java

```java
class Example {
    int x;
    float y;

    void main() {
        int a = 5;
        float b = 3.14;

        if (a > b) {
            print(a);
        } else {
            print(b);
        }

        while (a > 0) {
            a = a - 1;
        }
    }

    int add(int a, int b) {
        return a + b;
    }
}
```

Salida esperada para inputValid.txt

```
-- class
--- IDENTIFIER: Example
---- {
----- RULE_TYPE
----- TYPE: int
--- IDENTIFIER: x
-- ;
----- RULE_TYPE
----- TYPE: float
--- IDENTIFIER: y
-- ;
----- RULE_METHODS
----- RULE_TYPE
----- TYPE: void
----- IDENTIFIER: main
----- (
------ RULE_PARAMS
----- )
----- {
-- RULE_BODY
--- RULE_VARIABLE
----- RULE_TYPE
----- TYPE: int
--- IDENTIFIER: a
--- =
--- RULE_EXPRESSION
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- LITERAL: 5
-- ;
--- RULE_VARIABLE
----- RULE_TYPE
----- TYPE: float
--- IDENTIFIER: b
--- =
--- RULE_EXPRESSION
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- LITERAL: 3.14
-- ;
--- RULE_IF
```

```
--- if
--- (
--- RULE_EXPRESSION
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- IDENTIFIER: a
------ >
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- IDENTIFIER: b
--- )
---- RULE_STATEMENT_BLOCK
---- {
-- RULE_BODY
--- RULE_CALL_METHOD
--- IDENTIFIER: print
--- (
---- RULE_PARAM_VALUES
--- RULE_EXPRESSION
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- IDENTIFIER: a
--- )
-- ;
---- }
--- else
---- RULE_STATEMENT_BLOCK
---- {
-- RULE_BODY
--- RULE_CALL_METHOD
--- IDENTIFIER: print
--- (
---- RULE_PARAM_VALUES
--- RULE_EXPRESSION
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- IDENTIFIER: b
--- )
-- ;
---- }
```

```
--- RULE_WHILE
--- while
--- (
--- RULE_EXPRESSION
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- IDENTIFIER: a
------ >
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- LITERAL: 0
--- )
---- RULE_STATEMENT_BLOCK
---- {
-- RULE_BODY
--- RULE_ASSIGNMENT
--- IDENTIFIER: a
--- =
--- RULE_EXPRESSION
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- IDENTIFIER: a
------- -
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- LITERAL: 1
-- ;
---- }
----- }
----- RULE_METHODS
----- RULE_TYPE
----- TYPE: int
----- IDENTIFIER: add
----- (
------ RULE_PARAMS
----- RULE_TYPE
----- TYPE: int
------ IDENTIFIER: a
------ ,
----- RULE_TYPE
```

```
----- TYPE: int
------ IDENTIFIER: b
----- )
----- {
-- RULE_BODY
--- RULE_RETURN
--- return
--- RULE_EXPRESSION
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- IDENTIFIER: a
------- +
-------- RULE_A
--------- RULE_B
---------- RULE_C
---------- IDENTIFIER: b
-- ;
----- }
---- }
Parsing completed with recovery.
```