Nihar Gupte
1001556441
Test Cases Document - Test cases, including each test path, the corresponding test data to execute the test path, and the expected output.

For the entirety of the document the following format shall be followed:
#. “name of the function in printtokens.java”

#	Test path	Test Data/ Input	Expected Output
1	…	…	…

- Each test case shall begin with the “Start” node (0th node) and end with the “End” node from its corresponding CFG diagram as given in the file - “CFG.pdf”
- All the test cases for a given function achieve edge coverage.
- Null strings denoted by “\0”
- Special instructions and format for the main method detailed at the end of the document before main method

---

1. String get_token(BufferedReader br)

#	Test path	Test Data/ Input	Expected Output
1	[Start, 1, 2, End]	\0	null
2	[Start, 1, 3, 4, 5, 4, 6, 7, End]	\n\0	null
3	[Start, 1, 3, 4, 6, 8, 9, End]	(	“(“
4	[Start, 1, 3, 4, 6, 8, 10, 12, 14, 15, End]	\na	“a”
5	[Start, 1, 3, 4, 6, 8, 10, 12, 13, 14, 15, End]	;	“;”
6	[Start, 1, 3, 4, 6, 8, 10, 11, 12, 14, 15, End]	”	“””
7	[Start, 1, 3, 4, 6, 8, 10, 12, 14, 16, 17, 18, 19, 21, 22, End]	AB	“AB”
8	[Start, 1, 3, 4, 6, 8, 10, 12, 14, 16, 17, 18, 20, 17, 21, 22, End]	ABC\0	“ABC”
9	[Start, 1, 3, 4, 6, 8, 10, 12, 14, 16, 17, 21, 23, 24, End]	ABC(	“ABC”
10	[Start, 1, 3, 4, 6, 8, 10, 12, 14, 16, 17, 21, 23, 25, 26, End]	ABC”	“ABC”
11	[Start, 1, 3, 4, 6, 8, 10, 12, 14, 16, 17, 21, 23, 25, 27, 28, End]	ABC;	“ABC”
12	[Start, 1, 3, 4, 6, 8, 10, 12, 14, 16, 17, 21, 23, 25, 27, 29, End]	“ABC”	““ABC””

2. boolean is_token_end(int str_com_id, int res)

#	Test path	Test Data/ Input	Expected Output
1	[Start, 1, 2, End]	0, -1	true
2	[Start, 1, 3, 4, 5, End]	1, 34	true
3	[Start, 1, 3, 4, 6, End]	1, 67	false
4	[Start, 1, 3, 7, 8, 9, End]	2, 10	true

5	[Start, 1, 3, 7, 8, 10, End]	2, 68	false
6	[Start, 1, 3, 7, 11, 12, End]	0, 40	true
7	[Start, 1, 3, 7, 11, 13, 14, End]	0, 32	true
8	[Start, 1, 3, 7, 11, 13, 15, End]	0, 69	false

3. boolean is_keyword(String str)

#	Test path	Test Data/ Input	Expected Output
1	[Start, 1, 2, End]	and	true
2	[Start, 1, 3, End]	nihar	false

4. boolean is_num_constant(String str)

#	Test path	Test Data/ Input	Expected Output
1	[Start, 1, 7, End]	A	false
2	[Start, 1, 2, 3, 5, End]	1A	false
3	[Start, 1, 2, 3, 4, 2, 6, End]	10	true

5. boolean is_str_constant(String str)

#	Test path	Test Data/ Input	Expected Output
1	[Start, 1, 7, End]	1	false
2	[Start, 1, 2, 3, 4, End]	“”	true
3	[Start, 1, 2, 3, 5, 2, 6, End]	“A	false

6. boolean is_identifier(String str)

#	Test path	Test Data/ Input	Expected Output
1	[Start, 1, 7, End]	1	false
2	[Start, 1, 2, 3, 5, End]	a!	false
3	[Start, 1, 2, 3, 4, 2, 6, End]	a1	true

7. void main(String[] args)

Clarification for inputs for main():

Test Case #1: String[] args has more than 1 entry (length>1), and the token stream is therefore not found. Hence the test data/input field is empty. Hence the output – "Error!,please give the token stream"

Test Case #2: This case corresponds to the system reading token stream / input from the file "emptyFile.txt." For this test case, the file is empty i.e. EOF reached on reading first character. Hence, no output as systems exits.

Test Case #3 and onwards: These cases correspond to the system reading one input at a time as a line from the command line interface. Different inputs are given in order to satisfy Program Level Testing and entire program wide edge coverage.

The format for program level testing is as follows (for simplification purposes):

[Start, 1, 2, non-mainMethod.7, 3, End]

Where non-mainMethod.7 represents the nodes of the 7th test case from the function non-mainMethod and the other nodes in the test case represent the nodes from the main function.

For example, case #3 as below is [Start, 1, 2, 6, 7, get_token.2, 8, 9, print_token.1, 10, get_token.1, 8, 11, End]. get_token.2 when expanded is the second test case for get_token method i.e. the nodes [Start, 1, 3, 4, 5, 4, 6, 7, End], so case 3 now becomes: [Start, 1, 2, 6, 7, [Start, 1, 3, 4, 5, 4, 6, 7, End], 8, 9, print_token.1, 10, get_token.1, 8, 11, End] and we continue replacing the other non-main method functions as well. In other words, wherever one sees get_token.2, this implies that it is a placeholder for all the nodes from 2nd test case of get_token function, as defined above, in the test cases table of get_token. This is done for simplifying the test cases and not having huge complex test cases for system wide testing.

Cases 14 to 17 satisfy edge coverage for print_token, in addition to cases 3-13 satisfying edge coverage for get_token. As noted below, every node in get_token (cases 1-12) and every node in print_token (cases 1-8) is covered.

#	Test path	Test Data/ Input	Expected Output
1	[Start, 1, 3, 5, End]		Error!,please give the token stream
2	[Start, 1, 3, 4, 6, 7, 8, 11, End]	<reading from emptyFile.txt>	(no output)
3	[Start, 1, 2, 6, 7, get_token.2, 8, 9, print_token.1, 10, get_token.1, 8, 11, End]	\n\0	(no output)
4	[Start, 1, 2, 6, 7, get_token.3, 8, 9, print_token.3, 10, get_token.1, 8, 11, End]	(	lparen.

5	[Start, 1, 2, 6, 7, get_token.4, 8, 9, print_token.4, 10, get_token.1, 8, 11, End]	\na	identifier,"a".
6	[Start, 1, 2, 6, 7, get_token.5, 8, 9, print_token.8, 10, get_token.1, 8, 11, End]	;	comment,;.
7	[Start, 1, 2, 6, 7, get_token.6, 8, 9, print_token.1, 10, get_token.1, 8, 11, End]	”	error,""".
8	[Start, 1, 2, 6, 7, get_token.7, 8, 9, print_token.4, 10, get_token.1, 8, 11, End]	AB	identifier,"AB".
9	[Start, 1, 2, 6, 7, get_token.8, 8, 9, print_token.4, 10, get_token.1, 8, 11, End]	ABC\0	identifier,"ABC".
10	[Start, 1, 2, 6, 7, get_token.9, 8, 9, print_token.4, 10, get_token.1, 8, 11, End]	ABC(	identifier,"ABC". lparen.
11	[Start, 1, 2, 6, 7, get_token.10, 8, 9, print_token.4, 10, get_token.1, 8, 11, End]	ABC”	error,"ABC"".
12	[Start, 1, 2, 6, 7, get_token.11, 8, 9, print_token.4, 10, get_token.1, 8, 11, End]	ABC;	identifier,"ABC". comment,;.
13	[Start, 1, 2, 6, 7, get_token.12, 8, 9, print_token.4, 10, get_token.1, 8, 11, End]	ABC	identifier,"ABC".
14	[Start, 1, 2, 6, 7, get_token.12, 8, 9, print_token.2, 10, get_token.1, 8, 11, End]	and	keyword,"and".
15	[Start, 1, 2, 6, 7, get_token.12, 8, 9, print_token.5, 10, get_token.1, 8, 11, End]	5	numeric,5.
16	[Start, 1, 2, 6, 7, get_token.12, 8, 9, print_token.6, 10, get_token.1, 8, 11, End]	#c	character,"c".
17	[Start, 1, 2, 6, 7, get_token.12, 8, 9, print_token.7, 10, get_token.1, 8, 11, End]	“hello”	string,"hello".