5: arithmetic operation: +

line 5: left paranthesis: (

line 5: variable/function name: fibonacci

line 5: left paranthesis: (

line 5: arithmetic operation: -

line 5: variable/function name: N

line 5: arithmetic const: 1

line 5: right paranthesis: )

line 5: right paranthesis: )

line 5: left paranthesis: (

line 5: variable/function name: fibonacci

line 5: left paranthesis: (

line 5: arithmetic operation: -

line 5: variable/function name: N

line 5: arithmetic const: 2

line 5: right paranthesis: )

line 5: right paranthesis: )

line 5: right paranthesis: )

line 5: right paranthesis: )

line 5: right paranthesis: )

1. **Input**: *sample5.txt*

:variable

**Output:**

Error message: Token cannot be recognized

Error message: No token can be recognized