MS1:Special Strings at the End of Programs

To figure out what kind of mutants would result in similar misleading location information, we try to replace the "//" with other tokens, like "+", "a", "1", "!", "=", and "&", and replace the following string "author_Oliver_PETRUCCI" with different strings like "author", "author Oliver", "author + Oliver", "import java.io", "protected void Changes()". We also try to move the mutated source code (the last line of source code) to other places. Evaluation results suggest that similar misleading location information appears if and only if:

• The mutated source code is on the last line of the program;

• The last line of source code begins with a special character (i.e., "%" or "!");

• The special character is followed by an import statement or a string composed of English charters and numbers only.

MS2:Missing Left Bracket in IF Statement

To identify the exact specification of mutants that would result in similar misleading location information, we try to replace the left bracket and its preceding space with different tokens, to remove the left racket but keep its preceding space, and to make similar mutation on other IF statements and other statements (like FOR and WHILE statements). Evaluation results suggest that most of such mutations would not lead to similar misleading diagnoses. As a conclusion of the analysis, similar misleading location information appears if and only if:

• The mutation revises an IF statement;

• And the conditional statement within the revised IF statement is a single variable instead of a complex expression (like x +Y > 0);

• The revised IF statement is followed by an ELSE statement;

• The revised IF statement is not the first IF statement within the enclosing method, i.e., it follows another IF statement within the same method;

• The revision removes the first '(' of the IF statement and its preceding spaces. It can even move keyword if.

MS3: Missing Keyword "Try"

To explicitly specify the exact mutants that would result in
similar misleading location information, we remove keyword "try" in different places, remove the "{" following the removed keyword at the same time, and replace the keyword with randomly generated tokens. Evaluation results suggest that similar misleading location information appears when the mutant meets all of the following conditions:

• The mutant removes a keyword "try" only;

• The removed TRY statement is nested by another TRY statement;

• Both the removed TRY statement and its enclosing TRY statement are accompanied with CATCH or FINALLY statements.

Notably, the modified TRY statement should be nested by another TRY statement. Otherwise, the misleading location information cannot be generated.

MS4: Special and Confusing Tokens Appended to Right Braces

To figure out what kind of mutants would result in such kind of misleading location

information, we try to append different tokens to right braces, to append tokens to right braces in different places, and to append tokens to other special block separators, like left braces "{" and right brackets ")". Evaluation results suggest that similar misleading location information appears when the mutant meets all of the following conditions:

• The mutant appends a toke to a right brace;
• The appended brace is immediately followed by another right brace;
• The appended token begins with one of the following strings: "%%", "+[", and "\\".
• The rest of appended token is composed of English characters only.

MS5: Incomplete Block Comments

To investigate what kind of incomplete block comments would result in similar misleading location information, we try to mutate block comments in different places. Evaluation results suggest that similar misleading location information appears when the mutant meets all of the following conditions:

• The mutant destroys the ending mark ("*/") of a block comment (Noted as BC1);
• The involved document contains at least one more block comment (Noted as BC2) after the revised one;
• The two block comments (i.e., BC1 and BC1) are separated by some statements, and removing such statements would result in syntactic errors within the involved document.

MS6: Misleading Location Information Due to Comments

We try various similar mutations around code comments, and results suggest that similar misleading location information appears when the mutant meets all of the following conditions:

• The mutant revises a single line of source code that immediately follows code comments;
• The revised code is intended to declare a type (class or interface);
• The revised type declaration begins with "** extends/implements " or "** (" where ** represents any number of English letters.

MS7: Incorrect Instantiation of Anonymous Class

To figure out what kind incorrect instantiation of anonymous classes could result in similar misleading location information by ECJ, we revise different instantiation in different places with different mutation operators. Evaluation results suggest that a mutant results in similar diagnose if:

• It revises the instantiation of an anonymous class;
• The keyword "new" in the instantiation statement is typed in as "&*" or "/*" by mistake where * represent any number of English letters.