**Expr: expression**

**Var: variable**

**Const: constant**

**Arg: argument**

1.

**type** CALL **feature**

args: EXPRESSION\*

r: ROUTINE

**syntax**: Java

r (args)

**syntax**: Eiffel

args.count ≠ 0 -> r (args) | r

**end**

**pattern** SWAPPPED\_ARGUMENTS **for**

c: CALL

**with**

a1, a2: EXPRESSION

**where**

a1 c.args ; a2 c.args

a1.index ≠ a2.index

**fix**

c [a1 <- a2, a2 <- a1]

**end**

**pattern** PLUS\_MINUS **for**

e: SUM

**with**

e1, e2: EXPRESSION

**where**

e1 = e.first

e2 = e.second

**fix**

DIFFERENCE [first: e1; second: e2]

**End**

**2.**

**pattern** PLUS\_MINUS **for**

e: == **PLUS**[left: e1, right: e2]

***description***

*Buggy: If (expr1 == expr2) {}*

*Fixed: If(expr1 != expr2) {}*

**fix**

**MINUS**[left: e1, right: e2]

**End**

**3.**

**Pattern** replacement of == with !=

**EQ\_BIN\_OP[left: e1, op: ==, right: e2] ⇒**

**NOT\_EQ\_ BIN\_OP[left: e1, op: !=, right: e2]**

**BIN\_OP: left: EXPR, op: = | != | < | ≤, right: EXPR**

**\begin{lstlisting}[captionpos=b, basicstyle=\fontsize{0.27cm}{0.27cm}]**

**pattern EQ\_NOTEQ for**

**e: EQ\_BIN\_OP**

**with**

**e1, e2: EXPRESSION**

**where**

**e1 = e.first**

**e2 = e.second**

**fix**

**NOT\_EQ\_ BIN\_OP [first $\leftarrow$ e1, second $\leftarrow$ e2]**

**end**

**\end{lstlisting}**

4.

**Pattern** False\_True

**CALL**[f, args: [left\*, False, right\*] ⇒ **CALL**[f, [\*left, True, \*right]]

**f(x, y) CALL[f, [x, y]]**

***Description:***

***Buggy: func(listArg1, False, listArg2)***

***Fixed: func(listArg1, True, listArg2)***

**5.**

**Pattern** replacement of == with !=

Expr1, Expr2 Expr

**Buggy**: If (expr1 == expr2) {}  
**Fixed**: If(expr1 != expr2) {}

1. Frequency: 0
   1. Buggy: If (expr1 == expr2) {}
   2. Fixed: If(expr1 != expr2) {}
2. Frequency: 1
   1. **Buggy: If (arg1 != arg2) {}**
   2. **Fixed: If(arg1 == arg2) {}**
3. Frequency: 1
   1. **Buggy: If (arg1 < arg2) {}**
   2. **Fixed: If(arg1 <= arg2) {}**
4. Frequency: 1
   1. **Buggy: If (arg1 <= arg2) {}**
   2. **Fixed: If(arg1 < arg2) {}**
5. Frequency: 1
   1. **Buggy: If (arg1 <= arg2) {}**
   2. **Fixed: If(arg1 == arg2) {}**
6. Frequency: 0
   1. Buggy: If (arg1 == arg2) {}
   2. Fixed: If(arg1 <= arg2) {}
7. Frequency: 1
   1. **Buggy: If (arg1 != arg2) {}**
   2. **Fixed: If(arg1 < arg2) {}**
8. Frequency: 0
   1. Buggy: If (arg1 < arg2) {}
   2. Fixed: If(arg1 != arg2) {}
9. Frequency:
   1. **Buggy: If (arg1 == arg2) {}**
   2. **Fixed: If(arg1 < arg2) {}**
10. Frequency: 0
    1. Buggy: If (arg1 < arg2) {}
    2. Fixed: If(arg1 == arg2) {}
11. Frequency:  **1**
    1. **Buggy: Return true;**
    2. **Fixed: Return false;**
12. Frequency: 0
    1. Buggy: Return false;
    2. Fixed: Return true;
13. Frequency: 1
    1. **Buggy: func(listArg1, False, listArg2)**
    2. **Fixed: func(listArg1, True, listArg2)**
14. Frequency:  **2**
    1. **Buggy: func(listArg1, True, listArg2)**
    2. **Fixed: func(listArg1, False, listArg2)**
15. Frequency: 1
    1. **Buggy: Break;**
    2. **Fixed: Continue;**
16. Frequency: 0
    1. Buggy: Break;
    2. Fixed: Continue;
17. Frequency: 0
    1. **Buggy: obj.someMember**
    2. **Fixed: if(obj != null) { obj.someMember }**
18. Frequency: 1
    1. **Buggy: var = expr;**
    2. **Fixed: var = expr + 1;**
19. Frequency:
    1. Buggy: var = expr;
    2. Fixed: var = expr - 1;
20. Frequency: 2
    1. **Buggy: func(listArg1, val, listArg2)**
    2. **Fixed: func(listArg1, val + 1, listArg2)**
21. Frequency: 1
    1. **Buggy: func(listArg1, val, listArg2)**
    2. **Fixed: func(listArg1, val - 1, listArg2)**
22. Frequency: 1
    1. **Buggy: if(expr1 < expr2)**
    2. **Fixed: if(expr1 < (expr2 - 1))**
23. Frequency: 2
    1. **Buggy: if (... (expr1 < expr2) ...)**
    2. **Fixed: if(expr1 < (expr2 + 1))**
24. Frequency: 0
    1. Buggy: if(expr1 > expr2)
    2. Fixed: if(expr1 > (expr2 - 1))
25. Frequency: 0
    1. Buggy: if(expr1 > expr2)
    2. Fixed: if(expr1 > (expr2 + 1))
26. Frequency: 0
    1. Buggy: if(expr1 < expr2)
    2. Fixed: if((expr1+1) < expr2)
27. Frequency: 0
    1. Buggy: if(expr1 < expr2)
    2. Fixed: if((expr1 -1) < expr2))
28. Frequency: 0
    1. Buggy: if(expr1 > expr2)
    2. Fixed: if((expr1 + 1) > expr2)
29. Frequency: 0
    1. Buggy: if(expr1 > expr2)
    2. Fixed: if((expr1 - 1) > expr2)

BM 5 Dec. 2023

deferred class BUG\_FIX\_PATTERN feature

original: CONSTRUCT

deferred end

corrected: CONSTRUCT

deferred end

end

class PLUS\_MINUS inherit

BUG\_FIX\_PATTERN

feature

a, b: EXPRESSION

original: EQUALITY\_EXPRESSION

ensure

original.first = a

original.second = b

end

corrected: EQUALITY\_EXPRESSION

ensure

original.first = b

original.second = a

end

end