

Pattern-Based Precondition Inference from Method Implementations

Shanyue Zhou

Master of Science

School of Computer Science
McGill University
Montreal, Quebec, Canada

August 2026

A thesis submitted to McGill University in partial
fulfillment of the requirements of the degree of
Master of Science

©Shanyue Zhou, 2026

Abstract

Method preconditions define the conditions that must be satisfied before a method is called. Often, such conditions are not documented and must be inferred. We present Prex, a pattern-based static analysis approach that infers preconditions from explicit checks in Java source code. For each precondition, Prex reports the condition required for a valid call and the reported exception in the case of a violation. Prex also attempts code transformations to convert the condition detected within a method’s implementation into an expression that can be evaluated by client code. It leverages the concept of a mutation boundary to separate precondition checks from exceptions that may occur after the method changes the program state. We evaluated Prex on 46 open source Java projects and manually reviewed independent samples of 150 reports. Among the 150 Prex reports, 149 were correct and 125 of the correct reports were executable by client code. In a systematic comparison with an alternative approach based on path expansion, Prex provided more precise results. In total, 122 of the 149 correct reports had no relevant method documentation, and 41

required nontrivial source code analysis to recover manually. Overall, explicit source-code checks provide precondition information that can be absent from method documentation, and Prex can help recover them.

Abrégé

Les préconditions de méthode définissent les conditions qui doivent être satisfaites avant l'appel d'une méthode. Souvent, ces conditions ne sont pas documentées et doivent être inférées. Nous présentons Prex, une approche d'analyse statique fondée sur des patrons qui infère des préconditions à partir de vérifications explicites dans le code source Java. Pour chaque précondition, Prex signale la condition requise pour un appel valide et l'exception signalée en cas de violation. Prex tente également d'effectuer des transformations de code afin de convertir la condition détectée dans l'implémentation d'une méthode en une expression pouvant être évaluée par le code client. Elle exploite le concept de frontière de mutation pour distinguer les vérifications de préconditions des exceptions qui peuvent survenir après que la méthode a modifié l'état du programme. Nous avons évalué Prex sur 46 projets Java à code source ouvert et avons examiné manuellement des échantillons indépendants de 150 rapports. Parmi les 150 rapports de Prex, 149 étaient corrects et 125 des rapports corrects pouvaient être évalués par le code client. Dans une comparaison systématique

avec une approche alternative fondée sur l’expansion de chemins, Prex a fourni des résultats plus précis. Au total, 122 des 149 rapports corrects ne disposaient d’aucune documentation de méthode pertinente, et 41 nécessitaient une analyse non triviale du code source pour être retrouvés manuellement. Dans l’ensemble, les vérifications explicites du code source fournissent des informations sur les préconditions qui peuvent être absentes de la documentation des méthodes, et Prex peut aider à les retrouver.

Contributions of Authors

The author is responsible for the entirety of this thesis, working under the supervision and guidance of Professor Martin P. Robillard.

Acknowledgements

I express my dearest gratitude towards Prof. Martin Robillard. He has been a great mentor throughout my graduate studies. His seriousness towards research and his great insight in software engineering have helped me throughout the entire master's program. His invaluable guidance is a great asset to my future research.

I would also like to thank my parents, without their support, mentally and financially, I would not have been able to have the chance to pursue my graduate studies.

I thank the friends and members of the Software Technology Lab, Avinash, Bhagya, Billy, Divya, Edwin, Fanny, Giane, Keyu, Lanese, Shalaleh, Tamara, Veronica, Vy, and Professor Jin Guo, for their support and friendship during my graduate studies. I would also like to thank my friends, Ben, Cindy, Claire, Daniel, Feiyang, Jenny, Mike, Talise, Wen, Yifan, and Yuan.

Contents

Acronyms	x
1 Introduction	1
2 Related Work	8
2.1 Scope and Organization of Related Work	9
2.2 Implementation-Based Analyses	11
2.2.1 Exception Behavior Extraction	11
2.2.2 Documentation and Test-Based Condition Recovery	13
2.3 Precondition Mining from Client-Side Patterns	14
2.4 Contract and Invariant Inference	15
2.4.1 Object Protocol Analysis	15
2.4.2 Static Contract and Annotation Inference	16
2.4.3 Dynamic Invariant Inference	16
3 Approach	18
3.1 Scope and Terminology	18
3.2 Problem Formulation	20
3.3 Analysis Algorithm	21
3.4 Implementation Details	24
3.5 Limitations and Exclusions	30
4 Evaluation	32

4.1	Sampling Approach and Dataset	33
4.2	RQ1: Precision of Prex	35
4.3	RQ2: Testability	36
4.4	RQ3: Comparison	37
4.5	RQ4: Value	40
5	Discussion	44
5.1	RQ1: Precision of Prex Precondition Reports	44
5.2	RQ2: Client Testability of Extracted Preconditions	45
5.3	RQ3: Comparison with WIT's Path-Expansion Analysis	46
5.4	RQ4: Complementing Precondition Documentation	47
5.5	Tradeoffs and Threats to Validity	48
5.5.1	Sampling Scope	48
5.5.2	Manual Classification	48
5.6	Future Work	49
5.6.1	Increasing Analysis Coverage	49
5.6.2	Using Extracted Preconditions in Development	50
6	Conclusion	51
	Bibliography	53
A	Evaluation Samples	59
A.1	Prex Evaluation Sample	59
A.2	WIT Evaluation Sample	79

List of Tables

2.1	Evidence and reported conditions in related precondition and specification inference research directions.	10
4.1	Evaluation Dataset	34
4.2	Precision comparison on independent samples from each tool.	38
4.3	Coding Guide for Precondition Documentation	41
4.4	Documentation quality and source recovery effort among correct Prex reports. . . .	42
A.1	Prex evaluation sample (150 reports): Reported preconditions.	60
A.2	Prex evaluation sample (150 reports): Manual-review labels.	74
A.3	WIT evaluation sample (150 reports): Reported preconditions.	80
A.4	WIT evaluation sample (150 reports): Manual-review labels.	97

Acronyms

API	Application Programming Interface
AST	Abstract Syntax Tree
CI	Confidence Interval
DBC	Design by Contract
JDK	Java Development Kit
JSON	JavaScript Object Notation
OSS	Open-Source Software
RQ	Research Question
WIT	Path-expansion tool for extracting exception preconditions

1

Introduction

Developers routinely define conditions on the execution of methods [1]. In modern programming languages such as Java, a method signature gives basic information about a call, including the method name, the input argument types, and the return type. This information helps callers know how to form a syntactically valid call. However, the signature does not explain all conditions that must hold for the call to be valid [2]. For example, an integer argument may need to be within a certain range, or a method may be valid only during a certain state of the method's target object. Calling a method when such conditions are violated can lead to exceptions, incorrect results, or undefined behavior [1, 3].

This missing information can make a method interface ambiguous [4]. From the signature alone, callers may not know whether a method rejects invalid input, silently handles it, or assumes that such input will never be provided. Meyer introduced Design by Contract (DbC) to reduce

Introduction

this ambiguity by treating an interface as a contract between the caller and the implementation [1, 5]. The caller is responsible for satisfying the requirements stated by the contract. In return, the implementation is responsible for providing the promised result when those requirements hold.

The contract is usually described using three kinds of properties: preconditions, postconditions, and invariants [1, 5]. A precondition is a predicate that must be true when a method starts executing. A postcondition is a predicate that must be true when the method completes and the invariant is a predicate about object state that must hold across public operations, often at method entry and exit points [1]. These properties make explicit which part of the behavior is the caller's responsibility and which part is the implementation's responsibility [6].

This thesis focuses on preconditions, the caller-side part of the contract. Listing 1.1 provides an example of a method with two precondition checks in Java.

```
private boolean readOnly;  
private String content = "";  
  
/**  
 * @throws IllegalStateException if the document is read-only  
 * @throws IllegalArgumentException if text is null  
 */  
public void append(String text) {  
    if (readOnly) {  
        throw new IllegalStateException();  
    }  
    if (text == null) {
```

Introduction

```
        throw new IllegalArgumentException();
    }
    content += text;
}
```

Listing 1.1: Two precondition checks in a document append operation

The comment above the method documents exceptions for callers. Each `@throws` entry names an exception and may state its condition. The first entry states that `append` throws an `IllegalStateException` if the document is read-only. This documentation entry informs authors of client code, which calls the analyzed method, about the constraint in ordinary language, but not how to determine whether the document is read-only before calling `append`. The second entry directly identifies `null` as the rejected input value, which callers can check before the method call.

Each conditional code block works as a *precondition check*. Preconditions checks can depend on both the current state of a method's target object and the input passed to it. The first `if` block checks the `readOnly` field, which constitutes private state of the object rather than user input. When `readOnly` is true, the method throws an `IllegalStateException`. The second `if` statement checks the input argument `text` and rejects `null` with an `IllegalArgumentException`. After both checks pass, the obligation on the caller is considered satisfied, and the method can execute the append operation.

For these preconditions to be helpful, they must be understandable and testable before the call is made. A precondition is *testable* when the condition can be tested before the call with input arguments and publicly accessible values. A condition over inaccessible details of the object's state, such as involving private implementation structure, may still be a valid precondition but the

Introduction

client cannot check it directly. Untestable preconditions break the principle of Design by Contract. Since the caller is responsible for satisfying the preconditions, the caller needs a practical way to validate them before calling the method [7].

Preconditions must also be communicated clearly to callers. In Java, developers can indicate preconditions through documentation, source code and, in a limited sense, annotations [8]. However, in many software projects, there is no formal support for documenting and reasoning about preconditions, and usage is haphazard [9].

In this context, users of software packages are often left uncertain about the preconditions for calling a method. If the documentation is lacking or incorrect, violated preconditions can lead to exceptions and undefined behaviors. Avoiding this eventuality can require defensive programming in user code [10]. The goal of our research is to provide automated, precise, and deterministic support for identifying preconditions in existing source code. This thesis contributes a solution for the Java language based on general principles intended to carry over to comparable programming languages.

When documentation does not state a precondition, the method implementation can provide another source of evidence [11]. Static analysis provides a way to examine source code without executing the program. It can systematically inspect an entire codebase or selected parts of one to recover information from the structure of the code [12].

We present Prex (Precondition Extractor), a tool-supported static analysis approach to extract explicit precondition checks from Java source code and convert them into precondition descriptors. Each descriptor records the *guard* condition required for a valid call, the violation *signal*, and a

Introduction

mapping from guard symbols to their method boundary representations. The analysis recognizes common Java precondition patterns, including assertions and conditional throws. Prex leverages the concept of a *mutation boundary* to exclude runtime failures that can occur only after the method has started changing program state. Finally, it attempts to express guards using method arguments and public query methods so that client code can evaluate them before invocation. These guards are therefore testable. Some preconditions depend on hidden state for which no public expression exists; Prex identifies such expressions as *untestable*.

The design of Prex is conservative. If it cannot establish that a candidate is an explicit precondition check before the mutation boundary, it discards the candidate. Every reported descriptor can be traced to the source expressions and control-flow decisions used to construct it. This makes the analysis deterministic and allows its results to be checked and reproduced. The approach does not rely on generated text, training data, or runtime executions.

The closest alternative to the Prex approach is WIT, which uses bounded path analysis to extract conditions that lead Java methods to throw exceptions [3]. WIT targets exception behavior: a reported condition explains when an exception can be reached. Our goal is more specific, as we seek explicit checks that define obligations on the user before method execution. An exception path is therefore excluded if it crosses the mutation boundary, even if its path condition is valid. Other work mines preconditions from repeated guards at API call sites or infers broader program properties from executions [13, 14]. In contrast, our evidence comes from explicit checks in the implementation of the target method and its helpers. This choice limits recovery to explicit preconditions, but it ties every precondition report to a specific check in the source code and does not

Introduction

require a large usage corpus or a representative test suite.

We evaluated Prex on methods from 46 open-source Java projects and manually reviewed two independent samples of 150 precondition descriptors: one sampled from Prex and one sampled from WIT. Of the 150 Prex reports, 149 (99.3%) were correct. Among these correct reports, 125 were testable by users. This includes 12 cases in which the source check used hidden state but Prex recovered a public expression for the same state. In contrast, only 57 of the 150 sampled WIT reports were true and correctly-expressed preconditions. Most of the false positives were caused by exception paths that crossed the mutation boundary. Our results also show that precondition extraction is important to support developers, as we found that 122 of the 149 correct Prex reports identified preconditions that were not documented.

This thesis makes the following contributions:

- A novel static analysis algorithm for extracting preconditions from the computation of a mutation boundary and the detection of recognized patterns in source code;
- A technique to transform guard condition expressions to use method-boundary elements and to determine whether the resulting conditions are testable by client code;
- An empirical evaluation of the precision and testability of the output of the approach on 46 Java projects and a systematic comparison with a path expansion approach;
- Empirical data about the gap between implicitly checked and documented preconditions.
- An open source release of Prex to support reproducibility and use by general public¹.

¹The tool is released as an open source software on Codeburg [15].

Introduction

The rest of the thesis is organized as follows. In Chapter 2, we review research on exception behavior and precondition mining and explain how our work differs. Chapter 3 formulates the problem and describes the design and implementation of Prex. Chapter 4 presents our evaluation dataset, manual review process, and results. In Chapter 5, we discuss all the findings comprehensively. Finally, in Chapter 6 we conclude our research and the implications of our findings.

2

Related Work

Research on Java preconditions has followed several related directions. The direction closest to the present thesis studies the static extraction of exception conditions from method implementations. Another direction mines likely preconditions from recurring guard patterns at API call sites across large Java code corpora. There is also work on code inference based on contracts, annotations, and invariants. Across these directions, the main distinction from our work is that we focus on explicit precondition checks in method implementations, rather than on all exception-triggering conditions or broadly-inferred contracts. Section 2.1 distinguishes the scope of these conditions before the remainder of the chapter discusses the individual approaches.

2.1 Scope and Organization of Related Work

Precondition inference can use different sources of evidence. Behavioral interface specifications describe the intended behavior of program modules [4]. Inference techniques may instead use checks in the implementation, guards observed in client code, values observed during execution, or paths that lead to exceptions. The resulting condition has the meaning supported by its evidence: client-side mining reports potential API preconditions [13], dynamic inference reports likely invariants observed in executions [14], and exception analysis reports conditions under which a method can throw an exception [3].

This thesis studies explicit checks in method implementations. It focuses on checks made before the method begins changing program state. We also seek to describe each condition using method inputs and other information that a caller can access, where possible. The remainder of this chapter discusses implementation-based analyses, client-side precondition mining, and other forms of contract and invariant inference.

Table 2.1 provides a compact comparison of the directions discussed below. The table classifies each direction by the evidence used to infer a condition and by the kind of result it reports. The *Detailed discussion* column identifies the section that discusses each direction. The table does not compare their precision or usefulness, since the approaches have different goals and evaluation settings.

2.1 Scope and Organization of Related Work

Table 2.1: Evidence and reported conditions in related precondition and specification inference research directions.

Direction	Detailed discussion	Primary evidence	Reported condition
Exception behavior extraction [3, 16]	Section 2.2.1	Method implementation and exception paths	A condition under which a method can throw an exception.
Documentation and test-based condition recovery [17, 18]	Section 2.2.2	Documentation, test templates, or source analysis	Conditions based on documentation fragments or unit tests.
Client-side precondition mining [13, 19]	Section 2.3	Guards observed before API calls in client code	A potential API precondition inferred from recurring usage.
Object protocol analysis [20]	Section 2.4.1	Source-code locations that may indicate method-order constraints	Legal sequences of method calls and state transitions on an object.
Static contract and annotation inference [21, 22]	Section 2.4.2	Candidate annotations or assertions in source code	A contract clause or annotation for static checking.
Dynamic invariant inference [14]	Section 2.4.3	Values observed during program executions	A likely invariant that holds over observed executions.

2.2 Implementation-Based Analyses

Implementation-based analyses derive conditions from method implementations, exception paths, documentation, or test artifacts. The first subsection discusses analyses of exception behavior, and the second discusses approaches that recover conditions from documentation and test-related artifacts.

2.2.1 Exception Behavior Extraction

Several prior works extract exception conditions by reconstructing paths that lead to thrown exceptions. Marcilio and Furia presented WIT, a tool that extracts the exception preconditions of Java methods and constructors. WIT statically analyzes the paths in a method’s implementation that lead to an exception being thrown [3]. WIT is different in that it seeks to identify *exception preconditions*, i.e., an input configuration that triggers an exception. To do so, WIT builds paths that end in explicit throws or exceptional calls, inlines callees when their source code is available, and checks whether the resulting path condition is feasible. It uses bounded path exploration to extract the full path of the condition without building it. For this purpose, it relies on an encoding of paths limited to a core subset of Java features and library methods.

Marcilio and Furia evaluated WIT on the JDK and 46 Java projects. It reported 30,487 exception preconditions in 24,461 methods, and a manual analysis of a significant sample found that the reported preconditions were correct [3].

WIT reports conditions that can lead a method to throw an exception. In contrast, we report a

2.2 Implementation-Based Analyses

condition only when it acts as a precondition that users should satisfy before calling the method. For example, an exception may happen only after the method has already changed state, or the extracted formula may refer to internal state that users cannot directly test. We therefore use the mutation boundary and testability analysis to decide what and how to report as a precondition. In contrast to WIT, we aim to report only preconditions that users need to understand and validate before they call the method.

Buse et al. developed a technique to generate exception documentation based on interprocedural exception propagation analysis with symbolic execution-like path reasoning [16]. Their evaluation compared the generated documentation with more than 900 instances of existing exception documentation in almost two million lines of code. They found that the generated output was at least as good as the existing documentation 85% of the time and better 25% of the time [16]. The main connection to our research is that it statically extracts exception-related paths and turns them into documentation, again without the focus on producing precondition descriptors testable by client code.

Robillard and Murphy presented Jex, a static analysis tool that provides Java developers with the exception types that may arise at program points and the handlers that are present [23]. Overall, work on exception behavior extraction can be placed in the broader context of exception flow and error-handling behavior analysis [24, 25, 26]. This work investigates exception-triggering paths, propagation, and error-handling behavior in method implementations.

2.2.2 Documentation and Test-Based Condition Recovery

Exception conditions can be used as documentation or as input for creating executable artifacts. While Buse et al. use source analysis to produce documentation, related work on software testing starts from documented conditions. Goffi et al. presented Toradocu, which transforms Javadoc descriptions of exceptional behavior into test oracles using natural language processing and runtime instrumentation [17]. The resulting oracles can be combined with a test-input generation tool to create tests for exceptional behavior [17].

Nassif et al. presented DDescribe, a template-based technique that jointly generates unit tests and documentation fragments from one specification source [18]. The technique can detect and replace outdated documentation, since the test and documentation are generated from the same template [18]. These approaches show that the same behavioral fact can support documentation, testing, or both. They differ from this thesis because they use existing documentation or templates, whereas Prex extracts explicit precondition checks from method implementations.

Zhong et al. studied parameter constraints in Java libraries and where they appear, such as in Javadoc or implementation code [11]. Their PaRu approach locates constraints described in Javadoc and parameters associated with exception declarations or assertions, after which manual inspection categorizes the candidate rules and maps documentation rules to code rules [11]. The parameter rules are similar to preconditions, but may not have explicit guards in the code. This work studies existing descriptions and associated source locations, whereas Prex extracts explicit precondition checks from method implementations.

2.3 Precondition Mining from Client-Side Patterns

Another line of work extracts preconditions and constraints from recurring guard patterns at API call sites.

Ramanathan et al. statically infer preconditions that must hold when a procedure is called using interprocedural, path-sensitive dataflow analysis and predicate mining [2]. The basic idea of client-side mining is that if many call sites use the same guard before a call, that condition may reflect a precondition of the target API [13]. Nguyen et al. later use consensus over a large number of API usages to retain guard conditions that appear in the majority of usages and rank the inferred preconditions by frequency [19]. Later work observes that usage-based methods can produce stronger preconditions than intended, whereas implementation-based analysis can produce weaker ones, and argues for integrating the two approaches [27].

Ramanathan et al. evaluated their approach on large open-source benchmarks and reported that the derived specifications were precise and useful. Their implementation also discovered several critical, previously undocumented preconditions in well-tested libraries [2]. Nguyen et al. analyzed almost 120 million lines of code to infer preconditions for Java Development Kit APIs. They reported recall between 75-80% and precision between 82-84%, and found five preconditions missing from human-written specifications that a specification expert confirmed [13].

Client-side mining uses recurring usage as its evidence. In contrast, Prex recovers explicit checks from method implementations and can operate when client usage data is limited.

2.4 Contract and Invariant Inference

Other approaches infer properties that constrain the use of an API but are broader than one explicit check. They include object protocols, static contracts and annotations, and dynamic invariants.

2.4.1 Object Protocol Analysis

An object protocol describes legal sequences of calls on an object. It can model an object's concrete state as a finite set of abstract states and specify the method calls that move an instance between those states [20]. Consequently, the same method call can be valid in one abstract state and invalid in another. Beckman et al. developed ProtocolFinder, a tool that identifies source-code locations that may indicate such protocols [20]. In their study of several popular open-source Java programs, 7.2% of types were found to define protocols and 13% of classes acted as protocol clients [20]. ProtocolFinder is related to our work because it also uses source code patterns as evidence of usage constraints in Java programs. A precondition can depend on the same state, but it describes the requirement of one call rather than the sequence of calls that leads to that state. A report of an explicit precondition check can therefore help a client decide whether to make a particular call, without recovering the complete state-transition model of the object. Object protocols and method preconditions thus describe related usage constraints at different levels: protocols describe the history of interactions with an object, whereas this thesis extracts checks associated with individual method calls.

2.4 Contract and Invariant Inference

2.4.2 Static Contract and Annotation Inference

Flanagan et al. introduced Houdini, which infers candidate annotations with the modular static checker ESC/Java [21, 28]. Houdini generates a large set of candidate annotations, then iteratively runs ESC/Java to refute and remove candidates until a fixed point is reached. The authors report that, on four test programs totaling 50,000 lines of code, the inferred annotations reduced false alarms and the programmer time needed to find defects statically [21]. Even though Houdini is not focused on preconditions, it provides a scalable way to run template-like filters to infer annotations.

Cousot et al. formalize contract precondition inference from intermittent assertions and introduce abstract interpretation methods to infer both a static contract precondition and code that checks it at runtime [22]. Static contract and annotation inference therefore seeks properties that a checker can use to reason about a program without executing it. The inferred properties can express method requirements, but they are broader than the explicit method-entry checks considered in this thesis.

2.4.3 Dynamic Invariant Inference

Dynamic inference systems such as Daikon observe values produced during program executions and report likely invariants that hold at program points in the observed executions [14, 29]. These invariants can be used in assertions, documentation, and formal specifications [14]. Daikon supports C, C++, Java, and Perl programs, and its inferred invariants have been used for tasks including test-case generation, component-integration analysis, and theorem proving [14].

2.4 Contract and Invariant Inference

Nimmer and Ernst evaluated Houdini and Daikon as annotation assistants for ESC/Java, examining how static and dynamic inference help programmers complete verification tasks [30]. Dynamic inference can suggest useful properties without requiring an explicit check in the source code, but the reported properties depend on the executions available to the analysis. On the other hand, Prex uses source-code checks as its evidence and reports only conditions associated with explicit precondition checks before the mutation boundary.

3

Approach

We developed Prex, a source code analysis tool to detect preconditions from the source code of Java methods and produce guard expressions testable by client code.

3.1 Scope and Terminology

Generally speaking, a **precondition** is any predicate that must be true before a method is executed. In principle, executing a method without respecting all of its preconditions leads to undefined behavior. We refer to the execution of a method with one or more precondition violations as an **illegal execution**. In practice, the result of an illegal execution is often a runtime error indicated through Java's exception-handling mechanism. Other possible results include silently incorrect behavior and system crashes.

In the implementation of a method, a precondition can be either checked or unchecked. This

3.1 Scope and Terminology

distinction does not relate to the type of exception raised, but rather to whether there is code explicitly checking that the precondition holds. A precondition is checked if the method's implementation includes an explicit validation of input arguments or object state that aborts the method execution upon failure. In Java this check is typically implemented via a `throw` or `assert` statement. For example, the first statement of many of the implementations of `List#getFirst` checks whether the list is empty and explicitly throws an exception if it is. We refer to such code fragments as **precondition checks**. We define a precondition check as a predicate on the input to a method that occurs before the method has any side effect and that results in the execution of the method becoming *disabled*. Here, a side effect is a change to state outside the method's local variables[31, 32]. This includes updates to object fields. We call the **mutation boundary** the first reachable control-flow node on each path from the method entry point that is estimated to correspond to code with a side effect. For direct state mutations, this node is the first write to an instance or static field. A precondition check comprises two parts: a **guard**, which must hold for a valid call, and a violation **signal**, which is the type of exception raised when the guard does not hold. A precondition can also be unchecked in the sense that, although it is documented and/or its violation results in an exception being thrown, it is not possible to unambiguously identify a dedicated check for it.

The guards in precondition checks can be further categorized as **testable guards** or **untestable guards**. A testable guard can be evaluated by client code before calling the method. An untestable guard depends on information that is inaccessible outside the class.

Finally, preconditions can be documented with various degrees of quality. **Precise documentation** includes both a testable guard and signal. **Partial documentation** includes only fragments of

3.2 Problem Formulation

information or untestable code. In many cases, only a `@throws` clause is provided in the Javadocs. Finally, preconditions can, of course, remain **undocumented**.

Given the above terminology, we define the scope of our approach as a tool aimed at converting precondition checks into precise documentation, i.e., Javadocs that include testable guards and the violation signal.

3.2 Problem Formulation

We define the problem of precondition extraction as a function that takes as input a Java method and produces a set of **precondition descriptors**. We model a precondition descriptor p as a tuple (g, s, V) , where:

- g is the precondition guard, i.e., the boolean predicate that must be true to avoid invalidating the method's execution. In documentation, the guard is typically negated to describe the condition under which the violation signal occurs.
- s is the violation signal, expressed as the fully-qualified type of the exception (or error) thrown if the guard evaluates to false in an execution of the method. This element refers to a subtype of `Throwable`.
- V records how client code can obtain the values used in g before the call. A parameter is marked `DIRECT` if its value is directly accessible by the method caller, for example, through public fields or the method's input parameters. Other symbols map to equivalent public field accesses or public query expressions when available, and to `HIDDEN` when no accessible

3.3 Analysis Algorithm

equivalent is found.

Listing 3.1 shows the precondition descriptor extracted for method `ArrayList#add` of the Java Development Kit (JDK, version 25). The field `input` identifies the input method, and the `guard`, `signal`, and `map` correspond to elements g , s , V , respectively. The `map` shows that the public getter method `size()` has been identified for symbol `size` and that symbol `index` refers to a parameter. The field `description` combines the elements into a precise piece of documentation.

```
"input" : "ArrayList#add(index, E)",
"guard" : "!(index > size || index < 0)",
"signal" : "IndexOutOfBoundsException",
"map" : {
  "size" : "size()",
  "index" : "DIRECT"
},
"description" : "@Throws IndexOutOfBoundsException if index > size() || index < 0",
"testability" : "TESTABLE",
```

Listing 3.1: Example precondition output produced by Prex. Package names of Java elements are omitted for readability

3.3 Analysis Algorithm

Prex extracts precondition descriptors from a collection of Java source files. In principle, there is no requirement for the input files to logically correspond to a single project. However, the input collection is considered as a unit for the purpose of resolving dependencies, as described below.

3.3 Analysis Algorithm

Henceforth, we refer to the input collection of files simply as a *Java program*.

The analysis begins by parsing the program source code into a collection of abstract syntax tree (ASTs) using the JavaParser library.¹ Algorithm 1 presents the overall strategy to extract precondition descriptors from the input program. In addition to a Java program, the analysis takes as input a customizable catalog of common precondition-checking methods that are not part of the input Java program (e.g., `Objects#requireNonNull`).

Before processing individual methods, the `extractTypeHierarchy` function performs a preliminary pass over the ASTs of all compilation units to create a data structure that comprises all classes, their fields and visibility, and inheritance relations between classes (Line 1). This global context information is required to accurately resolve whether instance fields are accessible to client code.

The analysis then proceeds in a modular fashion by inspecting each method individually (Line 4). Prex first identifies potential precondition candidates in the method (Line 6). At this stage, a *precondition candidate* is either a recognized code pattern that results in an exception, or any method call (which could result in a downstream precondition check). The function `removeNonPreconditionFailures` filters these candidates using the estimated mutation boundary of each method and removes all non-precondition checks that lie beyond the boundary (Line 7).

For each candidate, we then inline any precondition check from downstream methods into the calling method’s context, ensuring that the precondition is attributed to the method that exposes it. As part of this propagation, the algorithm performs argument substitution so that the downstream method’s local conditions are accurately expressed in terms of the calling method’s scope. Once

¹<https://javaparser.org/>. We used version 3.27.1.

3.3 Analysis Algorithm

Algorithm 1: Overall Analysis Strategy

Input: J : Java program as ASTs
Input: L : Precondition catalog
Output: P : Set of all precondition descriptors for J

```
1  $T \leftarrow \text{extractTypeHierarchy}(J)$ ;  
2  $P \leftarrow \emptyset$ ;  
3 foreach method  $m \in J$  do  
4    $C \leftarrow \emptyset$ ;                                     // Potential Preconditions  
5    $P_m \leftarrow \emptyset$ ;  
6    $C \leftarrow \text{preconditionCandidates}(m, T, L)$ ;  
7    $C \leftarrow \text{removeNonPreconditionFailures}(C)$ ;  
8 end  
9 foreach candidate  $c \in C$  do  
10   $c' \leftarrow \text{propagateDownstreamChecks}(c, C, L)$ ;  
11  if  $\text{isPublic}(\text{declaringMethod}(c'))$  then  
12     $s \leftarrow \text{extractSignal}(c')$ ;  
13     $g \leftarrow \text{extractGuard}(c')$ ;  
14     $V \leftarrow \text{resolveVariableAccess}(c', T)$ ;  
15     $p \leftarrow (g, s, V)$ ;  
16     $P_m \leftarrow P_m \cup \{p\}$ ;  
17  end  
18 end  
19  $P \leftarrow P \cup P_m$ ;
```

downstream methods are inlined, the candidate is considered a complete precondition check (c' , Line 10).

In the final stage of the algorithm, we check if the precondition check's declaring method is public and, in the affirmative, we convert our internal representation of a precondition check into a precondition descriptor (Lines 12-15) and add it to the output (Lines 16 and 19).

3.4 Implementation Details

Prex relies on several dedicated strategies to transform AST nodes into precondition descriptors and classify their testability.

The precondition catalog (L) has two roles: identifying API methods that perform checks, such as `Objects#requireNonNull`, and identifying public state-querying methods used in guards, such as `List#size`. Every method listed in the catalog is assumed to have been manually verified as non-mutating. Prex includes selected methods from `java.util` and `java.lang` and allows users to extend the list with any method, including from project-specific code.

Recognized Patterns

In `preconditionCandidates`, we use three patterns to cover all explicit precondition checks that throw exceptions using `throw` or `assert` statements. A precondition is either always violated, or detected through a conditional check or an assertion statement. We convert the precondition checking process into three patterns: (1) unconditional throw, (2) assertion-based check, and (3) conditional throw. Prex is capable of handling variations in expressions of these general patterns, including early returns and preconditions raised in downstream method calls.

Unconditional Throw

This pattern corresponds to a `throw` statement in the direct path of the method entry that raises an exception whenever the method is executed. This structure is typically used in abstract classes, or

3.4 Implementation Details

for unsupported interface methods. For example, in Listing 3.2, the method `set` from `AbstractList` always throws an `UnsupportedOperationException`. This pattern corresponds to a precondition that can never be met.

```
public E set(int index, E element) {  
    throw new UnsupportedOperationException();  
}
```

Listing 3.2: Unconditional throw in `AbstractList`

Assert-based Checks

The second recognized pattern is the assertion statement. Each `assert` statement evaluates a boolean expression and throws an `AssertionError` if the condition is not satisfied (when assertion checking is enabled).

Conditional Throw

This more general form of precondition check uses an if-throw pattern. Listing 3.3 shows an abstract representation of this pattern. The `preGuardExecution()` and `postGuardExecution()` are respectively the code that executes before and after the precondition check. The expression in the if statement is the throw condition. `Prex` negates this condition when needed to produce the guard g required for a valid call. The mutation boundary of `preGuardExecution()` will be discussed in the next section.

```
methodName(inputArgs) {  
    preGuardExecution();  
    if (throwCondition) {
```

3.4 Implementation Details

```
        throw Exception e;
    }
    postGuardExecution();
}
```

Listing 3.3: Abstraction of conditional throw pattern

If the pre-guard execution does not cross the mutation boundary, we consider the if-throw block to represent a precondition. Prex then attempts to inline the elements used in the guard. The inlining logic aims to report the most accessible method or field that can be used to test the precondition. The depth of inlining is determined by the accessibility of the elements involved. For example, in Listing 3.4, `len` uses the value of `size()`. Since `size()` is a public method without state mutation, the tool replaces the access to `len` with a call to `size()` and negates the throw condition to recover the testable guard `size() != 0`. In contrast, in Listing 3.5, the throw condition contains a private method `computed` that is not accessible, so the tool inlines the helper methods logic to recover a testable guard.

```
public synchronized E peek() {
    int len = size();
    if (len == 0)
        throw new EmptyStackException();
    return elementAt(len - 1);
}
```

Listing 3.4: Conditional throw pattern precondition from class Stack

```
public void getResult(int i) {
```

3.4 Implementation Details

```
    if (computeJ(i) > 10) {  
        throw new IllegalArgumentException();  
    }  
}  
  
private int computeJ(int num) {  
    return num + 5;  
}
```

Listing 3.5: Input validation

Prex supports variations in expressions of the conditional. In Listing 3.6, the exception is placed in the else block. To extract the precondition, preconditionCandidates traverses the if-else branches upward from the throw statement, logically negating conditions to recover the required state. In this example, the extracted throw condition is

$$\neg(\text{initialCapacity} > 0) \ \&\& \ \text{initialCapacity} \neq 0$$

which the tool simplifies and negates to produce the guard $\text{initialCapacity} \geq 0$.

```
public ArrayList(int initialCapacity) {  
    if (initialCapacity > 0) {  
        this.elementData = new Object[initialCapacity];  
    } else if (initialCapacity == 0) {  
        this.elementData = EMPTY_ELEMENTDATA;  
    } else {  
        throw new IllegalArgumentException("Illegal Capacity: "+
```

3.4 Implementation Details

```
        initialCapacity);  
    }  
}
```

Listing 3.6: Cases where throw statement in the else block

The same traversal is applied to nested if statements and early returns. Starting from the throw statement, Prex walks through the control flow and collects the conditions that must hold to reach the violation signal. When a condition must be false to reach the signal, Prex negates the corresponding condition before adding it to the recovered guard.

Mutation Boundary

A precondition must be evaluated before a method performs any externally observable state mutation. We define the *mutation boundary* as the first reachable control-flow node on each path from the method entry point that is estimated to correspond to code with a side effect.

Due to aliasing, abstraction, and polymorphism, it is practically impossible to determine *precisely* whether a statement has a side-effect or not. To ensure that our approach remains conservative, our algorithm (`removeNonPreconditionFailures`) assumes that any statement has, by default, a side effect, and reverses this decision only if it can be verified that the statement has no side effect. To verify whether a statement is side-effect free, our analysis separately considers *mutating operators* and *method calls expressions*.

If a statement involves one or more mutating operators, such as assignment, increment, or decrement, Prex checks the target of the operation. Mutation operations applied to method-local

3.4 Implementation Details

variables are allowed, but all others are considered to be mutating. For non-primitive local references, the analysis also tracks name aliases introduced before the check, such as a local variable assigned from a field. If the alias is later used to update the referenced object, the update is also considered crossing the mutation boundary.

For statements involving method calls, Prex checks whether each call can be treated as non-mutating before the guard. Calls in the precondition catalog are considered non-mutating by definition, as they must be manually validated as such. Calls to methods defined outside the analyzed program and outside the precondition catalog are conservatively considered to be mutating. This approach also covers unresolved calls that may perform file or network I/O which the analysis does not separately model. For calls to methods within the analyzed program, we first determine whether they are statically bound or not (i.e., calls to static, final, or private methods). If a method is dynamically dispatched, we conservatively consider it to be mutating. If the method is statically dispatched, we recursively traverse its body to determine whether we can consider it non-mutating using the same procedure as described above. Finally, as a special case, if a call expression is to a method of a non-final class defined in the analyzed program *and* that has a single implementation, we analyze the single implementation instead of considering the method non-mutating, and issue a warning in the tool's output. During the development of the approach, we determined that excluding all dynamically dispatched calls was too conservative with respect to the mostly theoretical possibility that an external class could be injected by a third party.

3.5 Limitations and Exclusions

Resolving Variable Access

The `resolveVariableAccess` function evaluates the extracted guard to construct the variable mapping V and determine the precondition's testability. The function iterates through every variable and method in the guard. If a variable corresponds to a method input argument, it is mapped as `DIRECT`. Fields that are public or exposed via a getter method are mapped to their respective representations, while internal fields lacking public accessibility are flagged as `HIDDEN`. A method is considered a getter method for a field only if it is public, takes no parameters, does not cross the mutation boundary, and directly returns a field from the same class. For example, the guard for `ArrayList#getFirst` refers to the private field in `size != 0`. Mapping `size` to the getter method `size()` produces the testable guard `size() != 0`. The default implementation in `List#getFirst` already uses the public expression `!this.isEmpty()`, so its guard is directly testable.

3.5 Limitations and Exclusions

There are hypothetical cases of precondition checks that Prex intentionally excludes from analysis. Silent failures are excluded, as the method does not throw an exception when a precondition is violated, but instead silently fails or produces undefined behavior. It is difficult to statically detect silent failures as there is no clear indication of failure. We also ignore exceptions handled in catch blocks, since they represent internal error handling rather than checks before execution. We also exclude predicates evaluated in loops because of the difficulty of precisely identifying the exact conditions under which the precondition holds using only static analysis.

3.5 Limitations and Exclusions

Besides these intentional exclusions, the tool has a few known limitations. The `extractTypeHierarchy` function tracks arrays and collections as a whole, without tracking which element or nested object is mutated. This may make guards that depend on such state harder to recover. The static handling of downstream calls may miss checks that depend on dynamic dispatch or external code. Lastly, our detection of the mutation boundary can be imprecise when the method contains complex loops. To prevent false positives, we treat all loops before the precondition check as potentially mutating. Read-only loops may therefore cause valid precondition checks to be discarded.

4

Evaluation

We evaluated Prex through a benchmarking study and systematic comparison with a path-expansion approach. We analyzed precondition descriptors extracted from methods in 46 Java projects from the WIT dataset. The goal of our study was to provide a quantitative assessment of the estimated operational performance of pattern-based precondition extraction. We investigated the following questions:

- **RQ1 (Precision):** What proportion of preconditions reported by Prex are correct?
- **RQ2 (Testability):** What proportion of correct preconditions reported by Prex is testable by client code?
- **RQ3 (Comparison):** How does a pattern-based approach compare to a systematic path expansion approach?
- **RQ4 (Value):** How does pattern-based precondition extraction complement existing docu-

4.1 Sampling Approach and Dataset

mentation?

4.1 Sampling Approach and Dataset

We draw conclusions from the application of Prex to a sample of Java methods. We had three main requirements for our sampling approach. First, for the *ecological validity* of our results, methods in the sample should constitute examples of production code developers could reasonably expect to encounter in the wild. Second, to ensure the *internal validity* of our results, we aimed to sample to an extent that would enable the detailed manual review of each precondition in its code context. Finally, to ensure that our results were *comparable* to an alternative approach, we needed to draw from an existing public dataset. Given these requirements, we leveraged the dataset of Marcilio and Furia [3] and subsampled from it to reduce the scale to one that can be manually reviewed. We refer to this dataset as the *WIT dataset*, from the name of the corresponding tool.

The dataset comprises preconditions inferred from methods sampled from two sources: (1) classes from five modules of the Java Development Kit (JDK) version 11, and (2) classes from 46 open source Java projects (OSS). The paper reports two kinds of extracted exception preconditions: *expres*, whose feasibility is fully checked, and *maybes*, which are reported without the same guarantee. We used the 46 OSS project portion of the WIT dataset and excluded the JDK modules because we had used JDK classes as input data to develop the approach. The 46 OSS projects of the WIT dataset thus constitute unseen data for the purpose of our evaluation. Henceforth, references to the WIT dataset are to the OSS portion of the dataset. Excluding the JDK classes, we collected 53 222 precondition descriptors from the reports in the WIT replication package.

4.1 Sampling Approach and Dataset

Table 4.1: Evaluation Dataset

Approach	Output	Sample	Projects	Classes
Prex	18 412	150	31	145
WIT	53 222	150	31	144

To produce our sample, we applied Prex to all Java methods in the WIT dataset. This operation produced 18 412 precondition descriptors, or only about 35% of the amount yielded by WIT. From this list, we randomly sampled 150 precondition reports. These reports cover 31 projects, 145 classes, and 150 distinct method $\times$ exception entries. WIT reports a condition for 87 of these entries. We then randomly sampled another 150 precondition descriptors from WIT’s 53,222 results. These reports cover 31 projects, 144 classes, and 150 distinct method $\times$ exception entries. Table 4.1 summarizes our evaluation dataset. The full entries for both samples are available in Appendix A.

The two main parameters in this sampling approach are (1) the decision to sample per precondition descriptor as opposed to per method, and (2) the decision to sample 150 precondition descriptors per approach. The justification for our choice of precondition descriptor as the unit of analysis is simply that in operational conditions these are the data items developers would be inspecting. The justification for our selection of 150 items per approach is that we budgeted 25 hours for the focused manual review of results (300 items $\times$ an estimated average of five minutes per item).

The Appendix A contains the two manually reviewed 150-report samples, and the manual

4.2 RQ1: Precision of Prex

labeling conducted to map the two samples.

4.2 RQ1: Precision of Prex

We first investigated **RQ1: What proportion of preconditions reported by Prex are correct?**

Method We manually inspected each of the 150 preconditions reported by Prex in detail. We checked each reported precondition against the implementation from which it was extracted, tracing through downstream method calls when needed. We identified a precondition report as *correct* only if it satisfies the following requirements: (1) There is a path through the control flow of a method leading to a throw statement (called an *exception path*); (2) the exception path does not cross the mutation boundary; (3) the reported guard preserves the condition required by the source check, and (4) the variables and expressions in the reported guard refer to valid program entities: method inputs, instance variables accessible to the method, or expressions over them.

Results

In the Prex sample, 149 of the 150 reported conditions are correct. This observation corresponds to a sampled precision of 99.3%, with a Wilson binomial confidence interval of [96.3%, 99.9%] at the 95% level. The one incorrect report was due to a missing corner case in our implementation. The precondition is in a getter method, and Prex transforms the access to a private field into a call to the method containing the precondition, which creates a circular dependency.

4.3 RQ2: Testability

Answer to RQ1 (Precision): 149 of the 150 preconditions reported by Prex and randomly-sampled for inspection were correct. The single incorrect result was due to an implementation oversight, as opposed to an inherent limitation of the approach.

4.3 RQ2: Testability

Our answer to RQ1 validated that pattern-based precondition inference, as implemented through Prex, identifies preconditions with almost no false positives. The question is then, can we use these reports for software development? We thus sought to answer: **RQ2: What proportion of correct preconditions reported by Prex is testable by client code?**

Method As part of our systematic review of the results (see Section 4.2), we also manually assessed to what extent the reported precondition was *testable*. A tool condition is testable if every element (variable or method) needed to evaluate the reported precondition is accessible to client code outside the package. If a precondition check in source code uses hidden state, Prex attempts to recover a public query method for the same state as described in Section 3.3. When such a method is found, Prex can report a testable condition. If no expression visible to client code is available, the descriptor remains untestable. For convenience to developers and for the purpose of the research, Prex automatically reports whether the reported precondition descriptor is testable.

Results Among the 149 correct reports, 125 are testable by client code. In 113 cases, the source condition is already testable and Prex preserves that testability. In 12 cases, the condition relies on hidden state, but Prex recovers a public expression for the same state. The remaining 24 correct

4.4 RQ3: Comparison

reports depend on inaccessible state and remain untestable.

Answer to RQ2 (Testability): Among the 149 correct reports, 125 are testable by users. Prex preserves testability in 113 reports whose source condition is already testable and recovers a testable form in 12 reports whose source condition is inaccessible.

4.4 RQ3: Comparison

We manually reviewed the 150 reports in the WIT sample and compared the results with those from the Prex sample. The analysis answers **RQ3: How does a pattern-based approach compare to a systematic path expansion approach?**

Method The samples were drawn independently from the output of each tool. We used the same systematic review process as described in Sections 4.2 and 4.3, including the same definitions of *correctness* and *testability*. WIT targets the broader concept of exception behavior, so a report classified as incorrect under our criteria may still correctly describe an exception-triggering condition. The comparison therefore measures the quality of reports that a user may encounter in each output.

Results In the WIT sample, only 57 of the 150 precondition descriptors are correct according to our precondition definition. Most incorrect descriptors are due to mutation boundary violations: the exception-raising statement can be reached after the method has crossed a conservatively-estimated mutation boundary, so the condition is arguably not a clear precondition. Table 4.2 presents the two independent samples side by side.

4.4 RQ3: Comparison

Table 4.2: Precision comparison on independent samples from each tool.

Sample	Correct	Incorrect	Precision	95% CI
Prex	149	1	99.3%	[96.3, 99.9]%
WIT	57	93	38.0%	[30.6, 46.0]%

The comparison with WIT clearly shows the impact of considering the mutation boundary when extracting preconditions. Among the 93 WIT reports that do not satisfy our precondition definition, 69 describe exception paths that cross the mutation boundary. Prex does not report a condition for any of these 69 entries. The mutation boundary therefore removes many exception paths that are valid descriptions of exceptional behavior but are not method-entry preconditions under our definition.

The difference is also visible in the form of the reported guard condition. WIT may inline the code of downstream methods and expose an internal value even when the source condition is already expressed through a public getter. For example, in `JavaBeanDescriptor#getPrimitiveProperty()`, the source check uses `isPrimitiveType()`, which is a public method of the same class. Prex reports the condition as `!(isPrimitiveType())`. WIT reports `type != 6`, exposing the internal encoding used by the downstream method instead of the public method accessible to client code. Another case where WIT fails relates to inlining the behavior of lower level library methods. For example, in `BitmapCharacterSet#setStyle(int)`, the source guard checks the size of the receiver’s characters collection. WIT reports the condition as `size > 1`, which drops the receiver object expression and retains an unbound variable size. In contrast, Prex reports `characters.size() > 1`, preserving the value used

4.4 RQ3: Comparison

by the source.

In terms of overlap, Prex reports a condition for 40 of the 57 correct entries in the WIT sample, and all 40 Prex reports are correct. The 17 correct WIT reports not covered by Prex (i.e., the false negatives) are all explained by our conservative design decisions to (1) only analyze the input program (excluding dependencies), and (2) assume methods have side effects by default. Specifically, some precondition candidates were eliminated because they call methods outside the analyzed program and outside the precondition catalog. In the other cases, dynamically-bound methods were conservatively discarded but the manual inspection shows that potential bindings did not result in state mutation. In the Prex sample, WIT reports a condition for 87 entries. All 87 Prex reports are correct, while 82 of the corresponding WIT reports are also correct under our definition. The remaining five WIT reports do not preserve a correct precondition due to variables that are not in scope at the method entry point.

Answer to RQ3 (Comparison): A total of 93 out of 150 (62%) exception paths reported by the WIT path-expansion approach do not describe likely preconditions because they occur after the method may have mutated the state of the running program and/or involve symbols that are not in scope at the method entry point. Comparison with WIT identified 17 false negatives out of $(150 + 17 = 167)$ precondition reports, all of which are due to conservative assumptions in the design of Prex.

4.5 RQ4: Value

Finally, we sought to better understand the value proposition of our proposed approach. We thus investigated **RQ4: How does pattern-based precondition extraction complement existing documentation?**

Method To evaluate how Prex complements documentation, we examined whether the preconditions reported by Prex are documented and how much source code analysis is needed to recover them manually. For each method in the Prex sample, the first author reviewed its documentation and implementation. We labeled documentation based on the relevant `@throws` entry and its description. We separately labeled recovery effort based on the control flow and downstream calls that must be followed to recover the guard from the implementation. Table 4.3 defines both dimensions. `NONE` corresponds to undocumented preconditions, `UNEXPLAINED` and `UNSTRUCTURED` to fragmentary or unstructured documentation, and `PRECISE` to precise documentation. The quantitative results include only the 149 correct Prex reports.

Results Table 4.4 reports documentation quality and source recovery effort for the 149 correct Prex reports. These labels measure orthogonal dimensions. A precondition can be undocumented but easy to recover if it appears as a direct if statement followed by throw, or a downstream validation method. A documented exception can still be hard to understand if the source condition is spread across downstream method calls or control flow.

The documentation categorization shows evidence of poor documentation quality. In 122 of the 149 correct Prex reports, the relevant exception or condition is not even documented. In another 17

4.5 RQ4: Value

Table 4.3: Coding Guide for Precondition Documentation

Code	Description	Example
<i>Dimension: Documentation (ordinal scale)</i>		
NONE	The relevant exception or precondition is not documented.	No relevant <code>@throws</code> entry.
UNEXPLAINED	The exception is documented, but the condition is not explained.	<code>@throws IllegalArgumentException</code>
UNSTRUCTURED	The condition is described in natural language without identifying the exact program values and relations.	<code>@throws IllegalArgumentException</code> if the list is empty.
PRECISE	The documentation identifies the relevant program value and the relation that causes the exception.	<code>@throws IllegalArgumentException</code> if index < 0.
<i>Dimension: Recovery Effort (ordinal scale)</i>		
DIRECT	The guard appears directly in the method or in a dedicated validation call.	<code>assert input != null;</code>
INTERMEDIATE	Recovering the guard requires a simple substitution, calculation, or helper lookup.	<code>int i = input * 2; assert i > 5;</code>
COMPLEX	Recovering the guard requires combining branches or following nested helper logic.	<code>if (conditionA)...else if (conditionB)...else {throw Exception}</code>

cases, there is only partial documentation: six cases are UNEXPLAINED, and 11 cases are UNSTRUCTURED. Only 10 preconditions have corresponding documentation that names the relevant state and

4.5 RQ4: Value

Table 4.4: Documentation quality and source recovery effort among correct Prex reports.

Documentation	NONE	UNEXPLAINED	UNSTRUCTURED	PRECISE
Reports	122	6	11	10

Recovery effort	DIRECT	INTERMEDIATE	COMPLEX
Reports	108	21	20

the triggering relation. Overall, 139 of the 149 correct reports are missing precise documentation.

The categorization of manual recovery effort shows that many preconditions can be retrieved directly via inspection, but not all of them are easy for users to recover. The 108 `DIRECT` cases include direct null checks, simple input checks, assertions, and validation helper methods. The 21 `INTERMEDIATE` cases require simple code inlining, substitution, or local calculations. The 20 `COMPLEX` cases require significantly more effort to extract. Some guards appear only after reconstructing nested `if/else-if/else` branches, where the exception-throwing branch is reached only after combining earlier branch conditions. Other guards are hidden inside helper methods that also perform setup, input normalization, or queries of the object state. A method call may look like normal setup code, but the callee may perform an additional precondition check before returning. In such cases, a user has to follow the call, mentally inline the relevant guard, and convert it back into input from the client code.

4.5 RQ4: Value

Answer to RQ4 (Value) Our sample provides evidence of 122 cases of undocumented and 17 cases of partially-documented preconditions out of 150 inspected. Manually recovering the precondition can require challenging code inspection and reasoning in at least 20 cases.

5

Discussion

This chapter discusses the implications of the four research questions, the tradeoffs of the evaluation design, and the directions for future work. Each research question is restated with its main result before considering its meaning for precondition inference and use.

5.1 RQ1: Precision of Prex Precondition Reports

RQ1 asks whether a precondition report from Prex accurately represents an explicit check in the source code. We reviewed 150 sampled reports by tracing each condition to the relevant exception path and checking that it occurs before the method changes program state. Of these reports, 149 were correct. The remaining report was caused by an implementation self-reference in the prototype.

The answer to RQ1 validates our assumption that pattern-based precondition extraction can

5.2 RQ2: Client Testability of Extracted Preconditions

produce precise output by targeting code patterns that align with developer intent to check preconditions in method implementations. However, the inherent trade-off for a high-precision solution is the potential presence of false negatives (i.e., missed preconditions). We do not measure recall as part of RQ1. Recall would measure the proportion of all explicit preconditions in the studied methods that Prex reports, which requires a ground truth of all such preconditions. Nevertheless, comparison with a path expansion approach (see Section 4.4) confirmed that Prex did not miss any case identified through this alternative.

5.2 RQ2: Client Testability of Extracted Preconditions

RQ2 asks what proportion of correct preconditions reported by Prex can be tested by client code. A condition is testable when client code can evaluate it before calling the method. Among the 149 correct reports, 125 are testable by client code. Among the testable preconditions, 113 were already expressed using accessible values and 12 became testable after Prex recovered a public expression for hidden state.

Correctness and testability describe different properties of a report. A condition can preserve the required condition from the source check but still depend on state that users cannot observe before calling the method. Most correct Prex reports preserve a condition that is already expressed using inputs or public state. The 12 recovered cases show the additional value of variable mapping: the source check uses hidden state, but a public query method provides an equivalent expression that users can test. In the remaining 24 cases, no testable mapping was found, suggesting that these preconditions cannot be easily leveraged for design by contract. This observation motivates im-

5.3 RQ3: Comparison with WIT's Path-Expansion Analysis

provements in development practices, whereby precondition checks are expressed in terms of code that can be evaluated outside the method. By automatically identifying the testability of precondition checks, Prex supports efforts to improve code to make it more suitable for design by contract practices.

5.3 RQ3: Comparison with WIT's Path-Expansion Analysis

For RQ3, we compare Prex with WIT, which uses path expansion to report conditions under which exceptions can occur. We manually reviewed independent random samples of 150 reports from each tool using the same definition of a method precondition. Under this definition, 149 of the 150 Prex reports and 57 of the 150 WIT reports met the precondition criteria and are considered correct.

WIT reports feasible conditions that trigger exceptions, including conditions reached after the method has started executing. Prex reports explicit checks that act as obligations on the user before execution. The difference in output size follows from the different focus of each approach. First, the mutation boundary removes exception paths that occur after state changes, including the 69 cases identified in the WIT sample. Second, Prex follows checks within the analyzed program and methods listed in its precondition catalog, but does not expand exception behavior from other Java base-library methods. WIT follows such lower-level behavior when constructing exception paths. The smaller output therefore reflects a more targeted analysis scope of Prex, rather than a direct measure of recall. The answer to RQ3 thus illustrates how pattern-based precondition extraction is a solution that offers near-perfect precision at the cost of some false negatives in the presence

5.4 RQ4: Complementing Precondition Documentation

of precondition checks with complex logic or delegation to dynamically-bound and/or externally-implemented methods.

5.4 RQ4: Complementing Precondition Documentation

Among the 149 correct reports, 122 of the preconditions are not documented and 17 are partially documented.

RQ4 asks how pattern-based precondition extraction complements existing documentation. We compared each of the 149 correct reports with the relevant method documentation and recorded the effort needed to recover the condition manually from the source code. 122 of these reports were undocumented, 17 were only partially documented, and 10 were documented precisely. In 20 of the 149 cases, the preconditions require complex source inspection if users attempt to recover them manually.

The documentation gap is not limited to preconditions that are difficult to express. Many direct checks are also undocumented, even though their conditions are already present in a form close to what could be shown to users. This suggests that precondition checks and method documentation may be maintained as separate concerns. When checks are placed in helper methods or spread across the control flow, keeping the documentation complete becomes challenging. Prex can help connect documentation and code relating to precondition checks. Its descriptors bind the violation signal to the extracted guard and its source check, and express the condition using values available at the method boundary when possible. For direct checks, this information can be collected without requiring users to inspect the implementation. For checks obscured by local calculations,

5.5 Tradeoffs and Threats to Validity

branches, or helper methods, the analysis performs the substitutions needed to recover the condition. These descriptors can therefore provide a basis for completing or checking documentation when the source and its documentation have become inconsistent.

5.5 Tradeoffs and Threats to Validity

The evaluation combined automated analysis of 46 open-source Java projects with detailed manual review of two independent samples of 150 reports. This design provides direct evidence about correctness, testability, documentation quality, and the comparison with WIT, but it also introduces limits related to the sample scope and the manual classifications.

5.5.1 Sampling Scope

Our experimental design is centered around a detailed expert inspection of a small sample of results as opposed to a quantitative analysis of large amounts of unvalidated data. This trade-off provides us high confidence that the data items we report on are meaningful, but at the cost that we have not verified how the approach operates on all possible input scenarios in our sampling frame of 46 open-source projects.

5.5.2 Manual Classification

One potential threat to the internal validity comes from the manual labeling required by various components of the evaluation, including correctness, testability, documentation, and recovery effort associated with precondition descriptors. We design our classification schemes to minimize

5.6 Future Work

subjective judgement, but there remains the possibility of human error in any manual inspection process. However, given that each precondition report in the set of 300 had to be reviewed multiple times under different perspectives as part of the research, we estimate the risk of error to be very low and we include our raw data for independent verification. We did not conduct an inter-rater agreement evaluation on the classification because of the high level of effort and expertise involved. In our case, an inter-rater agreement metric would be more indicative of reviewer motivation and focus during the task than inherent subjectivity in the classification scheme.

5.6 Future Work

Prex reports explicit checks that it can trace in source code before a possible state change. This design supports high precision, but it can exclude checks that depend on loops, collections, or calls with unresolved side effects. The evaluation also showed that the resulting descriptors can expose undocumented requirements. We may help developers add this missing information to method documentation in the future.

5.6.1 Increasing Analysis Coverage

Future work can improve the analysis of collections, loops, and dynamic dispatch to improve recall while preserving the precision of the reported conditions. Extending the default precondition catalog with common library methods and identifying side-effect-free query methods could recover some of these cases. The analysis could also extend support to collections and loops, where a precondition may be evaluated for each element rather than through a single guard at the method

5.6 Future Work

entry point. For dynamically-bound calls, a future direction is to summarize all known implementations and determine when a call does not cross the mutation boundary. These extensions should be evaluated against the current precision criterion, since increasing coverage must not turn general exception paths into reported preconditions.

5.6.2 Using Extracted Preconditions in Development

Another direction is to use Prex to migrate precondition checks into assert statements, or to inject checks in client code, thus helping developers validate preconditions before a method call. The descriptor produced by Prex already connects a guard, a violation signal, and the source location of the check. For a testable precondition, these elements could support a candidate `@throws` entry, a client-side assertion, or a unit-test oracle. For an untestable precondition, the report could instead direct developers to the hidden state that prevents client validation. A study of this workflow could examine whether the reports help developers document preconditions, find inconsistencies between code and documentation, or revise interfaces whose preconditions cannot be tested by clients.

6

Conclusion

We presented Prex, a static analysis approach for extracting explicit precondition checks from recognized patterns in Java source code. Each precondition descriptor returned by Prex records the guard required for a valid call and the violation signal. To the extent possible, the guard expression is transformed into an expression that can be called by client code to check whether the precondition holds. A key aspect of our approach is the computation of a mutation boundary within methods. With this mutation boundary identified, we limit the analysis to checks that occur before the method changes program state. This important functionality allows us to distinguish likely preconditions from general exception-causing paths through the code, reported by previous approaches.

Our evaluation shows that pattern-based extraction can recover explicit preconditions with high precision. A total of 149 of the 150 sampled reports from our current prototype of Prex were cor-

Conclusion

rect. For several preconditions originally expressed in terms of hidden state, our approach was able to recover the equivalent publicly-testable checks. The remaining untestable conditions revealed contract design issues in which client code must satisfy a precondition without access to the relevant state information. The comparison with an existing path expansion approach also confirmed that our novel approach produces higher precision and more useful results, at the cost of some potential false negatives caused by a conservative approach to estimating state mutation. Finally, inspection of the documentation of sampled methods showed that the preconditions we extract were seldomly documented at the same level of detail as the code checks. Manually recovering these preconditions is nontrivial, especially when the recovery requires inspecting nested method calls.

Prex shows that explicit precondition checks contain contract information that is often missing from documentation or hard to understand. Its output can support documentation and can also identify untestable preconditions that may require changes to the method implementation.

Bibliography

- [1] B. Meyer, “Applying ’design by contract’,” *Computer*, vol. 25, no. 10, pp. 40–51, 1992.
- [2] M. K. Ramanathan, A. Grama, and S. Jagannathan, “Static specification inference using predicate mining,” *SIGPLAN Not.*, vol. 42, no. 6, p. 123–134, Jun. 2007. [Online]. Available: <https://doi.org/10.1145/1273442.1250749>
- [3] D. Marcilio and C. A. Furia, “Lightweight precise automatic extraction of exception preconditions in Java methods,” *Empirical Software Engineering*, vol. 29, no. 1, p. 30, Dec. 2023.
- [4] J. Hatcliff, G. T. Leavens, K. R. M. Leino, P. Müller, and M. Parkinson, “Behavioral interface specification languages,” *ACM Comput. Surv.*, vol. 44, no. 3, Jun. 2012. [Online]. Available: <https://doi.org/10.1145/2187671.2187678>
- [5] M. P. Robillard, *Introduction to Software Design with Java*, 3rd ed. Springer Cham, 2026.
- [6] W. Mostowski and M. Ulbrich, “Dynamic dispatch for method contracts through abstract predicates,” in *Proceedings of the 14th International Conference on Modularity*, ser. MODULARITY 2015. New York, NY, USA: Association for Computing Machinery, 2015, p. 109–116. [Online]. Available: <https://doi.org/10.1145/2724525.2724574>

BIBLIOGRAPHY

- [7] L. Burdy, Y. Cheon, D. Cok, M. D. Ernst, J. Kiniry, G. T. Leavens, K. Rustan, M. Leino, and E. Poll, “An overview of jml tools and applications1 1www.jmlspecs.org,” *Electronic Notes in Theoretical Computer Science*, vol. 80, pp. 75–91, 2003, eighth International Workshop on Formal Methods for Industrial Critical Systems (FMICS’03). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1571066104808107>
- [8] M. Monperrus, M. Eichberg, E. Tekes, and M. Mezini, “What should developers be aware of? an empirical study on the directives of API documentation,” *Empirical Software Engineering*, vol. 17, no. 6, p. 703–737, Dec. 2012.
- [9] S. H. Tan, D. Marinov, L. Tan, and G. T. Leavens, “@tcomment: Testing javadoc comments to detect comment-code inconsistencies,” in *Proceedings of the 5th IEEE International Conference on Software Testing, Verification and Validation*, 2012, pp. 260–269.
- [10] N. Polikarpova, I. Ciupa, and B. Meyer, “A comparative study of programmer-written and automatically inferred contracts,” in *Proceedings of the 18th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2009, p. 93–104. [Online]. Available: <https://doi.org/10.1145/1572272.1572284>
- [11] H. Zhong, N. Meng, Z. Li, and L. Jia, “An empirical study on API parameter rules,” in *Proceedings of the 42nd ACM/IEEE International Conference on Software Engineering*, ser. ICSE ’20, Oct. 2020, p. 899–911. [Online]. Available: <https://dl.acm.org/doi/10.1145/3377811.3380922>

BIBLIOGRAPHY

- [12] P. Cousot and R. Cousot, “Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints,” in *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, ser. POPL '77. New York, NY, USA: Association for Computing Machinery, 1977, p. 238–252. [Online]. Available: <https://doi.org/10.1145/512950.512973>
- [13] H. A. Nguyen, R. Dyer, T. N. Nguyen, and H. Rajan, “Mining preconditions of APIs in large-scale code corpus,” in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, Nov. 2014, p. 166–177. [Online]. Available: <https://dl.acm.org/doi/10.1145/2635868.2635924>
- [14] M. D. Ernst, J. H. Perkins, P. J. Guo, S. McCamant, C. Pacheco, M. S. Tschantz, and C. Xiao, “The Daikon system for dynamic detection of likely invariants,” *Science of Computer Programming*, vol. 69, no. 1–3, p. 35–45, Dec. 2007.
- [15] S. Zhou and M. P. Robillard, “Prex release 0.1,” 2026. [Online]. Available: <https://codeberg.org/mpr/Prex>
- [16] R. P. Buse and W. R. Weimer, “Automatic documentation inference for exceptions,” in *Proceedings of the 17th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2008, p. 273–282. [Online]. Available: <https://doi.org/10.1145/1390630.1390664>
- [17] A. Goffi, A. Gorla, M. D. Ernst, and M. Pezzè, “Automatic generation of oracles for exceptional behaviors,” in *Proceedings of the 25th SIGSOFT International Symposium*

BIBLIOGRAPHY

- on Software Testing and Analysis*, Jul. 2016, p. 213–224. [Online]. Available: <https://dl.acm.org/doi/10.1145/2931037.2931061>
- [18] M. Nassif, A. Hernandez, A. Sridharan, and M. P. Robillard, “Generating unit tests for documentation,” *IEEE Transactions on Software Engineering*, vol. 48, no. 9, pp. 3268–3279, 2022.
- [19] H. A. Nguyen, R. Dyer, T. N. Nguyen, and H. Rajan, “Consensus-based mining of API preconditions in big code,” in *Companion Proceedings of the 2015 ACM SIGPLAN International Conference on Systems, Programming, Languages and Applications: Software for Humanity*, ser. SPLASH Companion 2015, Oct. 2015, p. 5–6. [Online]. Available: <https://dl.acm.org/doi/10.1145/2814189.2816271>
- [20] N. E. Beckman, D. Kim, and J. Aldrich, “An empirical study of object protocols in the wild,” in *Proceedings of the 25th SIGPLAN SIGSOFT European Conference on Object-Oriented Programming*, 2011, pp. 2–26.
- [21] C. Flanagan and K. R. M. Leino, “Houdini, an annotation assistant for ESC/Java,” in *Proceedings of the 10th International Symposium of Formal Methods for Increasing Software Productivity*, 2001, p. 500–517.
- [22] P. Cousot, R. Cousot, and F. Logozzo, “Precondition inference from intermittent assertions and application to contracts on collections,” in *Proceedings of the 12th ACM International Conference on Verification, Model Checking, and Abstract Interpretation*, 2011, pp. 150–168.

BIBLIOGRAPHY

- [23] M. P. Robillard and G. C. Murphy, “Static analysis to support the evolution of exception structure in object-oriented systems,” *ACM Transactions on Software Engineering and Methodology*, vol. 12, no. 2, p. 191–221, Apr. 2003. [Online]. Available: <https://doi.org/10.1145/941566.941569>
- [24] M. Acharya and T. Xie, “Mining API error-handling specifications from source code,” in *Proceedings of the 12th International Conference on Fundamental Approaches to Software Engineering*, 2009, p. 370–384.
- [25] W. Weimer and G. C. Necula, “Mining temporal specifications for error detection,” in *Proceedings of the 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2005, p. 461–476.
- [26] M. P. Robillard and G. C. Murphy, “Analyzing exception flow in Java™ programs,” in *Proceedings of the 7th European Software Engineering Conference and 7th ACM SIGSOFT Symposium on the Foundations of Software Engineering*, 1999, pp. 322–337.
- [27] H. A. Nguyen, T. N. Nguyen, H. Rajan, and R. Dyer, “Towards combining usage mining and implementation analysis to infer API preconditions,” in *Proceedings of the 1st ACM SIGSOFT International Workshop on Automated Specification Inference*, 2018, p. 15–16. [Online]. Available: <https://doi.org/10.1145/3278177.3278182>
- [28] C. Flanagan, K. R. M. Leino, M. Lillibridge, G. Nelson, J. B. Saxe, and R. Stata, “Extended static checking for Java,” *ACM SIGPLAN Notices*, vol. 37, no. 5, p. 234–245, May 2002. [Online]. Available: <https://doi.org/10.1145/543552.512558>

- [29] M. D. Ernst, J. Cockrell, W. G. Griswold, and D. Notkin, “Dynamically discovering likely program invariants to support program evolution,” in *Proceedings of the 21st IEEE/ACM International Conference on Software Engineering*, 1999, p. 213–224. [Online]. Available: <https://doi.org/10.1145/302405.302467>
- [30] J. W. Nimmer and M. D. Ernst, “Invariant inference for static checking: an empirical evaluation,” *SIGSOFT Softw. Eng. Notes*, vol. 27, no. 6, p. 11–20, Nov. 2002.
- [31] A. Sălcianu and M. Rinard, “Purity and side effect analysis for java programs,” in *Verification, Model Checking, and Abstract Interpretation*. Springer Berlin Heidelberg, 2005, pp. 199–215.
- [32] D. Helm, F. Kübler, M. Eichberg, M. Reif, and M. Mezini, “A unified lattice model and framework for purity analyses,” in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, ser. ASE ’18. Association for Computing Machinery, 2018, p. 340–350. [Online]. Available: <https://doi.org/10.1145/3238147.3238226>



Evaluation Samples

This appendix presents the two 150-report evaluation samples used in Chapter 4.

A.1 Prex Evaluation Sample

Table A.1: Prex evaluation sample (150 reports): Reported preconditions.

id	project	class	method	exception	wit Condition	mnier Condition
jackson-databind-classutil-throwasmappingexception-4bec3504	jackson-databind	com.fasterxml.jackson.databind.util.ClassUtil	throwAsMappingException(DeserializationContext, IOException)	JsonMappingException.from(ctxt, eo.get		!(eo instanceof JsonMappingException)
jackrabbit-restore-accept-9f7349be	jackrabbit	org.apache.jackrabbit.jcr2spi.operation	accept(OperationVisitor)	AssertionErrorCause(eo)		status != STATUS_PENDING && status != STATUS_PENDING
accumulo-dynamicbloomfilter-or-9abc534b	accumulo	org.apache.accumulo.core.bloomfilter	or(Final Filter)	IllegalArgumentExceptionException		dbf.matrix.length != this.matrix.length dbf.nr != this.nr && !(filter == null !filter instanceof DynamicBloomFilter filter.vectorSize != this.vectorSize filter.nbHash != this.nbHash)
asm-sourceinterpreter-sourceinterpreter-9b9du923	asm	org.objectweb.asm.tree.analysis.SourceInterpreter	SourceInterpreter()	IllegalStateExceptionException		getClass() != SourceInterpreter.class
jmonkeyengine-terrainquad-supportmultiplecollisions-c9162c23	jmonkeyengine-terrainquad-supportmultiplecollisions-c9162c23	jmonkeyengine.com.jme3.terrain.geomipmap.TerrainQuad	setSupportMultipleCollisions(boolean)	IllegalStateExceptionException		!(picker instanceof BresenhamTerrainPicker) && picker != null
pdfbox-data-type-setvalue-fbfc4c61	pdfbox	org.apache.pdfbox.type.datatype	setValue(Object)	IllegalArgumentExceptionException		value == null && !isGoodType(value)
lucene-solr-bytesequencereoutputs-subtract-96b2ef75	lucene-solr	org.apache.lucene.util.fst.ByteSequenceOutputs	subtract(BytesRef, BytesRef)	AssertionError		output == null && output == null
lucene-solr-builder-build-50b55036	lucene-solr	org.apache.solr.client.solrj.impl.CloudHttp2SolrClient.Builder	build()	IllegalArgumentExceptionException		this.solrUri.isEmpty() && zkHosts.isEmpty() && stateProvider == null
lucene-solr-indexandtaxonomyrevision-open-05d8b722	lucene-solr	org.apache.lucene.replicator.IndexAndTaxonomyRevision	open(String, String)	AssertionError		!(source.equals(INDEX_SOURCE) source.equals(TAXONOMY_SOURCE)) && !(source.equals(INDEX_SOURCE) source.equals(TAXONOMY_SOURCE))
joda-time-localdate-get-05ba4ceb	joda-time	org.joda.time.LocalDate	get(DateTimeFieldType)	IllegalArgumentExceptionException		isSupported(fieldType) == false && fieldType != null
wicket-subtree-subtree-a768cd4e	wicket	org.apache.wicket.extensions.markup.html.repeater.tree.nested.Subtree	Subtree(String, final NestedTree<T>, final IModel<T>)	IllegalArgumentExceptionException		tree == null

id	project	class	method	exception	wit Condition	miner Condition
lucene-solr-knn-evaluator-dowork-ebcdfc2	lucene-solr	org.apache.solr.client.solrj.io.eval.KnnEvaluator	doWork(Object...)	IOException		(values[1] instanceof List) && !(values.length < 3)
camel-jmx-endpoint-createconsumer-a865089	camel	org.apache.camel.component.jmx.JMXEndpoint	createConsumer(Processor)	IllegalArgumentException Exception		isNotifyHigh() && !isNotifyLow() && getMonitorType().equals("gauge") && !getMonitorType().equals("string") && !ObjectHelper.isEmpty(getObservedAttribute()) && isPlatformServer() & getMonitorType() != null
camel-defaulthealthcheckregistry-register-357262f4	camel	org.apache.camel.impl.health.DefaultHealthCheckRegistry	register(Object)	IllegalArgumentException Exception		obj instanceof HealthCheck obj instanceof HealthCheckRepository
spoon-smpljavadsl-getwrappedelement-4241f2cf	spoon	spoon.smpl.SmplJava	getWrappedElement(CtElement)	IllegalArgumentException Exception		!(wrapper instanceof CtInvocation)
flink-hiveparserbasesemanticanalyzer-charsetstring-453c6f33	flink	org.apache.flink.table.planner.delegation.hive.copy.HiveParserBaseSemanticAnalyzer	charSetString(String)	AssertionError		charSetString.charAt(0) != '0' && charSetString.charAt(0) != '\n' & charSetString.charAt(0) != '0'
swingx-xtaskpaneproperty-swingx-getsessionstate-5746675c	swingx	org.jdesktop.swingx.appframework.XProperties.XTaskPaneProperty	getSessionState(Component)	IllegalArgumentException Exception		tc instanceof JXTable && c != null
flink-nestedrowdata-setlong-f1c14851	flink	org.apache.flink.table.data.binary.NestedRowData	setLong(int, long)	AssertionError		!(pos >= 0) && !(pos >= 0)
jmonkeyengine-skeletonloader-endelement-e4eb2e1d	jmonkeyengine	jmonkeyengine.com.jme3.scene.plugins.ogre.SkeletonLoader	endElement(String, String)	AssertionError		position == null && qName.equals("keyframe") && !qName.equals("tracks") && !qName.equals("track") && !qName.equals("animation") && !qName.equals("bonehierarchy") && !qName.equals("bone") && !qName.equals("rotate") qName.equals("rotation") && !qName.equals("axis") && !qName.equals("translate") qName.equals("position") qName.equals("scale") && position == null
commons-lang-failablelongunaryoperator-andthen-f5e77176	commons-lang	lang3.function.FailableLongUnaryOperator	andThen(FailableUnaryOperator<E>)	AssertionError		Objects.requireNonNull(after)
lucene-solr-geo3ddistancecalculator-area-1cd4dd22b	lucene-solr	org.apache.lucene.spatial.spatial4j.Geo3dDistanceCalculator	area(Rectangle)	UnsupportedOperation Exception		TRUE

id	project	class	method	exception	wit Condition	miner Condition
wicket-abstractpageableview- setitemsperpage-c7722829	wicket	org.apache.wicket. markup.repeater. AbstractPageableView	setItemsPerPage(long)	IllegalArgumentException Exception		items < 1
lucene-solr-index-tree- getsplitdim-754e29a9	lucene-solr	org.apache.lucene.util. bkd.BKDReader.Index Tree	getSplitDim()	AssertionError		isLeafNode() != false && isLeafNode() != false
poi-hssfworkbook- getsheetat-28bfbaud	poi	org.apache.poi.hssf. usermodel. HSSFWorkbook	getSheetAt(int)	IllegalArgumentException Exception		index < 0 index > _sheets.size() - 1
lucene-solr-index-filenames- filenamefromgeneration- 6a235be4	lucene-solr	org.apache.lucene. index.IndexFileNames	fileNameFromGeneration(String, long)	AssertionError		(gen > 0) && gen != 0 && gen != -1 && (gen > 0)
asm-classnode-check- d81f67ef	asm	org.objectweb.asm. tree.ClassNode	check(final int)	UnsupportedClass VersionException		api < Opcodes.ASM7 && nestHostClass != null nestMembers != null && (api < Opcodes.ASM8 && access & Opcodes.ACC_ RECORD != 0 recordComponents != null) && (api < Opcodes. ASM9 && permittedSubclasses != null)
h2database-valueblob- getString-910f579c	h2database	org.h2.value.ValueBlob	getString()	getStringTooLong(charLength())		octetLength > Constants.MAX_STRING_LENGTH * 3 && (char Length >= 0L)
flink-shortparser-parsefield- 01ee92a5	flink	org.apache.flink.types. parser.ShortParser	parseField(byte[], int, int, char)	NumberFormatException Exception		length - 1 == 0 bytes[startPos] == delimiter && bytes[startPos] == '.' && bytes[startPos] != delimiter
byte-buddy- decoratingdynamicitybuilder- innertypeof-6dc147aa	byte-buddy	net.bytebuddy. dynamic.scaffold. inline.Decorating DynamicTypeBuilder	innerTypeOf(Type Description)	Unsupported Operation Exception		TRUE
lucene-solr- idversionsegmenttermsenum- postings-4e9220f3	lucene-solr	org.apache.lucene. sandbox.codecs. idversion.IDVersion SegmentTermsEnum	postings(PostingsEnum, int)	AssertionError		eof && eof
lucene-solr- columnatevaluator-dowork- ced1c4eb	lucene-solr	org.apache.solr.client. solrj.io.eval.ColumnAt Evaluator	doWork(Object, Object)	IOException		(value1 instanceof Matrix)
santuario-xml-security-java- domxpathfilter2transform- init-1a648a6c	santuario- xml-security- java	org.apache.jaxp.xml. dsig.internal.dom. DOMXPathFilter2 Transform	init(TransformParameter Spec)	InvalidAlgorithm Parameter Exception		params instanceof XPathFilter2ParameterSpec && params != null
lucene-solr-util-get-d7a091c7	lucene-solr	org.apache.lucene.util. fst.Util	get(FST<T>, BytesRef)	AssertionError		fst.inputType != FST.INPUT_TYPE.BYTE1 && fst.inputType != FST.INPUT_TYPE.BYTE1

id	project	class	method	exception	wit Condition	miner Condition
accumulo-merge-mergomatic-46884133	accumulo	org.apache.accumulo.core.util.Merge	mergomatic(AccumuloClient, String, Text, Text, long, boolean)	IllegalArgumentException Exception		table.equals(MetadataTable.NAME)
poi-formularecordaggregate-88371597	poi	org.apache.poi.hssf.record.aggregates.FormulaRecord	FormulaRecordAggregate(Record, SharedValueManager)	RecordFormatException Exception		stringRec == null && formulaRec.hasCachedResultString() && svm != null
jackrabbit-oak-pathutils-getname-717c5327	jackrabbit-oak	org.apache.jackrabbit.oak.commons.PathUtils	getName(String)	AssertionError		!(isValid(path)) && !(isValid(path))
flink-simplecollectingoutputview-reset-a0b512cd	flink	org.apache.flink.runtime.io.disk.SimpleCollectingOutputView	reset()	IllegalStateException Exception		this.fullSegments.size() != 0
jackrabbit-move-accept-b083095b	jackrabbit	org.apache.jackrabbit.jcr2spi.operation.Move	accept(OperationVisitor)	AssertionError		status != STATUS_PENDING && status != STATUS_PENDING
spring-data-commons-persistentproperty-getrequiredfield-c8da282f	spring-data-commons	org.springframework.data.mapping.PersistentProperty	getRequiredField()	IllegalArgumentException Exception		getField() == null
lucene-solr-integersequence-35878322	lucene-solr	org.apache.lucene.util.fst.IntSequence	add(IntRef, IntRef)	AssertionError		prefix == null && prefix == null
jackrabbit-checkin-accept-488e88cc	jackrabbit	org.apache.jackrabbit.jcr2spi.operation.Checkin	accept(OperationVisitor)	AssertionError		status != STATUS_PENDING && status != STATUS_PENDING
flink-overwritespec-apply-248821c2	flink	org.apache.flink.table.planner.plan.abilities.sink.OverwriteSpec	apply(DynamicTableSink)	TableException		!(tableSink instanceof SupportsOverwrite)
poi-hs-ls-slide-show-reorder-slide-7fd333d6	poi	org.apache.poi.hslf.usermodel.HSLFSlideShow	reorderSlide(int, int)	IllegalStateException Exception		_documentRecord.getSlideSlideListWithText() == null && !(oldSlideNumber > _slides.size() newSlideNumber > _slides.size()) && !(oldSlideNumber < 1 newSlideNumber < 1)
commons-configuration-2-configuration-beanutils-90742488	commons-configuration2	org.apache.commons.beanutils.ConstructorArg	forBeanDeclaration(final BeanDeclaration, final String)	AssertionError		Objects.requireNonNull(beanDeclaration, "beanDeclaration")
jackrabbit-privilegebits-add-102deb23	jackrabbit	org.apache.jackrabbit.core.security.authorization.PrivilegeBits	add(PrivilegeBits)	UnsupportedOperationException Exception		!(d instanceof ModifiableData)

id	project	class	method	exception	wit Condition	miner Condition
jackson-databind-managedreferenceproperty-setundreturn-b7a797d0	jackson-databind	com.fasterxml.jackson.databind.deser.impl.ManagedReferenceProperty	setAndReturn(Object, Object)	IllegalStateException		!(value instanceof Map<?, ?>) && !(value instanceof Collection<?>) && !(value instanceof Object[]) && _isContainer & value != null
jackrabbit-authcontextprovider-getauthcontext-aaa4ed5a	jackrabbit	org.apache.jackrabbit.core.security.authentication.AuthContextProvider	getAuthContext(Credentials, Subject, Session, PrincipalProvider Registry, String, String)	RepositoryException		!(isJAAS()) && !(isLocal())
flink-binaryrowdata-sermillar-36956015	flink	org.apache.flink.table.data.binary.BinaryRowData	setNullAt(int)	AssertionError		!(i >= 0) && !(i >= 0)
flink-reximplify-simplifyands-87d4a78c	flink	org.apache.calcite.rex.RexSimplify	simplifyAnds(Iterable<? extends RexNode>)	UnsupportedOperationException		paranoid
pdfbox-concatenate-process-77a6c48b	pdfbox	org.apache.pdfbox.contentstream.operator.state.Concatenate	process(Operator, List<COSBase>)	MissingOperandException		arguments.size() < 6
lucene-solr-geo3ddistancecalculator-calcoxybydisfrfrompt-yhorizaxisdeg-e8c4d479	lucene-solr	org.apache.lucene.spatial.spatial4j.Geo3dDistanceCalculator	calcBoxByDisFromPt_spatial.spatial4j.Geo3d double, SpatialContext)	UnsupportedOperationException		TRUE
jmonkeyengine-fbxlimbnode-createskeleton-b5b22bf3	jmonkeyengine-fbxlimbnode-createskeleton-b5b22bf3	jmonkeyenginecom.jme3.scene.plugins.fbx.anim.FbxLimbNode	createSkeleton(FbxNode)	UnsupportedOperationException		skeletonHolderNode instanceof FbxLimbNode
poi-xdgfshape-getboundsaspath-13a30c22	poi	org.apache.poi.xdgf.usermodel.XDGfShape	getBoundsAsPath()	Exception		getHeight()
poi-xddftrunproperties-setcharacterspacing-74813199	poi	org.apache.poi.xddf.usermodel.text.XDDFRunProperties	setCharacterSpacing(Double)	XDGfException.error("Height not specified", this)		spacing != null && spacing < -4000 4000 < spacing
spoon-elementsourcefragment-getsourcefragmentof-67820a11	spoon	spoon.support.sniper.internal.ElementSourceFragment	getSourceFragmentOf(SourcePositionHolder, int)	SpoonException		element != null && getElement() != element && firstChild == null && !getStart() == start && getEnd() == end) && getEnd() >= end && getStart() <= start && !getEnd() <= start
asm-modulenode-modulenode-a03d74c5	asm	org.objectweb.asm.tree.ModuleNode	ModuleNode(final String, final int, final String)	IllegalStateException		getClass() != ModuleNode.class

id	project	class	method	exception	wit Condition	miner Condition
lucene-solr-geo3dshape- getarea-f20b8c40	lucene-solr	org.apache.lucene. spatial.spatial4j.Geo3d Shape	getArea(SpatialContext)	Unsupported Operation Exception		TRUE
flink-binaryarraydata- getarray-064a90b5	flink	org.apache.flink.table. data.binary.Binary ArrayData	getArray(int)	AssertionError		!(pos < size) && !(pos < size)
wildfly-elytron- credentialstore-flush- 16d0f9b7	wildfly- elytron	org.wildfly.security. credential.store. CredentialStore	flush()	log.nonModifiable CredentialStore("flush")		!(isModifiable())
poi-thumbnail- getthumbnailaswmf- 7ae2d3bd	poi	org.apache.poi.hpsf. Thumbnail	getThumbnailAsWMF()	HPSFException		getClipboardFormatTag() != CFTAG_WINDOWS
wicket- compoundauthorizationstrategy- add-da819d74	wicket	org.apache.wicket. authorization. strategies.Compound AuthorizationStrategy	add(Authorization Strategy)	IllegalArgumentException Exception	null == strategy	strategy == null
flink-copyingiterator-iterator- 2e154247	flink	org.apache.flink.api. common.functions.util. CopyingIterator	iterator()	TraversableOnce Exception	!available	!(available)
h2database-simpleresultset- getascii-stream-ch165671	h2database	org.h2.tools.Simple ResultSet	getAsciiStream(int)	getUnsupported Exception()	TRUE	TRUE
h2database-jdbcresultset- update-ref-626d3a5d	h2database	org.h2.jdbc.JdbcResult Set	updateRef(int, Ref)	unsupported("ref") Exception	TRUE	TRUE
camel-j-crendpoint- setsessionlivecheckintervalonstart- e1fa0332	camel	org.apache.camel. component.jcr.Jcr Endpoint	setSessionLiveCheck IntervalOnStart(long)	IllegalArgumentException Exception	sessionLiveCheckIntervalOnStart <= 0	sessionLiveCheckIntervalOnStart <= 0
jackrabbit- clientaccesscontrolmanager- setpolicy-3c46f584	jackrabbit	org.apache.jackrabbit. rmi.client.security. ClientAccessControl Manager	setPolicy(String, Access ControlPolicy)	Unsupported Repository Operation Exception	TRUE	TRUE
poi-stringing-stringing- 89b41c00	poi	org.apache.poi.ss. formula.ptg.StringPtg	StringPtg(String)	IllegalArgumentException Exception	value.length() > 255	value.length() > 255
camel-saxstreamxmlreader- setProperty-c39fc4f3	camel	org.apache.camel. component.stax.Stax StreamXMLReader	setProperty(String, Object)	SAXNot Recognized Exception	((("http://xml.org/sax/properties/lexical-handler".equals(name)))	!("http://xml.org/sax/properties/lexical-handler".equals(name))
joda-time- datetimeformatterbuilder- append-c4a2a966	joda-time	org.joda.time.format. DateTimeFormatter Builder	append(DateTime Formatter)	IllegalArgumentException Exception	formatter == null	formatter == null

id	project	class	method	exception	wit Condition	miner Condition
jmonkeyengine-batchnode-onmaterialchange-309c4d96	jmonkeyenginecom	jme3.scene.BatchNode	onMaterialChange(Geometry)	Unsupported Operation Exception	TRUE	TRUE
spoon-patternbuilder-visititnewarray-849bd8b2	spoon	spoon.smpl.pattern.PatternBuilder	visitCNewArray(CtNewArray<T>)	NotImplemented Exception	TRUE	TRUE
accumulo-authenticate-result-iset-a918ca97	accumulo	org.apache.accumulo.core.clientImpl.thrift.ClientService.authenticate_result	isSet(_Fields)	IllegalArgumentException Exception	field == null	field == null
accumulo-updateerrors-iset-585f50d8	accumulo	org.apache.accumulo.core.dataImpl.thrift.UpdateErrors	isSet(_Fields)	IllegalArgumentException Exception	field == null	field == null
h2database-interval-ofdayshours-786d0dc3	h2database	org.h2.api.Interval	ofDaysHours(long, int)	IllegalArgumentException Exception	negative && (days > 0 hours > 0)	days > 0 hours > 0 && days hours < 0
flink-columnarmapdata-equals-58d3755a	flink	org.apache.flink.table.data.ColumnarMapData	equals(Object)	Unsupported Operation Exception	TRUE	TRUE
spoon-patternbuilder-visitcitymemberwildcardimportreference-1af55699	spoon	spoon.smpl.pattern.PatternBuilder	visitCtTypeMemberWildcardImportReference(CtTypeMemberWildcardImportReference)	NotImplemented Exception	TRUE	TRUE
lucene-solr-contextquery-hashcode-afbc3f51	lucene-solr	org.apache.lucene.search.suggest.document.ContextQuery	hashCode()	Unsupported Operation Exception	TRUE	TRUE
flink-savepointenvironment-gettaskeventdispatcher-184f5f65	flink	org.apache.flink.state.api.runtime.SavepointEnvironment	getTaskEventDispatcher()	Unsupported Operation Exception	TRUE	TRUE
lucene-solr-zksoilresourceloader-getconfigdir-36aadcb5	lucene-solr	org.apache.solr.cloud.ZkSolrResourceLoader	getConfigDir()	Zookeeper Exception	TRUE	TRUE
commons-lang-conversion-intarraytolong-b743b676	commons-lang	org.apache.commons.lang3.Conversion	intArrayToLong(final int[], final int)	IllegalArgumentException Exception	dstPos + 32 * nInts >= 96 && (srcPos != 0 src.length != 0) && nInts - 1 * 32 + dstPos >= 64 && (src.length == 0 && srcPos == 0 0 == nInts)	nInts - 1 * 32 + dstPos >= 64 && (src.length == 0 && srcPos == 0 0 == nInts)
xmlgraphics-fop-abstractbaselayoutmanager-reset-6838f82f	xmlgraphics-fop	org.apache.fop.layoutmgr.AbstractLayoutManager	reset()	Unsupported Operation Exception	TRUE	TRUE

id	project	class	method	exception	wit Condition	miner Condition
activiti- activityeventimpl- activityeventimpl- c929685d	Activiti	org.activiti.engine. delegate.event.impl. ActivitiEventImpl	ActivitiEventImpl(Object, ActivitiEventType)	ActivitiIllegal Argument Exception	entity == null	entity == null
lucene-solr-packagestoreapi- validatename-5aafdc45	lucene-solr	org.apache.solr. filestore.PackageStore API	validateName(String, boolean)	SolrException	null == path	path == null
lucene-solr- emptydocvaluesproducer- getbinary-42178c9a	lucene-solr	org.apache.lucene. index.EmptyDocValues Producer	getBinary(FieldInfo)	Unsupported Operation Exception	TRUE	TRUE
camel- lightweightruntimecamelcontext- getbeanprocessorfactory- 33d4f54b	camel	org.apache.camel. impl.lw.Lightweight RuntimeCamelContext	getBeanProcessorFactory()	Unsupported Operation Exception	TRUE	TRUE
flink- dummystreamexecutionenvironment- enablecheckpointing- 18bb09aa	flink	org.apache.flink.table. planner.utils.Dummy StreamExecution Environment	enableCheckpointing()	Unsupported Operation Exception	TRUE	TRUE
jmonkeyengine- bitmapcharacterstet-setstyle- a26552cd	jmonkeyenginecom	jme3.font.Bitmap CharacterSet	setStyle(int)	RuntimeException	size > 1	characters.size() > 1
jfreechart-standarddiakale- setmajortickincrement- db688e91	jfreechart	org.jfree.chart.plot.dial. StandardDialScale	setMajorTickIncrement(double)	IllegalArgument Exception	increment <= 0	increment <= 0.0
activiti-modelqueryimpl- modeltenantid-8434f6b8	Activiti	org.activiti.engine.impl. ModelQueryImpl	modelTenantId(String)	ActivitiIllegal Argument Exception	null == tenantId	tenantId == null
lucene-solr-characterutils- tocodepoints-0b71cd83	lucene-solr	org.apache.lucene. analysis.CharacterUtils	toCodePoints(char[], int, int, int)	IllegalArgument Exception	srcLen < 0	srcLen < 0
commons-math- accuratemath- incrementexact-979f8359	commons- math	org.apache.commons. math4.core.jdkmath. AccurateMath	incrementExact(final long)	Arithmetic Exception	Long.MAX_VALUE == n	n == Long.MAX_VALUE
poi-intlist-remove-d6fb08cd	poi	org.apache.poi.util.Int List	remove(final int)	IndexOutOfBoundsException	_limit <= index	index >= _limit
accumulo- conditionalmutation-isset- e1829f59	accumulo	org.apache.accumulo. core.dataImpl.thrift. TConditionalMutation	isSet(_Fields)	IllegalArgument Exception	field == null	field == null

id	project	class	method	exception	wit Condition	miner Condition
accumulo-zooreservation-attempt-bcf58451	accumulo	org.apache.accumulo.fate.zookeeper.ZooReservation	attempt(ZooReaderWriter, String, String, String)	IllegalArgumentException Exception	reservationID.contains(":")	reservationID.contains(":")
flink-messageiterator-remove-905d0eb9	flink	org.apache.flink.graph.remove() spargel.MessageIterator	remove()	UnsupportedOperation Exception	TRUE	TRUE
lucene-solr-jaspelllookup-4772eb49	lucene-solr	org.apache.lucene.search.suggest.jaspell.JaspellLookup	lookup(CharSequence, Set<ByteRef>, boolean, int)	IllegalArgumentException Exception	contexts != null	contexts != null
jackrabbit-sessionimpl-propertyexists-071ccc9d	jackrabbit	org.apache.jackrabbit.jcr2spi.SessionImpl	propertyExists(String)	RepositoryException	lalive	lalive
commons-math-abstractmultiplelinearregressionmath	commons-math	org.apache.commons.math4.legacy.stat.regression.AbstractMultipleLinearRegression	newSampleData(double[], int, int)	InsufficientDataException	data.length == nob * (nvars + 1) && nob * nvars <= nvars && data != null nob * nvars + 1 && data != null	
camel-lraurlbuilder-host-6d52ebf2	camel	org.apache.camel.service.lra.LRAUrlBuilder	host(String)	IllegalStateException Exception	(this.host != null)	this.host != null
h2database-simpleresultset-getrowid-6c3d2b22	h2database	org.h2.tools.SimpleResultSet	getRowId()	getUnsupportedException()	TRUE	TRUE
lucene-solr-lucene50compressingtermvectorsformat-lucene50compressingtermvectorsformat-compressing-cb461d6f	lucene-solr	org.apache.lucene.backward_codecs.lucene50.CompressingTermVectorsFormat	TermVectorsFormat(String, String, CompressionMode, int, int)	IllegalArgumentException Exception	chunkSize < 1	chunkSize < 1
poi-cfrulebase-setcomparisonoperation-b53c9832	poi	org.apache.poi.hssf.record.CFRuleBase	setComparisonOperation(byte)	IllegalArgumentException Exception	(operation < 0 operation > ComparisonOperator.max_operator)	operation < 0 operation > ComparisonOperator.max_operator
itext7-bytebuffer-charat-622eb329	itext7	com.itextpdf.kernel.xmp.impl.ByteBuffer	charAt(int)	IndexOutOfBoundsException	index >= length	!(index < length)
wicket-appendingstringbuffer-setlength-ec798a51	wicket	org.apache.wicket.util.string.appendingstring.Buffer	setLength(final int)	StringIndexOutOfBoundsException	newLength < 0	newLength < 0

id	project	class	method	exception	wit Condition	miner Condition
jackrabbit-oak- datastoreblobstore-init- 70aaba07	jackrabbit- oak	org.apache.jackrabbit. oak.plugins.blob. datastore.DataStore BlobStore	init(String)	Unsupported Operation Exception	TRUE	TRUE
santuario-xml-security-java- apachecanonicalizer- isfeaturesupported-79e78c0	santuario- xml-security- java	org.apache.jsp.xml. dsig.internal.dom. ApacheCanonicalizer	isFeatureSupported(String)	NullPointerException Exception	feature == null	feature == null
activiti- executorpertenantasyncexecutor- getprocessengineconfiguration- db4b19ca	Activiti executor	org.activiti.engine.impl. asyncexecutor. multitenant.Executor PerTenantAsync Executor	getProcessEngine Configuration()	Unsupported Operation Exception	TRUE	TRUE
itext7-pdfjks7- setexternaldigest-e402b0c5	itext7	com.itextpdf. signatures.PdfPKCS7	setExternalDigest(byte[], byte[], String)	new PdfException(SignException MessageConstant. UNKNOWN_KEY_ ALGORITHM).set MessageParams(digestEncryption Algorithm)	digestEncryptionAlgorithm.equals("RSA") && digestEncryption Algorithm != null	!(digestEncryptionAlgorithm.equals("ECDSA")) && (digest EncryptionAlgorithm.equals("DSA")) && !(digestEncryption Algorithm.equals("RSA")) && digestEncryptionAlgorithm != null
jmonkeyengine- socketconnector-close- c5d3ccd0	jmonkeyengine- kernel.tcp.Socket Connector	jmonkeyengine.com.jme3.network. kernel.tcp.Socket Connector	close()	Connector Exception	null == sock	sock == null
camel-jettyhttpcomponent- sethttpconfiguration- 83c6d5a6	camel	org.apache.camel. component.jetty.Jetty HttpComponent	setHttpConfiguration(Http Configuration)	IllegalArgumentException Exception	TRUE	TRUE
xmlgraphics-fop- systemfontmetric-snapper- getboundingbox-17d4053b	xmlgraphics- fop	org.apache.fop.render. java2d.SystemFont MetricsMapper	getBoundingBox(int, int)	Unsupported Operation Exception	TRUE	TRUE
xmlgraphics-fop-limiter- remove-69a6727f	xmlgraphics- fop	org.apache.fop. layoutmgr.LMiter	remove()	NoSuchElement Exception	curPos <= 0	!(curPos > 0)
lucene-solr- ebesubtractevaluator-dowork- 569678db	lucene-solr	org.apache.solr.client. solrj.io.eval. EBESubtractEvaluator	doWork(Object, Object)	IOException	first == null	null == first
lucene-solr- canberraevaluator-dowork- b12ac4ac	lucene-solr	org.apache.solr.client. solrj.io.eval.Canberra Evaluator	doWork(Object...)	IOException	TRUE	TRUE

id	project	class	method	exception	wit Condition	miner Condition
jmonkeyengine-lowpassfilter- sethighfreqvolume-d63f1172	jmonkeyengine-lowpassfilter- sethighfreqvolume-d63f1172	PassFilter	setHighFreqVolume(float)	IllegalArgumentException Exception	highFreqVolume > 1 highFreqVolume < 0	highFreqVolume < 0 highFreqVolume > 1
flink- manuallytriggeredscheduledexecutorservice- submit-c6b0924a	flink manuallytriggeredscheduledexecutorservice- submit-c6b0924a	org.apache.flink.core. testutils.Manually TriggeredScheduled ExecutorService	submit(Callable<T>)	Unsupported Operation Exception	TRUE	TRUE
lucene-solr-termsfacetmap- settemprefix-6a6cb49c	lucene-solr settemprefix-6a6cb49c	org.apache.solr.client. solr.request.json. TermsFacetMap	setTempPrefix(String)	IllegalArgumentException Exception	null == termPrefix	termPrefix == null
camel- lightweightruntimecamelcontext- getdebugger-3b564c20	camel lightweightruntimecamelcontext- getdebugger-3b564c20	org.apache.camel. impl.lw.Lightweight RuntimeCamelContext	getDebugger()	Unsupported Operation Exception	TRUE	TRUE
swingx-jxtipoftheday- showdialog-9113cd57	swingx showdialog-9113cd57	org.jdesktop.swing. JXTipOfTheDay	showDialog(Component, final Preferences, boolean)	IllegalArgumentException Exception	null == showOnStartupPref	showOnStartupPref == null
poi- packagerelationshipcollection-	poi packagerelationshipcollection-	org.apache.poi. openxml4j.opc. PackageRelationship Collection(OPCPackage, PackagePart)	PackageRelationship Collection(OPCPackage, PackagePart)	IllegalArgumentException Exception	container == null	container == null
packagerelationshipcollection- b84ea23e	packagerelationshipcollection- b84ea23e	Collection				
lucene-solr- codepointcountfilter- codepointcountfilter- ab264994	lucene-solr codepointcountfilter- codepointcountfilter- ab264994	org.apache.lucene. analysis. miscellaneous. CodepointCountFilter	CodepointCountFilter(TokenStream, int, int)	IllegalArgumentException Exception	min < 0	min < 0
jackrabbit-davresourceimpl- move-6bc9bc1f	jackrabbit move-6bc9bc1f	org.apache.jackrabbit. webdav.simple.Dav ResourceImpl	move(DavResource)	DavException	node == null	lexists()
poi-streamingsheetwriter- createtempfile-1e2d49ff	poi createtempfile-1e2d49ff	org.apache.poi.xssf. streaming.Streaming SheetWriter	createTempFile()	RuntimeException	TRUE	TRUE
jackrabbit- abstractrepositoryservice- checkout-bbce24fa	jackrabbit abstractrepositoryservice- checkout-bbce24fa	org.apache.jackrabbit. spi.common.Abstract RepositoryService	checkout(SessionInfo, NodeId, NodeId)	Unsupported Repository Operation Exception	TRUE	TRUE
jackrabbit-setprimarytype- create-4b1db2b7	jackrabbit create-4b1db2b7	org.apache.jackrabbit. jcr2spi.operation.Set PrimaryType	create(NodeState, Name)	IllegalArgumentException Exception	nodeState == null null == primaryTypeName	nodeState == null primaryTypeName == null

id	project	class	method	exception	wit Condition	miner Condition
camel-redefinition-param- e94825b	camel	org.apache.camel. model.rest.Rest Definition	param(RestOperation ParamDefinition)	IllegalArgumentException Exception	get\Verbs().isEmpty	get\Verbs().isEmpty()
spring-cloud-gcp-gcsession- append-53031e04	spring-cloud- gcp	org.springframework. cloud.gcp.storage. integration.Gcs Session	append(InputStream, String)	Unsupported Operation Exception	TRUE	TRUE
dubbo-unsafestringwriter- write-ad607b0a	dubbo	org.apache.dubbo. common.io.Unsafe StringWriter	write(char[], int, int)	IndexOutOfBoundsException	cs.length < off cs.length < len + off len < 0 len < -off off < 0 off > cs.length len < 0 off + len > cs.length off + len < 0	
poi- littleendianbytearrayoutputstream- writebyte-ad81caf3	poi	org.apache.poi.util. LittleEndianByteArray OutputStream	writeByte(int)	RuntimeException	_endIndex - _writeIndex < 1	1 > _endIndex - _writeIndex
jmonkeyengine-androididgl- glshadersource-d663f0f8	jmonkeyengine-androididgl-	jme3.renderer. android.AndroidGL	glShaderSource(int, String[, IntBuffer])	Unsupported Operation Exception	string.length != 1	string.length != 1
jackrabbit- abstractrepositoryservice- checkquerystatement- f6b91938	jackrabbit	org.apache.jackrabbit. spi.commons.Abstract RepositoryService	checkQueryStatement(SessionInfo, String, String, Map<String, String>)	Unsupported Repository Operation Exception	TRUE	TRUE
lucene-solr- keepwordfilterfactory- keepwordfilterfactory- 700cbb19	lucene-solr	org.apache.lucene. analysis. miscellaneous.Keep WordFilterFactory	KeepWordFilterFactory()	defaultCtor Exception()	TRUE	TRUE
jackrabbit-searchindex- addnode-c1092973	jackrabbit	org.apache.jackrabbit. core.query.lucene. SearchIndex	addNode(NodeState)	Unsupported Operation Exception	TRUE	TRUE
poi-ssfprintsetup-setscale- 25cf3d4e	poi	org.apache.poi.xssf. usermodel.XSSFPrint Setup	setScale(short)	POIXMLException	scale > 400 scale < 10	scale < 10 scale > 400
camel-xchangeendpoint- createconsumer-b10a9555	camel	org.apache.camel. component.xchange. XChangeEndpoint	createConsumer(Processor)	Unsupported Operation Exception	TRUE	TRUE
commons-text-strbuilder- replace-2694cf3f	commons- text	org.apache.commons. text.StrBuilder	replace(final StrMatcher, final String, final int, int, final int)	StringIndexOutOfBoundsException	startIndex < 0	startIndex < 0

id	project	class	method	exception	wit Condition	miner Condition
jmonkeyengine-selectorkernel-initialize-b78fb095	jmonkeyengine-kernel-top.Selector	Kernel	initialize()	IllegalStateException	null != thread	thread != null
accumulo-tabletresource-manager-newsconfilenamanager-21775d25	accumulo-tserver.TabletServerResourceManager	TabletResourceManager	newScanFileManager(ScanDispatch)	IllegalStateException	closed	closed
jackrabbit-pathfactoryimpl-create-934e6902	jackrabbit-impl-jackrabbit-spi.common.name	PathFactoryImpl	create(Name, int)	IllegalArgumentException	name == null	name == null
camel-datalakeserviceclientwrapper	camel-org.apache.camel.component.azure.storage.dataLakeClient	ClientWrapper	getDataLakeFileSystemName()	IllegalArgumentException	(fileSystemName == null) && (!true)	ObjectHelper.isEmpty(fileSystemName)
getdataLakeFileSystemClientWrapper-2a22a907	santuario-xml-security-java-xml-security-stream-writer	XMLSecurityStreamWriter	setNamespaceContext(NamespaceContext)	NullPointerException	context == null	context == null
activiti-eventssubscriptionqueryimpl-executionid-a7e4faa	Activiti-eventSubscriptionQueryImpl	EventSubscriptionQueryImpl	execute()	ActivitiIllegalArgument	executionId == null	executionId == null
entitymanagersessionfactory-entitymanagersessionfactory-a148bd48	Activiti-entityManagerSessionFactory	EntityManagerSessionFactory	init()	ActivitiIllegalArgument	entityManagerFactory == null	entityManagerFactory == null
itext7-unresolver-unresolver-87a37459	itext7-unresolver-unresolver-styledxmparser	Unresolver	init(String)	IllegalArgumentException	baseUri == null	baseUri == null
logging-log4j2-mutableinstant-inifromepochmilli-fcb067fc	logging-log4j2-mutableInstant	MutableInstant	initFromEpochMilli(long, final int)	IllegalArgumentException	nanoOfMillisecond >= 1000000 nanoOfMillisecond < 0	nanoOfMillisecond < 0 nanoOfMillisecond >= NANOS_PER_MILLI
itext7-imagedatafactory-createbig2-163aa180	itext7-imageDataFactory	ImageDataFactory	createBig2(byte[], int)	IllegalArgumentException	page < 1	page < 1

id	project	class	method	exception	wit Condition	miner Condition
jackrabbit-databasefilesystem-exists-a8a390f5	jackrabbit	org.apache.jackrabbit.core.fs.db.DatabaseFileSystem	exists(String)	IllegalStateException	initialized	initialized
lucene-solr-log4j2watcher-setloglevel-157459d9	lucene-solr	org.apache.solr.logging.log4j2.Log4j2Watcher	setLogLevel(String, String)	AssertionError		level == null && level == null
itext7-xmputils-converttointeger-3bf1d261	itext7	com.itextpdf.kernel.xmp.XMPUtils	convertToInteger(String)	XMPEException		rawValue == null rawValue.length() == 0
flink-replicatinginputformat-getruntimecontext-bd6bcb68	flink	org.apache.flink.api.common.io.ReplicatingInputFormat	getRuntimeContext()	RuntimeException		!(this.replicatedIF instanceof RichInputFormat)

Table A.2: Prex evaluation sample (150 reports): Manual-review labels.

id	miner Correctness	wit Correctness	miner Correctness	source Checkability	wit Checkability	miner Checkability	documentation	check Effort
jackson-databind-lassutil-throwasmappingexception-4bec3504	N/A		TRUE	Input Validation	N/A	TRUE	NONE	INTERMEDIATE
jackrabbit-restore-accept-9f7349be	N/A		TRUE	Private state	N/A	FALSE	NONE	PURE
accumulo-dynamicbloombfilter-or-9abc53ab	N/A		TRUE	Hybrid Untestable	N/A	FALSE	NONE	COMPLEX
asm-sourceinterpreter-sourceinterpreter-9b9d4923	N/A		TRUE	Public Visible State	N/A	TRUE	NONE	INTERMEDIATE
jmonkeyengine-terrainquad-supportmultiplicecollisions-c9162c23	N/A		TRUE	Private state	N/A	FALSE	NONE	INTERMEDIATE
pdfbox-datatype-setvalue-fbfc4c61	N/A		TRUE	Hybrid Untestable	N/A	FALSE	NONE	COMPLEX
lucene-solr-bytesequencoutput-subtract-96b2c475	N/A		TRUE	Input Validation	N/A	TRUE	NONE	PURE
lucene-solr-builder-build-50b55036	N/A		TRUE	Private state	N/A	FALSE	NONE	COMPLEX
lucene-solr-indexandtaxonomyrevision-open-05d8b722	N/A		TRUE	Input Validation	N/A	TRUE	NONE	PURE
joda-time-localdate-get-05ba4eeb	N/A		TRUE	Hybrid Testable	N/A	TRUE	WORD	INTERMEDIATE
wicket-subtree-subtree-a768dae	N/A		TRUE	Input Validation	N/A	TRUE	NONE	INTERMEDIATE
lucene-solr-kmnevaluator-dowork-ebcd6c2	N/A		TRUE	Public Visible State	N/A	TRUE	NONE	COMPLEX
camel-jmxendpoint-createconsumer-a86508f9	N/A		TRUE	Hybrid Testable	N/A	TRUE	NONE	COMPLEX
camel-defaulthealthcheckregistry-register-357262f4	N/A		TRUE	Input Validation	N/A	TRUE	NONE	COMPLEX
spoon-smpljavaidl-getwrappedelement-d241f2cf	N/A		TRUE	Input Validation	N/A	TRUE	NONE	INTERMEDIATE
flink-hiveparserbasemanalytics-charsetstring-453c6f33	N/A		TRUE	Hybrid Testable	N/A	TRUE	NONE	COMPLEX
swingx-xtaskpane-property-getsessionstate-5746675c	N/A		TRUE	Hybrid Untestable	N/A	TRUE	NONE	COMPLEX
flink-nestedrowdata-setlong-f1c14851	N/A		TRUE	Hybrid Untestable	N/A	TRUE	NONE	COMPLEX
jmonkeyengine-skeletonloader-endelement-e4eb2c1d	N/A		TRUE	Input Validation	N/A	TRUE	VAR	PURE
commons-lang-failablelongunaryoperator-andthen-f5c77176	N/A		TRUE	Unconditionally Throw	N/A	TRUE	NONE	PURE
lucene-solr-geo3distancecalculator-area-1cddd22b	N/A		TRUE	Input Validation	N/A	TRUE	NONE	PURE
wicket-abstractpageableview-setitemsperpage-c7722829	N/A		TRUE	Public Visible State	N/A	TRUE	NONE	PURE
lucene-solr-indextree-getsplittim-75de29a9	N/A		TRUE	Hybrid Untestable	N/A	TRUE	NONE	PURE
poi-hssfworkbook-getsheetat-28bfbaed	N/A		TRUE	Input Validation	N/A	TRUE	NONE	COMPLEX
lucene-solr-indexfilenames-filenamefromgeneration-6a235be4	N/A		TRUE	Input Validation	N/A	TRUE	NONE	INTERMEDIATE
asm-classnode-check-d81f676f	N/A		TRUE	Hybrid Untestable	N/A	TRUE	NONE	COMPLEX
h2database-valueblob-getstring-910f579c	N/A		TRUE	Private state	N/A	FALSE	NONE	PURE
flink-shortparser-parsefield-01ee92e5	N/A		TRUE	Input Validation	N/A	TRUE	WORD	COMPLEX
byte-buddy-decoratingdynamicitypebuilder-innertypeof-6dc147aa	N/A		TRUE	Unconditionally Throw	N/A	TRUE	NONE	PURE
lucene-solr-idversionsegmentinternum-postings-4e9220f3	N/A		TRUE	Private state	N/A	FALSE	NONE	PURE
lucene-solr-columnnevaluator-dowork-ced1c4eb	N/A		TRUE	Input Validation	N/A	TRUE	NONE	INTERMEDIATE
santuario-xml-security-java-domxpathfilter2transform-init-1a648a6c	N/A		TRUE	Input Validation	N/A	TRUE	NONE	INTERMEDIATE
lucene-solr-util-get-d7a091c7	N/A		TRUE	Input Validation	N/A	TRUE	NONE	PURE
accumulo-merge-mergomatic-d6884133	N/A		TRUE	Input Validation	N/A	TRUE	NONE	PURE

id	wit Correctness	miner Correctness	source Checkability	wit Checkability	miner Checkability	documentation	check Effort
poi-formularecordaggregate-formularecordaggregate-88371597	N/A	TRUE	Input Validation	N/A	TRUE	NONE	COMPLEX
jackrabbit-osk-pathutilis-getname-7f7c5327	N/A	TRUE	Input Validation	N/A	TRUE	NONE	PURE
flink-simplecollectingoutputview-view-reset-a0b512cd	N/A	TRUE	Private state	N/A	FALSE	NONE	PURE
jackrabbit-move-accept-b083095b	N/A	TRUE	Private state	N/A	FALSE	NONE	PURE
spring-data-commons-persistentproperty-getrequiredfield-c8da282f	N/A	FALSE	Public Visible State	N/A	TRUE	NONE	INTERMEDIATE
lucene-solr-intsequenceoutputs-add-35878322	N/A	TRUE	Input Validation	N/A	TRUE	NONE	PURE
jackrabbit-checkin-accept-488e88cc	N/A	TRUE	Private state	N/A	FALSE	NONE	PURE
flink-overwritespec-apply-248821c2	N/A	TRUE	Input Validation	N/A	TRUE	NONE	INTERMEDIATE
poi-hsflideshow-reorderslide-7fd333d6	N/A	TRUE	Hybrid Untestable	N/A	FALSE	NONE	COMPLEX
commons-configuration-constructortarg-forbeanddeclaration-90742488	N/A	TRUE	Input Validation	N/A	TRUE	VAR	PURE
jackrabbit-privilegebits-add-1024cb23	N/A	TRUE	Private state	N/A	FALSE	WORD	INTERMEDIATE
jackson-databind-managedreferenceproperty-setandreturn-b7a797d0	N/A	TRUE	Hybrid Untestable	N/A	FALSE	NONE	COMPLEX
jackrabbit-authcontextprovider-getauthcontext-aaa4ed5a	N/A	TRUE	Public Visible State	N/A	TRUE	WORD	COMPLEX
flink-binaryrowdata-setnullat-36956015	N/A	TRUE	Input Validation	N/A	TRUE	NONE	PURE
flink-reximplify-simplifyands-87d4a78c	N/A	TRUE	Hybrid Untestable	N/A	FALSE	NONE	COMPLEX
pdfbox-concatenate-process-77a6c48b	N/A	TRUE	Input Validation	N/A	TRUE	NONE	PURE
lucene-solr-geo3ddistancecalculator-calculatorbydisfrompt-yhorizaxisdeg-e8cd4d79	N/A	TRUE	Unconditionally Throw	N/A	TRUE	NONE	PURE
jmonkeyengine-fbxlimbnode-createskeleton-b5b22bf3	N/A	TRUE	Input Validation	N/A	TRUE	NONE	PURE
poi-xdgfshape-getboundsaspath-13a30c22	N/A	TRUE	Private state	N/A	TRUE	NONE	INTERMEDIATE
poi-xddfrunproperties-setcharacterspacing-74813199	N/A	TRUE	Input Validation	N/A	TRUE	NONE	PURE
spoon-elementsourcefragment-getsourcefragmentof-67820a11	N/A	TRUE	Hybrid Untestable	N/A	TRUE	NONE	COMPLEX
asm-modulenode-modulenode-a03d74c5	N/A	TRUE	Hybrid Testable	N/A	TRUE	NONE	INTERMEDIATE
lucene-solr-geo3dshape-getarea-f20b8c40	N/A	TRUE	Unconditionally Throw	N/A	TRUE	NONE	PURE
flink-binaryarraydata-getarray-064a90b5	N/A	TRUE	Hybrid Untestable	N/A	TRUE	NONE	PURE
wildfly-elytron-credentialstore-flush-16d0f9b7	N/A	TRUE	Public Visible State	N/A	TRUE	POOR	PURE
poi-thumbnaill-getthumbnaillastwmf-7ae2d3bd	N/A	TRUE	Public Visible State	N/A	TRUE	VAR	PURE
wicket-compoundauthorizationstrategy-add-da819d74	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
flink-copyingiterator-iterator-2e154247	TRUE	TRUE	Private state	FALSE	FALSE	NONE	INTERMEDIATE
h2database-simpleresultset-getasciistream-cb165671	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
h2database-jdbcresultset-updateref-626d3a5d	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	WORD	PURE
camel-jrendpoint-setsessionlivecheckintervalstart-e1fa0332	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
jackrabbit-elientaccesscontrolmanager-setpolicy-3c46f584	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
poi-stringpg-stringpg-89b41c00	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
camel-staxstream.xmlreader-setproperty-c39fe4f3	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	INTERMEDIATE
joda-time-datetimeformatterbuilder-append-c4a2a966	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
jmonkeyengine-batchnode-onmaterialchange-309c4d96	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE

id	wt Correctness	minor Correctness	source Checkability	wt Checkability	minor Checkability	documentation	check Effort
spoon-patternbuilder-visitnewarray-849a8b2	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
accumulo-authenticate-result-isset-a9186a97	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
accumulo-updatereaders-isset-585f50d8	TRUE	TRUE	Input Validation	TRUE	TRUE	POOR	PURE
h2database-interval-ofdayshours-78600dc3	TRUE	TRUE	Input Validation	TRUE	TRUE	WORD	INTERMEDIATE
flink-columnarinputdata-equals-58d3755a	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
spoon-patternbuilder-visitctypememberwildcardimportreference-laf5699	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
lucene-solr-contextquery-hascode-afbc3f51	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
flink-savepointenvironment-gettaskeventdispatcher-184f5f65	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
lucene-solr-zkresolverceloader-getconfigdir-36aadb5	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
commons-lang-conversion-intarraytolong-b743b676	TRUE	TRUE	Input Validation	TRUE	TRUE	VAR	INTERMEDIATE
xmlgraphics-fop-abstractbaselayoutmanager-reset-683882f	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
activiti-activityeventimpl-activityeventimpl-c929685d	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
lucene-solr-packagetestapi-validatefilename-5aa4dd5	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
lucene-solr-emptydocvaluesproducer-getbinary-42178e9a	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
camel-lightweightruntimecamelcontext-getbeanprocessorfactory-3344f54b	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
flink-dummystreamexecutionenvironment-enablecheckpointing-18bb09aa	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
jmonkeyengine-bitmapcharacter-setstyle-a26552cd	FALSE	TRUE	Private state	N/A	FALSE	NONE	PURE
jfreechart-standarddialscale-setmajortickincrement-db688e91	TRUE	TRUE	Input Validation	TRUE	TRUE	VAR	PURE
activiti-modelqueryimpl-modeltenantid-8434f6b8	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
lucene-solr-characterutils-tocodepoints-db71cd83	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
commons-math-accuratemath-incrementexact-979f8359	TRUE	TRUE	Hybrid Unstable	FALSE	TRUE	WORD	PURE
poi-intlist-remove-d6fb08cd	TRUE	TRUE	Input Validation	TRUE	TRUE	VAR	PURE
accumulo-tconditionalmutation-isset-e1829f59	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
accumulo-zooreservation-attempt-bcf58451	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
flink-messageiterator-remove-905d0eb9	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
lucene-solr-jspelllookup-lookup-d7f2eb49	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
jackrabbit-sessionimpl-propertyexists-071ec9d	TRUE	TRUE	Private state	FALSE	TRUE	POOR	PURE
commons-math-abstractmultiplelinearregression-newsampleddata-caffd664	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	INTERMEDIATE
camel-lraurbuilder-host-6d52ebf2	TRUE	TRUE	Private state	TRUE	FALSE	NONE	PURE
h2database-simpleresultset-getrowid-6c3db22	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
lucene-solr-lucene50compressingtermvectorsformat-lucene50compressingtermvectorsformat-cb461d6f	TRUE	TRUE	Input Validation	TRUE	TRUE	POOR	PURE
poi-cfrulebase-setcomparisonoperation-b53c9832	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
itext7-bytebuffer-charat-622eb329	TRUE	TRUE	Hybrid Unstable	FALSE	TRUE	NONE	INTERMEDIATE
wicket-appendingstringbuffer-setlength-ec798a51	TRUE	TRUE	Input Validation	TRUE	TRUE	VAR	PURE
jackrabbit-ok-datastoreblobstore-init-70aab07	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
santuario-xml-security-java-apachecanonicalizer-isfeaturesupported-79fe78c0	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE

id	wt Correctness	miner Correctness	source Checkability	wt Checkability	miner Checkability	documentation	check Effort
activiti-executorpertenantasyncexecutor-getprocessengineconfiguration-db4b19ca	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
itext7-pdfpks7-setexternaldigest-e402b0c5	FALSE	TRUE	Input Validation	N/A	TRUE	NONE	COMPLEX
jmonkeyengine-socketconnector-close-c5d3acd0	TRUE	TRUE	Private state	FALSE	FALSE	NONE	PURE
camel-jettyhttpcomponent-sethttpconfiguration-83c6d5a6	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	WORD	PURE
xmgraphics-fop-systemfontmetricmapper-getboundingbox-17d4053b	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
xmgraphics-fop-limiter-remove-69a6727f	TRUE	TRUE	Private state	FALSE	TRUE	NONE	PURE
lucene-solr-cbesubtractevaluator-dowork-569678db	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
lucene-solr-canberraevaluator-dowork-b12ac4ac	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
jmonkeyengine-lowpassfilter-sethighreqvolume-d63f1172	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
flink-manuallytriggeredscheduledexecutor-service-submit-e6b0924a	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
lucene-solr-termsfacetmap-settermprefix-6a6cb49c	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
camel-lightweighttraintimecamelcontext-getdebugger-3b564c20	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
swingx-jxtpoftheday-showdialog-9113c057	TRUE	TRUE	Input Validation	TRUE	TRUE	VAR	PURE
poi-packagerelationshipcollection-packagerelationshipecollection-b84ea23e	TRUE	TRUE	Input Validation	TRUE	TRUE	POOR	PURE
lucene-solr-codepointcountfilter-codepointcountfilter-ab264994	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
jackrabbit-davresourcempl-move-6bc9bc1f	FALSE	TRUE	Private state	N/A	TRUE	NONE	PURE
poi-streamingsheetwriter-createtemplate-1e2d49ff	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
jackrabbit-abstractrepository-service-checkout-bbce246a	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	VAR	PURE
jackrabbit-setprimarytype-create-4bfdb2b7	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
camel-restdefinition-param-e9d4825b	FALSE	TRUE	Public Visible State	N/A	TRUE	NONE	PURE
spring-cloud-gcp-gcsession-append-53031e04	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	POOR	PURE
dubbo-unsafestringwriter-write-ad607b0a	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
poi-littleendianbytearrayoutputstream-writebyte-ad81caf3	TRUE	TRUE	Private state	TRUE	FALSE	NONE	PURE
jmonkeyengine-androididg-glsheadsource-d663f0f8	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
jackrabbit-abstractrepository-service-checkquerystatement-f6b91938	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
lucene-solr-keepwordfilterfactory-keepwordfilterfactory-700cbb19	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
jackrabbit-searchindex-addnode-cl092973	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
poi-ssfprintsetup-setscale-25cf3d4e	TRUE	TRUE	Input Validation	TRUE	TRUE	WORD	PURE
camel-xchangeendpoint-createseconsumer-b10a9555	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	WORD	PURE
commons-text-strbuilder-replace-2694cf3f	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
jmonkeyengine-selectorkernel-initialize-b78fb095	TRUE	TRUE	Input Validation	TRUE	TRUE	WORD	PURE
accumulo-tabletresource-manager-newscanfilemanager-21775d25	TRUE	TRUE	Private state	FALSE	FALSE	NONE	PURE
jackrabbit-pathfactoryimpl-create-934c6902	TRUE	TRUE	Private state	FALSE	FALSE	NONE	PURE
camel-datalake-serviceclientwrapper-getdatalakefilesystemelementwrapper-2a22a907	FALSE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
santuario-xml-security-java-xmlsecuritystreamwriter-setnamespacecontext-1f27fa23	TRUE	TRUE	Input Validation	N/A	TRUE	NONE	PURE
activiti-eventsubscriptionqueryimpl-executionid-a7e4afaa	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE

id	wit Correctness	miner Correctness	source Checkability	wit Checkability	miner Checkability	documentation	check Effort
activiti-entitymanagersessionfactory-entitymanagersessionfactory-a148bd48	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
itext7-uniresolver-uniresolver-87437459	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
logging-log4j2-mutableinstant-initfromepochmilli-fcb067fc	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
itext7-imagedatafactory-createjpg2-163aa180	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
jackrabbit-databasefilesystem-exists-a8a390f5	TRUE	TRUE	Private state	FALSE	FALSE	NONE	PURE
lucene-solr-log4j2watcher-setloglevel-157459d9	N/A	TRUE	Input Validation	N/A	TRUE	NONE	PURE
itext7-xmputils-converttointeger-3bf1d261	N/A	TRUE	Input Validation	N/A	TRUE	NONE	INTERMEDIATE
flink-replicatinginputformat-getruntimecontext-bd66cb68	N/A	TRUE	Private state	N/A	FALSE	NONE	INTERMEDIATE

A.2 WIT Evaluation Sample

Table A.3: WIT evaluation sample (150 reports): Reported preconditions.

id	project	class	method	exception	wit Condition	mlhier Condition
lucene-solr-anonymous-1206c527-ac6d-4833-9a82-833e9b2a3a4c-objectval-b9172c5a	lucene-solr	org.apache.lucene.queries.function.valuesource.IfFunction	Anonymous-1206c527-ac6d-4833-9a82-833e9b2a3a4c	Unsupported Operation Exception	TRUE	
flink-nestedrowdata-gettimestamp-cf18ac9b	flink	org.apache.flink.table.data.binary.NestedRowData	getTimestamp(int, int)	IllegalState Exception		((precision <= 3)) && ((8 + offset + nullBitsSizeInBytes + pos * 8 <= size) && (((offset + nullBitsSizeInBytes + pos * 8 >= 0 && address + offset + nullBitsSizeInBytes + pos * 8 <= addressLimit - 8) && (address > addressLimit)))
lucene-solr-geo3dpoint-newboxquery-295faa3e	lucene-solr	org.apache.lucene.spatial3d.Geo3DPoint	newBoxQuery(java.lang.String, org.apache.lucene.spatial3d.geom.PlanetModel, double, double, double, double)	IllegalArgumentException Exception		(((((minLatitude != minLatitude) minLatitude < -90.0D minLatitude > 90.0D)) && ((minLongitude != minLongitude) minLongitude < -180.0D minLongitude > 180.0D)))
flink-defaults-lotracker-addslot-ea725cb7	flink	org.apache.flink.runtime.resourcemanager.slotmanager.DefaultSlotTracker	addSlot(org.apache.flink.runtime.clusterframework.types.SlotID, org.apache.flink.runtime.clusterframework.types.ResourceProfile, org.apache.flink.runtime.resourcemanager.registration.TaskExecutorConnection, org.apache.flink.api.common.JobID)	NullPointerException Exception		null == taskManagerConnection && null != resourceProfile && null != slotId
flink-minibatchgroupaggfunction-open-a9aab2c2	flink	org.apache.flink.runtime.operators.aggregate.MinibatchGroupAggFunction	open(org.apache.flink.runtime.context.ExecutionContext)	NullPointerException Exception		null == updateType && stateRetentionTime > 0 && ctx != null
spoon-codefactory-getreferences-abb960ud-jmonkeyengine-lwjgl-gltextureimage3d-1a2239ca	spoon	spoon.reflect.factory.CodeFactory	getReferences(java.util.List<E>)	IllegalArgumentException Exception		elements.size() < 0
		jmonkeyengine-lwjgl-gltextureimage3d-1a2239ca	glTexImage3D(int, int, int, int, int, int, java.nio.ByteBuffer)	RenderException		limit == position && data != null && limit != 0

id	project	class	method	exception	wit Condition	miner Condition
santuario-xml-security-java-xml-security-stream-065f0e34	santuario-xml-security- xml-security- java	org.apache.xml- security.stax.impl.util. ReplaceableOutput Stream	write(byte[], int, int)	throw outOf BoundsCheck FromIndexSize(offset, off, len, b. length)	((b.length off len) < 0 len > b.length - off)	
flink-nestdrowdata-getint-at1844b80	flink	org.apache.flink.table. data.binary.Nested RowData	getInt(int)	IndexOutOf BoundsException	((!(4 + offset + nullBitsSizeInBytes + pos * 8 <= size)) && (offset + nullBitsSizeInBytes + pos * 8 - offset + nullBitsSizeInBytes + pos * 8 / size * size < size - 3) && (((offset + nullBitsSizeInBytes + pos * 8 - offset + nullBitsSizeInBytes + pos * 8 / size * size >= 0 && address + offset + nullBitsSizeInBytes + pos * 8 - offset + nullBitsSizeInBytes + pos * 8 / size <= addressLimit - 4)) && ((address > addressLimit))))	
logging-log4j2-strsubstitutor-replacein-35e89893	logging- log4j2	org.apache.logging. log4j.core.lookup.Str Substitutor	replaceIn(org.apache. logging.log4j.core.Log Event, java.lang.String Buffer, int, int)	IndexOutOf BoundsException	(length > source.length() length < offset offset < 0) && null != source	
lucene-solr-termordvalcomparator-comparevalues-5b4527de	lucene-solr	org.apache.lucene. search.Field Comparator.TermOrd ValComparator	compareValues(org. apache.lucene.util.Bytes Ref, org.apache.lucene. util.BytesRef)	IllegalArgumentException Exception	((!(val1 == null) && (!(val2 == null) && (((this.offset > this.offset + this.length) && ((this.offset < 0)) && ((this.offset + this.length > this.bytes.length))) && (val2.offset > (val2.offset + val2.length))))	
jackrabbit-sessionimpl-haspermission-1e50440f	jackrabbit	org.apache.jackrabbit. core.SessionImpl	hasPermission(java.lang. String, java.lang.String)	Repository Exception	closed != null	
closure-compiler-anonymous-8a83fab3-0d18-428d-89e9-97469cf3e684-createscope-869b7c73	closure- compiler	com.google.javascript. jscomp.ScopeCreator. Anonymous-8a83fab3- 0d18-428d-89e9- 97469cf3e684	createScope(com.google. javascript.rhino.Node, com. google.javascript.jscomp. AbstractScope<?, ?>)	IllegalState Exception	TRUE	
itext7-table-addheadercell-717f0747	itext7	com.itextpdf.layout. element.Table	addHeaderCell(java.lang. String)	IllegalArgumentException Exception	columnWidths == null && header == null	
wicket-validationerror-addkey-2393aa67	wicket	org.apache.wicket. validation.Validation Error	addKey(org.apache.wicket. validation.Validator<?> java.lang.String)	IllegalArgumentException Exception	null == validator	(notNull())
commons-lang-stringutils-replaceeachrepeatedly-853dae1	commons- lang	org.apache.commons. lang3.StringUtils	replaceAllRepeatedly(java.lang.String, java.lang. String[], java.lang.String[])	IllegalArgumentException Exception	((ArrayUtils.getLength(searchList) < 0) && ((Math.max((int) ((Arrays.asList(searchList)).size() / .75f) + 1, 16) < 0)) && ((Math.max((int) ((Arrays.asList(searchList)).size() / .75f) + 1, 16) > 1 « 30)) && (0.75f <= 0 (0.75f != 0.75f))))	

id	project	class	method	exception	wit Condition	miner Condition
lucene-solr-jetty-solrnummer-new-client-d6571dae	lucene-solr	org.apache.solr.client.solrj.embedded.JettySolrRunner	newClient()	IllegalArgumentException Exception	baseSolrUrl == null	
spoon-replacementvisitor-visitannotationfieldaccess-fb4fe621	spoon	spoon.support.visitor.replace.ReplacementVisitor	visitCiAnnotationField Access(spoon.reflect.code.CiAnnotationField Access-<T> getDate(org.w3c.dom.Element, java.util.Locale)	InvalidReplace Exception	annotationFieldAccess.getTarget() == original && replace.length > 1	
xmlgraphics-fop-planrenderer-getdate-c73090ca	xmlgraphics-fop	org.apache.fop.plan.Renderer	getDate(org.w3c.dom.Element, java.util.Locale)	StringIndexOutOfBoundsException	length() < 4	
jackrabbit-oak-systempropertysupplier-create-99bd873b	jackrabbit-oak	org.apache.jackrabbit.oak.commons.properties.SystemPropertySupplier	create(java.lang.String, U)	NullPointerException Exception	null == propName	
flink-kafkasourcebuilder-setclientidprefix-756f83ae	flink	org.apache.flink.connector.kafka.source.KafkaSourceBuilder	setClientIdPrefix(java.lang.String)	NullPointerException Exception	null == prefix	
flink-binaryarraydata-setlong-5d020da4	flink	org.apache.flink.data.binary.BinaryArrayData	setLong(int, long)	IndexOutOfBoundsException	((!(segments.length == 1)) && (((!(offset + 4) + pos >> 3 + 4) + pos >> 3 + 4) + pos >> 3 - ((offset + 4) + pos >> 3 + 4) + pos >> 3 / size * size >= 0 && address + ((offset + 4) + pos >> 3 + 4) + pos >> 3 + 4) + pos >> 3 - ((offset + 4) + pos >> 3 + 4) + pos >> 3 / size * size < addressLimit)) && (!(address > addressLimit)))	
accumulo-livetserverset-getcurrentservers-aace445a	accumulo	org.apache.accumulo.server.manager.LiveTServerSet	getCurrentServers()	IllegalArgumentException Exception	((!(Math.max((int) (currentInstances.keySet().size() / .75f) + 1, 16) < 0)) && ((Math.max((int) (currentInstances.keySet().size() / .75f) + 1, 16) > 1 < 30) && (0.75f <= 0 (0.75f != 0.75f)))	
flink-defaultcatalogtable-copy-8c17e90f	flink	org.apache.flink.catalog.DefaultCatalogTable	copy(java.util.Map<java.lang.String, java.lang.String>)	NullPointerException Exception	null == schema	
commons-math-elitisticlistpopulation-elitisticlistpopulation-0b0f56e4	commons-math	org.apache.commons.math4.legacy.genetics.ElitisticListPopulation	ElitisticListPopulation(int, double)	OutOfRangeException Exception	elitismRate > 1 elitismRate < 0	elitismRate < 0 elitismRate > 1

id	project	class	method	exception	wit Condition	miner Condition
flink-bytevaluearray-addall-79c4f6d5	flink	org.apache.flink.graph.types.valuearray.ByteValueArray	addAll(org.apache.flink.graph.types.valuearray.ValueArray flink.types.ByteValue<flink.types.ByteValue>)	RuntimeException	(position + source.position > data.length) && (!(isBounded)) && ((!(position + source.position <= data.length)) && (!(position + source.position >= (data.length + (data.length > 1)))) && (!(Integer.MAX_VALUE - 8 <= (data.length + (data.length > 1)))) && ((!(currentCapacity + (currentCapacity > 1)) < position + source.position))	
jackrabbit-versioncontrolleditemcollection-checkin-ad9c4868	jackrabbit	org.apache.jackrabbit.webdav.jcr.VersionControlledItemCollection	checkin()	DavException	item == null	
flink-catalogfunctionimpl-catalogfunctionimpl-9066c51d	flink	org.apache.flink.table.catalog.CatalogFunctionImpl	CatalogFunctionImpl(java.lang.String, org.apache.flink.table.catalog.FunctionLanguage)	IllegalArgumentException Exception	StringUtils.isNullOrWhitespaceOnly(className)	
jackrabbit-oak-oakbufferedindexfile-readbytes-0d9d365c	jackrabbit-oak	org.apache.jackrabbit.oak.plugins.index.lucene.directory.OakBufferedIndexFile	readBytes(byte[], int, int)	IOException	len > length - position len < 0	
jmonkeyengine-spotlight-spotlight-93d5f1e8	jmonkeyengine-spotlight	com.jme3.light.SpotLight	SpotLight(com.jme3.math.Vector3f, float, createProducer())	IllegalArgumentException Exception	((!(int) packedAngleCos) == ((int) (outerAngleCos - 0.001f * 1000))) && (packedAngleCos == 0.0f)	
camel-hdfsendpoint-createproducer-t51ccd5b	camel	org.apache.camel.component.hdfs.HdfsEndpoint	createProducer()	IllegalArgumentException Exception	hasSplitStrategies() && append	
commons-beel-cpessc-visitor-visitconstantvalue-2daccfd5	commons-beel	org.apache.beel.verifier.statcs.Pass2Verifier.CPESSC_Visitor	visitConstantValue(org.apache.beel.classfile.ConstantValue)	ClassConstraint Exception	cplen <= obj.getNameIndex() obj.getNameIndex() < 0	
dubbo-consumermodel-consumermodel-3a0ba4da	dubbo	org.apache.dubbo.rpc.model.ConsumerModel	ConsumerModel(java.lang.String, java.lang.Object, org.apache.dubbo.rpc.model.ServiceDescriptor, org.apache.dubbo.config.ReferenceConfigBase<?>, org.apache.dubbo.rpc.model.ServiceMetadata)	IllegalArgumentException Exception	serviceKey.isEmpty() null == serviceKey	

id	project	class	method	exception	wit Condition	miner Condition
flink-taskmetricgroup-	flink	org.apache.flink.	getOrAddOperator(java.	StringIndexOutOf	operatorNameLength() > 80 && length() < 80 && null != operator	
getoraddoperator-4c9b3798		runtime.metrics.	lang.String)	BoundsException	Name	
		groups.TaskMetric				
		Group				
byte-buddy-	byte-buddy	net.bytebuddy.asm.	bindLambda(java.lang.	IllegalArgumentException	((!(TypeDescription.ForLoadedType.of(functionalInterface)).is	
withoutomapping-		Advice.WithCustom	Class<T>, java.lang.	Exception	Interface()))	
bindlambda-32a74b97		Mapping	reflect.Constructor<?>,			
			java.lang.Class<?>)			
poi-hssf-sheet-	poi	org.apache.poi.hssf.	createFreezePane(int, int,	IllegalArgumentException	(colSplit > _maxColumns - 1)	
createfreezepane-f6b6bc79		usermodel.	int, int)	Exception		
		HSSFSheet				
xmlgraphics-fop-	xmlgraphics-	org.apache.fop.	getGlyphsAvailable(int,	IndexOutOf	reverseOrder && index == indexLast - offset && index >= -offset	
glyphprocessingstate-	fop	complexscripts.fonts.	boolean, org.apache.fop.	BoundsException	&& ignoreTester != null	
getglyphsavailable-c4133842		GlyphProcessingState	complexscripts.util.Glyph			
			Tester)			
lucene-solr-resultsetimpl-	lucene-solr	org.apache.solr.client.	getType()	SQLException	closed	
gettype-a9b2805d		solr.io.sql.ResultSet				
		Impl				
wicket-data-table-	wicket	org.apache.wicket.	addTopToolbar(org.	IllegalArgumentException	null == toolbar	
addtoptoolbar-291fe8de		extensions.markup.	apache.wicket.extensions.	Exception		
		html.repeater.data.	markup.html.repeater.data.			
		table.DataTable	table.AbstractToolbar)			
camel-	camel	org.apache.camel.	asyncRequestBody(java.	IllegalState	status != 5	
defaultproducer-template-		impl.engine.Default	lang.String, java.lang.	Exception		
asyncrequestbody-afed1e32		ProducerTemplate	Object, java.lang.			
			Class<T>)			
activiti-valuedataobject-	Activiti	org.activiti.bpmn.	getType()	StringIndexOutOf	structureRef.indexOf(':') < -1	
gettype-f1a6c228		model.ValuedData		BoundsException		
		Object				
commons-math-statutils-	commons-	org.apache.commons.	geometricMean(double[])	NotPositive	values.length < 0 && null != values	
geometricmean-600fe704	math	math4.legacy.stat.Stat		Exception		
		Utils				
xmlgraphics-fop-	xmlgraphics-	org.apache.fop.pdf.	output(java.io.Output	IllegalArgumentException	((!(this.document != null)) && (this.parent != null)) && (get	
pdimageobject-output-	fop	PDFImageXObject	Stream)	Exception	Document().getProfile().isPDFV7Active()) && (32 < 0)	
09052584						

id	project	class	method	exception	wit Condition	miner Condition
flink-normalizedkeyutil-putcharnormalizedkey-c83a41b8	flink	org.apache.flink.api.common.typeutils.base.NormalizedKeyUtil	putCharNormalizedKey(char, org.apache.flink.core.memory.MemorySegment, int, int)	IndexOutOfBoundsException	numBytes == 1 && address <= addressLimit && (address >= addressLimit - offset offset < 0)	
wicket-anonymous-ec2adc75-2ec9-4a68-82f8-10b4642922bb-next-6b603b0b	wicket	org.apache.wicket.markup.MarkupFragment	Anonymous-ec2adc75-2ec9-4a68-82f8-10b4642922bb	IndexOutOfBoundsException	((!(index++ < 0) (index++ > size))) && ((!(index++ < 0) (index++ > size)))	
poi-packagingurihelper-resolveparturi-92f40a2b	poi	org.apache.poi.openxml4j.opc.PackagingURIHelper	resolvePartUri(java.net.URI, java.net.URI)	IllegalArgumentException Exception	null == sourcePartUri null != scheme	
commons-math-abstractfieldintegrator-geteventhandlers-f5de1212	commons-math	org.apache.commons.math4.legacy ode.AbstractFieldIntegrator	getEventHandlers()	IllegalArgumentException Exception	eventsStates.size() < 0	
jfreechart-xydrawableannotation-xydrawableannotation-788685e8	jfreechart	org.jfree.chart.annotations.XYDrawable	XYDrawableAnnotatoin(double, double, double, double, org.jfree.chart.Drawable)	IllegalArgumentException Exception	((!(drawable == null) && ((!(x <= 0.0D) && ((!(x <= Double.MAX_VALUE)))	(nullNotPermitted())
closure-compiler-codeprintertest-testtpeannotationsmemberstub-c8785652	closure-compiler	com.google.javascript.jscomp.CodePrinter	testTypeAnnotations(MemberStub)	IllegalStateException	null == root	
jackrabbit-nodeimpl-cancelmerge-15e039e2	jackrabbit	org.apache.jackrabbit.jcr2spi.NodeImpl	cancelMerge(javax.jcr.version.Version)	RepositoryException	lalive	
commons-lang-failablestream-foreach-fc5485d2	commons-lang	org.apache.commons.lang3.Streams	forEach(org.apache.commons.lang3.Functions.Stream, FailableConsumer<O, ?>)	IllegalStateException	terminated	
asm-type-tostring-9564e369	asm	org.objectweb.asm.Type	toString()	StringIndexOutOfBoundsException	sort == 10 && (valueBegin - valueEnd > 2 valueBegin < 1 length() - valueEnd < 1)	
wicket-buttonresource-respond-cc0328c5	wicket	org.apache.wicket.examples.hangman.Leifer.ButtonResource	respond(org.apache.wicket.request.resource.Resource Attributes)	IllegalArgumentException Exception	PARAMETER_GUESSED == null	
accumulo-standaloneclustercontrol-suspend-c4034a18	accumulo	org.apache.accumulo.cluster.standalone.StandaloneClusterControl	suspend(org.apache.accumulo.minicluster.ServerType, java.lang.String)	NullPointerException Exception	null == options	

id	project	class	method	exception	wit Condition	miner Condition
flink-sourceoperator-sourceoperator-2f2a6c31	flink	org.apache.flink.streaming.api.operators.SourceOperator	SourceOperator(org.apache.flink.util.function.FunctionWithException<org.apache.flink.api.connector.source.SourceReaderContext, org.apache.flink.api.connector.source.SourceReader<OUT, SplitT>, java.lang.Exception>, org.apache.flink.runtime.operators.coordination.OperatorEventGateway, org.apache.flink.core.io.SimpleVersionedSerializer<SplitT>, org.apache.flink.api.common.eventtime.WatermarkStrategy<OUT>, org.apache.flink.streaming.runtime.tasks.ProcessingTimeService, org.apache.flink.configuration.Configuration, java.lang.String, boolean)	NullPointerException Exception	null == watermarkStrategy && null != operatorEventGateway && null != readerFactory && null != splitSerializer	
flink-rmatgraph-setconstants-da08cef8	flink	org.apache.flink.graph.generator.RMatGraph	setConstants(float, float, float)	IllegalArgumentException Exception	((!(a >= 0.0f && b >= 0.0f && c >= 0.0f && a + b + c <= 1.0f)))	
commons-math-diagonalmatrix-add-c991ba85	commons-math	org.apache.commons.math4.legacy.linear.DiagonalMatrix	add(org.apache.commons.math4.legacy.linear.DiagonalMatrix)	MatrixDimensionMismatch Exception	loanAdd(m)	
lucene-solr-fieldinfo-setomitnorms-2d77007a	lucene-solr	org.apache.lucene.index.FieldInfo	setOmitsNorms()	IllegalStateException Exception	((!(indexOptions == IndexOptions.NONE) && (!(indexOptions != IndexOptions.NONE) && !(storeTermVector)) && (storePayloads)))	
flink-objecthashset-objecthashset-a885555a	flink	org.apache.flink.table.runtime.util.collections.ObjectHashSet	ObjectHashSet(int, float)	IllegalArgumentException Exception	f <= 0 && expected > 1073741824	

id	project	class	method	exception	wit Condition	miner Condition
camel-objectnode-addchildren-67564d26	camel	org.apache.camel.component.salesforce.api.dto.composite.ObjectNode	addChildren(java.lang.String, org.apache.camel.component.salesforce.api.dto.AbstractObjectBase, org.apache.camel.component.salesforce.api.dto.AbstractObjectBase...)	IllegalArgumentException Exception	null == others && first != null && labelPlural != null	
accumulo-remoteshell-remoteshell-4237af44	accumulo	org.apache.accumulo.cluster.RemoteShell	RemoteShell(java.lang.String, java.lang.String[], java.io.File, java.util.Map<java.lang.String, java.lang.String>, long, org.apache.accumulo.cluster.RemoteShellOptions)	NullPointerException Exception	null == options	
jmonkeyengine-hdrenderer-reshape-99d512ea	jmonkeyengine-hdrenderer-reshape-99d512ea	HDRenderer	reshape(com.jme3.renderer.ViewPort, int, int)	IllegalArgumentException Exception	(h <= 0 w <= 0) && mainSceneFB != null	
wildfly-elytron-identitymappingbuilder-build-0296c485	wildfly-elytron-identitymappingbuilder-build-0296c485	org.wildfly.security.auth.realm.IdapLdapSecurityRealmBuilder	build()	throw log.builderAlreadyBuilt();	built	
commons-configuration-configurationpropertiesfactory-configurationpropertiesfactory-24590a1c	commons-configuration-configurationpropertiesfactory-configurationpropertiesfactory-24590a1c	org.apache.commons.configuration2.spring.ConfigurationPropertiesFactory	afterPropertiesSet()	IllegalArgumentException Exception	compositeConfiguration == null && configurations.length == 0 && locations.length == 0 && configurations != null	
flink-abstractresource-manager-driver-initialize-13ad4cb5	flink-abstractresource-manager-driver-initialize-13ad4cb5	org.apache.flink.runtime.resourcemanager.active.AbstractResourceManagerDriver	initialize(org.apache.flink.runtime.resourcemanager.active.ResourceEventHandler<WorkerType>, org.apache.flink.util.concurrent.ScheduledExecutor, java.util.concurrent.Executor)	NullPointerException Exception	null == resourceEventHandler	
camel-staxjxpathiterator-expression-staxjxpathiterator-expression-c175ce893	camel	org.apache.camel.component.stax.StAXJXPathIteratorExpression	StAXJXPathIteratorExpression(java.lang.Class<T>, boolean)	IllegalArgumentException Exception	handled == null	(notNull())

id	project	class	method	exception	wit Condition	miner Condition
flink- pythonkeyedcoprocessoperator- onprocessingtime-f75d94e0	flink	org.apache.flink. streaming.api. operators.python. PythonKeyedCo ProcessOperator	onProcessingTime(org. apache.flink.streaming.api. operators.internal Timer<org.apache.flink. types.Row, org.apache. flink.runtime.state.Void Namespaces>)	IllegalArgumentException Exception	((TimeDomain.PROCESSING_TIME == TimeDomain. PROCESSING_TIME) && (fieldByPosition != null) && (fieldBy Position != null) && (fieldByPosition != null) && (fieldByPosition != null) && ((fieldByPosition != null)))	
camel- defaultpgppublickeyaccessor- defaultpgppublickeyaccessor- 26036ff09	camel	org.apache.camel. converter.crypto. DefaultGPPublicKey Accessor	DefaultGPPublicKey Accessor(byte[])	IllegalArgumentException Exception	null == publicKeyRing	(notNull())
h2database-simpleresultset- getcolumnType-50ae0e0c	h2database	org.h2.tools.Simple ResultSet	getColumnType(int)	throw outOf BoundsCheck Index(outOf, (	columnIndex >= 1 && columnIndex - size >= 1 && columnIndex <= size	
jackrabbit-orderhandler- value-c603c8f1	jackrabbit	org.apache.jackrabbit. commons.jackrabbit. user.Authorizable QueryManager.Query Translator.Order Handler	value(ava.lang.String)	IOException size)	((("property".equals(currentKey))) && ((("direction".equals(current Key))) && ((("ignoreCase".equals(currentKey))))	
joda-time-years- yearsbetween-54126dff	joda-time	org.joda.time.Years	yearsBetween(org.joda. time.ReadableInstant, org. joda.time.ReadableInstant)	IllegalArgumentException Exception	end == null null == start	
flink-restclientconfiguration- fromconfiguration-17019c10	flink	org.apache.flink. runtime.rest.RestClient Configuration	fromConfiguration(org. apache.flink.configuration. Configuration)	NullPointerException Exception	config == null	(checkNotNull())
camel-objectnode- addchildren-2b6b2846	camel	org.apache.camel. component.salesforce. api.dto.composite. ObjectNode	addChildren(org.apache. camel.component. salesforce.api.dto.Abstract DescribedObjectBase, org.apache.camel. component.salesforce.api. dto.AbstractDescribed ObjectBase...)	IllegalArgumentException Exception	null == others && first != null	

id	project	class	method	exception	wit Condition	miner Condition
lucene-solr-anonymous-57647235-4b0a-465d-98fe-28598109e2c7-add-754b09ee	lucene-solr	org.apache.solr.common.util.JavaBinCodec Anonymous-57647235-4b0a-465d-98fe-28598109e2c7	add(long)	throw outOfBoundsCheck FromIndexSizeOutOfBoundsException, offset, buf.length, buf.length	((!(v & 0xff00000000000000L) == 0) && (!(v >= 0x0f)) && ((pos >= buf.length) && ((buf.length 0 buf.length) < 0 buf.length > buf.length - 0)))	
lucene-solr-conjunctionspans-positions-cost-02d4cf08e	lucene-solr	org.apache.lucene.search.spans.ConjunctionSpans	positionsCost()	UnsupportedOperationException	TRUE	
asterisk-java-listidplanevent-setpriority-214db761	asterisk-java	org.asteriskjava.manager.event.ListDiaplanEvent	setPriority(java.lang.String)	throw NumberFormatException, forInputString(priorityString)	((!priorityString == null) && (!PRIORITY_HINT.equals(priorityString)) && ((!priorityString == null) && ((10 < 2) && (!(10 > 36)) && (!(len > 0))))	
swingx-treetablehacker-hithandledetectionfrom-jx-tree-table-edit-cell-20330a25	swingx	org.jdesktop.swing.XJTreeTable Hacker	hitHandledDetectionFromEditCell(int, java.util.EventObject)	IllegalArgumentException	!(false) && (column < 0 column >= getColumnCount())	
commons-configuration-propertiesbuilderparametersimplinheritfrom-6b2fa101	commons-configuration	org.apache.commons.configuration2.builder.PropertiesBuilder ParametersImpl	inheritFrom(java.util.Map<java.lang.String, ?>)	IllegalArgumentException	null == source	
jfreechart-meterinterval-meterinterval-eb69023	jfreechart	org.jfree.chart.plot.MeterInterval	MeterInterval(java.lang.String, org.jfree.data.Range)	IllegalArgumentException	label == null	
flink-embeddedexecutorfactory-embeddedexecutorfactory-71a9c5f8	flink	org.apache.flink.client.deployment.application.executor.EmbeddedExecutorFactory	EmbeddedExecutorFactory(java.util.Collection<org.apache.flink.api.common.JobID>, org.apache.flink.runtime.dispatcher.Dispatcher Gateway, org.apache.flink.util.concurrent.ScheduledExecutor)	NullPointerException	null == submittedJobIds	

id	project	class	method	exception	wit Condition	miner Condition
flink-declarativeslotmanager-start-499cbe8	flink	org.apache.flink.runtime.resourceManager.slotmanager.DeclarativeSlotManager	start(org.apache.flink.runtime.resourceManager.slotmanager.DeclarativeSlotManager)	NullPointerException	newResourceManagerId == null	
jackrabbit-workspaceoperationstestcopy-99169b55	jackrabbit	org.apache.jackrabbit.test.api.observation.WorkspaceOperationTest	testCopy()	IllegalStateException	null == obsMgr	
lucene-solr-jasperlternarysearchtrie-82da027d	lucene-solr	org.apache.lucene.search.suggest.jasperlternarysearchtrie.JasperlternarySearchTrie	JasperlternarySearchTrie(java.nio.file.Path, boolean)	IllegalArgumentException	compression && defaultCharBufferSize <= 0	
poi-ole10native-ole10native-d212018a	poi	org.apache.poi.poifs.filesystem.Ole10Native	Ole10Native(byte[], int)	RuntimeException	(LittleEndianConsts.INT_SIZE > count - pos)	
flink-filesystemblobstore-filesystemblobstore-cca566f1	flink	org.apache.flink.runtime.blob.FileSystemBlobStore	FileSystemBlobStore(org.apache.flink.core.fs.FileSystem, java.lang.String)	NullPointerException	fileSystem == null	
flink-kinesisdynamic-sink-copy-a3f29c3f	flink	org.apache.flink.streaming.connectors.kinesis.table.KinesisDynamicSink	copy()	NullPointerException	encodingFormat == null && consumedDataType != null && null != producerProperties && null != stream	
camel-skiplastbyteinputstream-read-d74f21d7	camel	org.apache.camel.util.SkipLastByteInput	read(byte[], int, int)	IndexOutOfBoundsException	((!(buf == null)) && ((off len (off + len) (buffer.length - (off + len))) < 0))	
flink-closingfsdataoutputstream-wrapsafe-61d28987	flink	org.apache.flink.core.fs.ClosingFSDaOutputStream	wrapSafe(org.apache.flink.core.fs.FSDaOutputStream, org.apache.flink.core.fs.SafetyNetCloseableRegistry)	NullPointerException	null == registry	
flink-bufferwritingresultpartition-setup-4b5ca08a	flink	org.apache.flink.runtime.io.network.partition.BufferWritingResultPartition	setup()	IllegalStateException	((!(this.bufferPool == null))	

id	project	class	method	exception	wit Condition	miner Condition
jackrabbit-oak-anonymous-06d24382-5fb9-4402-9f63-oak	jackrabbit-oak	org.apache.jackrabbit.oak.segment.Segment	readSegment(org.apache.jackrabbit.oak.segment.SegmentId)	SegmentNotFound Exception	TRUE	
ca3b6ed67fcl-readsegment-40e90863	Store.Anonymous-06d24382-5fb9-4402-9f63-ca3b6ed67fcl	org.apache.lucene.document.XYShape	newBoxQuery(java.lang.String, org.apache.lucene.document.ShapeField, QueryRelation, float, float, float, float)	IllegalArgumentException Exception	((!(minX > maxX)) && (!(minY > maxY)) && (!(minX <= 0.0F)) && (minX <= Float.MAX_VALUE == false))	
lucene-solr-xyshape-newboxquery-66aa8600	lucene-solr	org.apache.lucene.document.XYShape	newBoxQuery(java.lang.String, org.apache.lucene.document.ShapeField, QueryRelation, float, float, float, float)	IllegalArgumentException Exception	((!(minX > maxX)) && (!(minY > maxY)) && (!(minX <= 0.0F)) && (minX <= Float.MAX_VALUE == false))	
flink-bufferbuilder-append-c128x0c2	flink	org.apache.flink.runtime.io.network.buffer.BufferBuilder	append(java.nio.ByteBuffer)	IndexOutOfBoundsException	((!(isFinished())) && (cachedPosition == FINISHED_EMPTY) && (limit - position <= maxCapacity - 0) && ((positionMarker.getCached() needed (positionMarker.getCached() + needed)) < 0))	
commons-math-baseabstractunivariate-solver-solve-3a277d17	commons-math	org.apache.commons.math4.legacy.analysis.solvers.BaseAbstractUnivariateSolver	solve(int, FUNC, double, double)	NullPointerException Exception	f == null	
swingx-componenttree-tablemodel-convertvaluetotext-l d39175e	swingx	util.ComponentTreeTableModel	convertValueToText(java.lang.Object)	StringIndexOutOfBoundsException	lastIndexOf(".", length()) >= -1 && lastIndexOf(".", length()) - length() > -1 && name != null	
logging-log4j2-log4jmarker-remove-cfa42a2e	logging-log4j2	org.apache.logging.log4j.MarkerManager	remove(org.apache.logging.log4j.Marker)	IllegalArgumentException Exception	((parent == null) && ((this.parents == null) && (this.parents.length == 1) && (parent == null)))	
flink-resource-profile-fromresources-10662a10	flink	org.apache.flink.runtime.clusterframework.types.ResourceProfile	fromResources(double, int)	IllegalArgumentException Exception	(((((taskHeapMemoryMB « 20) >= 0)))	
accumulo-kerberosrenewalit-stopkdc-dfaab8c47	accumulo	org.apache.accumulo.test.functional.KerberosRenewalIT	stopKdc()	NullPointerException Exception	(kdc != null) && (krbEnabledForTts != null) && (MiniClusterHarness.USE_KERBEROS_FOR_IT_OPTION == null)	
lucene-solr-typostingsenum-getpayload-574001d1	lucene-solr	org.apache.lucene.backward_codecs.lucene50.compressing.Lucene50.CompressingTermVectorsReader	getPayload()	IllegalStateException Exception	doc == -1 && Integer.MAX_VALUE != -1	

id	project	class	method	exception	wit Condition	miner Condition
jfreechart- defaultkeyvalues- insertvalue-3919be65	jfreechart	org.jfree.data.Default KeyedValues	insertValue(int, K, java. lang.Number)	IllegalArgumentException Exception	getItemCount() < position position < 0	
commons-text-strsubstitutor- replace-d50ecd1f	commons- text	org.apache.commons. text.StrSubstitutor	replace(char[], int, int)	StringIndexOutOfBoundsException BoundsException	(offset > source.length offset < 0) && null != source	
poi-spmutils-getgrpprptl- 68aue5ed	poi	org.apache.poi.hwpf. spm.SpmUtils	getGrpprpt(java.util. List<byte[]>, int)	RecordFormatException Exception	((!(size < 0L)) && (size > (long) Integer.MAX_VALUE))	
flink-builder-column- 1c2b3d35	flink	org.apache.flink.table. api.Schema.Builder	column(java.lang.String, org.apache.flink.table. types.AbstractData Type<?>)	NullPointerException Exception	dataType == null && columnName != null	
lucene-solr-restore- getparams-cfc2b790	lucene-solr	org.apache.solr.client. solr.request.Collection AdminRequest. Restore	getParams()	SolrException	followAliases != null && nrReplicas != null && null != replication Factor	
jackrabbit-derbyfilesystem- close-a13db187	jackrabbit	org.apache.jackrabbit. core.fs.db.DerbyFile System	close()	IllegalStateException Exception	initialized	
logging-log4j2- eventresolverinterceptors- populateinterceptors- f455bece	logging- log4j2	org.apache.logging. log4j.layout.template. json.resolver.Event ResolverInterceptors	populateInterceptors(java. util.List<java.lang.String>)	NullPointerException Exception	((!(TemplateResolverInterceptor.CATEGORY == null)) && (plugin Packages == null))	
byte-buddy- withcustommapping- bindlambda-7ef0c638	byte-buddy	net.bytebuddy.asm. Advice.WithCustom Mapping	bindLambda(java.lang. Class<T>, java.lang. reflect.Method, java.lang. Class<?>, net.bytebuddy. dynamic.scaffold.Method Graph.Compiler)	IllegalArgumentException Exception	((!(TypeDescription.ForLoadedType.of(functionalInterface)).is Interface()))	
xmlgraphics-fop- pdfcidssysteminfo- topdfstring-f8ea215d	xmlgraphics- fop	org.apache.fop.pdf. PDFCIDSystemInfo	toPDFString()	IllegalArgumentException Exception	((!(0 < 0)) && ((0 - value.length » coder > 0) && (count < 0) && (COMPACT_STRINGS && coder == LATIN1) && (count > 0)))	
lucene-solr- blockpackedReaderiterator- blockpackedReaderiterator- d5c3bc08	lucene-solr	org.apache.lucene.util. BlockPackedReader PackedBlockPacked ReaderIterator	BlockPackedReader Iterator(org.apache.lucene. store.DataInput, int, int, long)	IllegalArgumentException Exception	MIN_BLOCK_SIZE > blockSize MAX_BLOCK_SIZE < block Size	

id	project	class	method	exception	wit Condition	miner Condition
flink-streamtableenvironmentimpl-fromchangelogstream-eb4d22de	flink	org.apache.flink.table.api.bridge.java.internal.StreamTableEnvironmentImpl	fromChangelogStream(org.apache.flink.streaming.api.datastream.DataStream<org.apache.flink.types.Row>, org.apache.flink.table.api.Schema)	NullPointerException	dataStream == null && null != schema	
poi-ptg-readtokens-8920e99d	poi	org.apache.poi.ss.formula.ptg.Ptg	readTokens(int, org.apache.poi.util.LittleEndianInput)	IllegalArgumentException	size < -8	
flink-intvaluearray-add-7dc2ae39	flink	org.apache.flink.graph.types.valuearray.IntValueArray	add(org.apache.flink.graph.add(IntValue)	RuntimeException	(position + 1 > data.length) && ((!(isBounded)) && ((!(position + 1 <= data.length) && ((!(position + 1 >= (data.length + (data.length » 1)))) && ((Integer.MAX_VALUE - 8 <= (data.length + (data.length » 1)))) && ((int) (currentCapacity + (currentCapacity » 1)) < position + 1))	
flink-binarymapdata-pointto-1f68be4c	flink	org.apache.flink.table.data.binary.BinaryMapData	pointTo(org.apache.flink.core.memory.MemorySegment[], int, int)	IllegalStateException	address > addressLimit && size > 3 && address - addressLimit > -4 && offset - size > -4	
lucene-solr-geosouthrectangle-geosouthrectangle-41f9d437	lucene-solr	org.apache.lucene.spatial3d.geom.GeoSouthRectangle	GeoSouthRectangle(org.apache.lucene.spatial3d.geom.PlanetModel, java.io.InputStream)	IllegalArgumentException	SerializableObject.readDouble(inputStream) >= -1.	
flink-rowtypeinfo-gettypeat-40794da5	flink	org.apache.flink.api.java.typeutils.RowTypeInfo	getTypeAt(java.lang.String)	IllegalStateException	5707963267948966 && SerializableObject.readDouble(inputStream) <= 1.5707963267948966 && tabs(SerializableObject.readDouble(inputStream)) > 3.1415926535897931	
itext7-pdfutils-spec-createembeddedfilespec-d50e9f01	itext7	com.itextpdf.kernel.pdf.filespec.PdfFileSpec	createEmbeddedFileSpec(com.itextpdf.kernel.pdf.Document, byte[], java.lang.String, java.lang.String, com.itextpdf.kernel.pdf.PdfName)	Exception	((compiled) && ((!(0 < groups.length) && ((!(0 < localsPos.length) && ((!(parentPattern.matchRoot.match(this, from, text))) && ((!(parentPattern.matchRoot.match(this, from, text))) && (first < 0)	
jmonkeyengine-lwjgl-gltex-glbuffer-subdata-d0ba7cd9	jmonkeyengine	com.jmonkeyengine.lwjgl.gltex.glbuffer.SubData	glBufferSubData(int, long, java.nio.IntBuffer)	RenderException	limit == position && data != null && limit != 0	
camel-joorlanguage-createexpression-54a09838	camel	org.apache.camel.language.joor.Language	createExpression(java.lang.String)	StringIndexOutOfBoundsException	preCompile	

id	project	class	method	exception	wit Condition	miner Condition
flink-binaryarraydata-getboolean-37b7d725	flink	org.apache.flink.table.data.binary.BinaryArrayData	getBoolean(int)	IndexOutOfBoundsException	address <= addressLimit && elementOffset + pos - size <= -1 && (address >= addressLimit - elementOffset - pos elementOffset < -pos)	
flink-statefulsequencesource-initializestate-f02af46c	flink	org.apache.flink.streaming.api.functions.source.StatefulSequenceSource	initializeState(org.apache.flink.runtime.state.FunctionInitializationContext)	IllegalStateException	((!(this.checkpointedState == null))) && ((this.checkpointedState == null))	
dubbo-ioutils-geturl-a069baf6	dubbo	org.apache.dubbo.common.utils.IOUtils	getURL(java.lang.String)	IllegalArgumentException	null == resourceLocation	(notNull())
closure-compiler-anonymous-fac23e1e-b74f-4b00-8c1e-fd8d6b6cc8f8-enterscope-9d310c1b	closure-compiler	com.google.javascript.jscomp.NodeUtil	enterScope(com.google.javascript.jscomp.NodeTraversal)	Exception	TRUE	
jmonkeyengine-lwjgl-glunifommatrix4-7a2cf860	jmonkeyengine	lwjgl.GLUniformMatrix4f	glUniformMatrix4f(int, boolean, java.nio.FloatBuffer)	RendererException	limit == 0 && null != value	
lucene-solr-pathhierarchytokenizer-8a995940	lucene-solr	org.apache.lucene.analysis.path.HierarchyTokenizer	PathHierarchyTokenizer(int)	IllegalArgumentException	skip < 0	
jackrabbit-impersonationimpl-getimpersonators-40d3a066	jackrabbit	org.apache.jackrabbit.core.security.user.ImpersonationImpl	getImpersonators()	InvalidItemStateException	(status == 2 status == 3)	
flink-kubernetesleaderretrievaldriver-createleaderretrievaldriver-17657528	flink	org.apache.flink.kubernetes.KubernetesLeaderRetrievalDriverFactory	createLeaderRetrievalDriver(org.apache.flink.runtime.leaderretrieval.KubernetesLeaderRetrievalHandler, org.apache.flink.runtime.rpc.FatalErrorHandler)	NullPointerException	leaderEventHandler == null && configMapName != null && kubeClient != null	
commons-configuration-handler-setfilelocator-d92d1a53	commons-configuration	org.apache.commons.configuration2.io.FileLocator	setFileLocator(final FileLocator)	IllegalArgumentException	locator == null	locator == null
jmonkeyengine-mesh-gettrianglecount-1c172e23	jmonkeyengine	jmonkeyengine.com.jme3.scene.Mesh	getTriangleCount(int)	IllegalArgumentException	lodLevels == null && lod != 0	lod != 0 && lodLevels == null

id	project	class	method	exception	wit Condition	miner Condition
camel-timerendpoint-createproducer-ff00d	camel	org.apache.camel.component.timer.TimerEndpoint	createProducer()	RuntimeException	TRUE	TRUE
activiti-timerjobqueryimpl-messages-8b11f98	Activiti	org.activiti.engine.impl.TimerJobQueryImpl	messages()	ActivitiIllegalArgument Exception	onlyTimers	onlyTimers
flink-equiujoin-equiujoin-919249c	flink	org.apache.flink.api.java.operators.JoinOperator	EquiJoin(DataSet<I>, DataSet<I2>, Keys<I1>, Keys<I2>, FlatJoinFunction<I1, I2, OUT>, JoinFunction<I1, I2, OUT>, TypeInformation<OUT>, JoinHint, String, JoinType)	NullPointerException Exception	function == null	function == null
commons-lang-objectutils-const-short-7ef1a952	commons-lang	org.apache.commons.lang3.ObjectUtils	CONST_SHORT(final int)	IllegalArgumentException Exception	Short.MIN_VALUE > v Short.MAX_VALUE < v	v < Short.MIN_VALUE v > Short.MAX_VALUE
spoon-substitutor-visitmethod-c4fde28d	spoon	spoon.smp1.Substitutor	visitCMethod(CtMethod<T>)	NotImplemented Exception	TRUE	TRUE
commons-bcel-pass3bverifier-invalidreturntypeerror-a6c8136c	commons-bcel	org.apache.bcel.verifier.structurals.Pass3bVerifier	invalidReturnTypeError(final Type, final Method Gen)	StructuralCodeConstraint Exception	TRUE	TRUE
flink-fscheckpointstreamfactory-fscheckpointstreamfactory-a29af122	flink	org.apache.flink.runtime.state.filesystem.FsCheckpointStreamFactory	FsCheckpointStreamFactory(FileSystem, Path, Path, int, int)	IllegalArgumentException Exception	fileStateSizeThreshold < 0	fileStateSizeThreshold < 0
dubbo-javabeandescriptor-getprimitiveproperty-f8807c3d	dubbo	org.apache.dubbo.common.beanutil.JavaBeanDescriptor	getPrimitiveProperty()	IllegalStateException Exception	type != 6	!(isPrimitiveType())
spoon-patternbuilder-visitannotation-85b05c5f	spoon	spoon.smp1.pattern.PatternBuilder	visitCtAnnotation(CtAnnotation<A>)	NotImplemented Exception	TRUE	TRUE
lucene-solr-stopfilterfactory-stopfilterfactory-fce70556	lucene-solr	org.apache.lucene.analysis.core.StopFilterFactory	StopFilterFactory()	defaultCtor Exception()	TRUE	TRUE
jackrabbit-versioncontroladsourcesimpl-update-c37b25c2	jackrabbit	org.apache.jackrabbit.webdav.simple.VersionControlledResourceImpl	update(UpdateInfo)	DavException	TRUE	TRUE

id	project	class	method	exception	wit Condition	miner Condition
spoon-patternbuilder- visitcannotationtype- 06458832	spoon	spoon.smpl.pattern. PatternBuilder	visitCAnnotationType(Ct AnnotationType<A>)	NotImplemented Exception	TRUE	TRUE
camel-readerinputstream- read-0427b4a9	camel	org.apache.camel. component.cxf.util. ReaderInputStream	read(byte[], int, int)	IndexOutOf BoundsException	(b.length < len + off len < 0 off < 0) && b != null	len < 0 off < 0 len > b.length && b != null
lucene-solr-querycommand- setfilterlist-eebc7f0e	lucene-solr	org.apache.solr. search.Query Command	setFilterList(List<Query>)	IllegalArgument Exception	filter != null	filter != null
jackrabbit-abstractsession- setnamespaceprefix- 10a5c314	jackrabbit	org.apache.jackrabbit. commons.Abstract Session	setNamespacePrefix(String, String)	IllegalArgument Exception	null == prefix	prefix == null
lucene-solr- doubledocvaluesfield- setlongvalue-72495416	lucene-solr	org.apache.lucene. document.DoubleDoc ValuesField	setLongValue(long)	IllegalArgument Exception	TRUE	TRUE
poi-stylestable- setmaxnumberofdataformats- e060f647	poi	org.apache.poi.xssf. model.StylesTable	setMaxNumberOfData Formats(int)	IllegalArgument Exception	getNumDataFormats() > num && num < 0	num < 0 && num < getNumDataFormats()
jackson-databind- beanserializertbuilder- settypeid-666d1cf	jackson- databind	com.fasterxml.jackson. databind.ser.Bean SerializerBuilder	setTypeId(Annotated Member)	IllegalArgument Exception	_typeid != null	_typeid != null
flink-genericdatasourcebase- genericdatasourcebase- a6971365	flink	org.apache.flink.api. common.operators. GenericDataSource Base	GenericDataSourceBase(T, Operator Information<OUT>)	IllegalArgument Exception	format == null	format == null
jackrabbit-staleproperty- issame-f386f2c2	jackrabbit	org.apache.jackrabbit. jcr2spi.StateProperty	isSame(Item)	createISE()	TRUE	TRUE
lucene-solr- uniformdistributionevaluator- dowork-fda7bf5	lucene-solr	org.apache.solr.client. solr.io.eval.Uniform DistributionEvaluator	doWork(Object, Object)	IOException	first == null	null == first
flink-genericinputformat- createinputplits-c2be0403	flink	org.apache.flink.api. common.io.Generic InputFormat	createInputSplits(int)	IllegalArgument Exception	numSplits < 1	numSplits < 1
commons-lang-strokenizer- add-0671cddb	commons- lang	org.apache.commons. lang3.text.StrTokenizer	add(final String)	Unsupported Operation Exception	TRUE	TRUE

Table A.4: WIT evaluation sample (150 reports): Manual-review labels.

id	wit Correctness	miner Correctness	source Checkability	wit Checkability	miner Checkability	documentation	check Effort
lucene-solr-anonymous-1206c527-ac6d-4833-9a82-833e9b2a3a4c-objectval-b9177a5a	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
flink-nestedrowdata-gettimestamp-cf18ae9b	FALSE	N/A	Hybrid Untestable	N/A	N/A	NONE	COMPLEX
lucene-solr-geo3dpoint-newboxquery-295faa3e	FALSE	N/A	N/A	N/A	N/A	POOR	COMPLEX
flink-defaultslottracker-addstor-ea725cb7	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
flink-minibatchgroupaggfunction-open-a9ab2c2	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
spoon-codefactory-getreferences-abd960ad	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
jmonkeyengine-lwjgl-glteximage3d-1a2239ea	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
santuario-xml-security-java-replaceableoutputstream-write-065f0c34	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-nestedrowdata-getint-a1844b80	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
logging-log4j2-strsubstitutor-replacein-35e89893	FALSE	N/A	Hybrid Untestable	N/A	N/A	NONE	COMPLEX
lucene-solr-termordvalcomparator-comparevalues-5b527de	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
jackrabbit-sessionimpl-haspermission-1e50d40f	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
closure-compiler-anonymous-8a83fab3-0d18-428d-89e9-97469cf3e684-createscope-869b7c73	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
itext7-table-addheadercell-7f70747	TRUE	N/A	Private state	FALSE	N/A	POOR	COMPLEX
wicket-validationerror-addkey-2393aa67	TRUE	TRUE	Input Validation	FALSE	TRUE	NONE	INTERMEDIATE
commons-lang-stringutils-replaceeachrepeatedly-853dae1	FALSE	TRUE	N/A	N/A	N/A	WORD	COMPLEX
lucene-solr-jetty-solrrunner-newclient-d6571dae	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
spoon-replacemementvisitor-visittanotationofidaccess-fb4fe621	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
xmlgraphics-fop-planrenderer-getdate-c73090ca	FALSE	N/A	Input Validation	N/A	N/A	NONE	COMPLEX
jackrabbit-oak-systempropertysupplier-create-99bd873b	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-kafkasourcebuilder-setclientidprefix-756783ae	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-binaryarraydata-setlong-5d020da4	FALSE	N/A	Input Validation	N/A	N/A	NONE	INTERMEDIATE
accumulo-livetserverset-getcurrentservers-aace445a	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-defaultcatalogable-copy-8c17e90f	FALSE	N/A	Input Validation	N/A	N/A	NONE	COMPLEX
commons-math-ellipticlistpopulation-ellipticlistpopulation-0b0f56e4	TRUE	TRUE	Hybrid Untestable	FALSE	FALSE	VAR	INTERMEDIATE
flink-bytevaluearray-addall-79e4f6d5	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
jackrabbit-versioncontrolledmetacollection-checkin-ad9c4868	FALSE	N/A	Private state	N/A	N/A	NONE	COMPLEX
flink-catalogfunctionimpl-catalogfunctionimpl-9066c51d	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
jackrabbit-oak-oakbufferedindexfile-readbytes-0d9d365c	FALSE	N/A	Hybrid Untestable	N/A	N/A	NONE	INTERMEDIATE
jmonkeyengine-spotlight-spotlight-93d5f1e8	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
camel-hdfsendpoint-createproducer-b51cc45b	FALSE	N/A	Hybrid Untestable	N/A	N/A	POOR	PURE
commons-beel-epesc-visitor-visitconstantvalue-2dacefd5	FALSE	N/A	N/A	N/A	N/A	NONE	COMPLEX
dubbo-consumermodel-consumermodel-3a0ba4da	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
flink-taskmetricgroup-getoraddoperator-4c9b3798	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX

id	wit Correctness	miner Correctness	source Checkability	wit Checkability	miner Checkability	documentation	check Effort
byte-buddy-withcustommapping-bindlambdab-32a74b97	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
poi-hs1sheet-createfreezepane-d6b6bc79	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
xmlgraphics-fop-glyphprocsingsstate-getglyphsavailable-c4133842	FALSE	TRUE	N/A	N/A	N/A	WORD	COMPLEX
lucene-solr-resultsetimpl-gettype-a9b2805d	TRUE	N/A	Private state	FALSE	N/A	NONE	PURE
wicket-datable-adddtooolbar-291fe8de	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
camel-defaultproducer-template-asyncrequestbody-aced1e32	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
activiti-value-ddataobject-gettype-flaf6c228	FALSE	N/A	Private state	N/A	N/A	NONE	INTERMEDIATE
commons-math-statutils-geometricmean-600fe704	FALSE	N/A	N/A	FALSE	N/A	NONE	COMPLEX
xmlgraphics-fop-pdfimagesobject-output-09052584	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
flink-normalizedkeyutil-putchamormalizedkey-c83a41b8	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
wicket-anonymous-ec2adc75-2ec9-4a68-8218-10b4642922bb-next-66603b0b	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
poi-packagingurihelper-resolveparturi-92f40a2b	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
commons-math-abstractfieldintegrator-geteventhandlers-f5de1212	FALSE	N/A	Input Validation	N/A	N/A	POOR	PURE
jfreechart-xydrawableannotation-xydrawableannotation-788685e8	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
closure-compiler-codeprinter-testtypeannotationsmemberstub-c8785652	TRUE	TRUE	Input Validation	TRUE	TRUE	WORD	COMPLEX
jackrabbit-nodeimpl-cancelmerge-15e039c2	FALSE	N/A	Private state	N/A	N/A	POOR	COMPLEX
commons-lang-failblatestream-foreach-fc5485d2	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
asm-type-tostring-9564e369	FALSE	TRUE	N/A	N/A	N/A	WORD	COMPLEX
wicket-buttonresource-respond-c0328c5	FALSE	N/A	Private state	N/A	N/A	NONE	COMPLEX
accumulo-standaloneclustercontrol-suspend-c4034a18	FALSE	N/A	Hybrid Untestable	N/A	N/A	NONE	COMPLEX
flink-sourceoperator-sourceoperator-2f2a6c31	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
flink-rmatgraph-setconstants-d4a08cef8	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
commons-math-diagonalmatrix-add-c991ba85	FALSE	N/A	Hybrid Testable	N/A	N/A	WORD	PURE
lucene-solr-fieldinfo-setomitsnorms-2d77007a	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-objecthashset-objecthashset-a885555a	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
camel-objectnode-addchildren-67564d26	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
accumulo-remoteshell-renoteshell-4d237af44	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
jmonkeyengine-hdrenderer-reshape-99d512ea	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
wildfly-elytron-identitymappingbuilder-build-02966c485	TRUE	N/A	Private state	FALSE	N/A	NONE	PURE
commons-configuration-configurationpropertiesfactorybean-afterpropertieset-24590afc	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
flink-abstractsource-manager-driver-initialize-13ad4cb5	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
camel-staxjaxbiteratorexpression-staxjxaxbiteratorexpression-c175e893	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
flink-pythonkye-dcoprocessoperator-onprocessingtime-475b94e0	FALSE	TRUE	N/A	N/A	TRUE	NONE	PURE
camel-defaultpgppublickeyaccessor-defaultpgppublickeyaccessor-26036f09	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	COMPLEX
h2database-simpleresultset-getcolumnname-pe-50ae0e0c	FALSE	N/A	Hybrid Testable	N/A	TRUE	NONE	PURE
jackrabbit-orderhandler-value-c603e8f1	TRUE	N/A	Private state	FALSE	N/A	NONE	INTERMEDIATE

id	wit Correctness	miner Correctness	source Checkability	wit Checkability	miner Checkability	documentation	check Effort
joda-time-years-between-54126dff	TRUE	N/A	Input Validation	TRUE	N/A	NONE	COMPLEX
flink-restclientconfiguration-fromconfiguration-17019c10	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
camel-objectnode-addechildren-2b6b2846	TRUE	N/A	Input Validation	FALSE	N/A	NONE	PURE
lucene-solr-anonymous-5764f235-4b0a-465d-98fe-28598109e2c7-add-754b09ee	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
lucene-solr-conjunctionspans-positions-cost-024cf08e	TRUE	N/A	Unconditionally Throw	TRUE	N/A	NONE	COMPLEX
asterisk-java-listdialplaneevent-setpriority-214db761	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
swingx-treetablehack-hithandledetectionfrom-ditell-20330a25	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
commons-configuration-properties-builderparametersimpl-inheritfrom-6b2fa101	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
jfreechart-meterinterval-meterinterval-e6690f23	TRUE	N/A	Input Validation	TRUE	N/A	VAR	INTERMEDIATE
flink-embeddedexecutorfactory-embeddedexecutorfactory-71a9e5f8	FALSE	N/A	Input Validation	N/A	N/A	NONE	PURE
flink-declarativeslotmanager-start-499ecbe8	FALSE	N/A	Input Validation	N/A	N/A	NONE	PURE
jackrabbit-workspaceoperationtest-testcopy-99169b55	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
lucene-solr-jaspellernarysearchtrie-jaspellernarysearchtrie-82da027d	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
poi-ole10native-ole10native-d212018a	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
flink-filesystemblobstore-filesystemblobstore-cca566f1	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
flink-kinesisdynamicsink-copy-a3f29c3f	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
camel-skiplastbyteinputstream-read-d7d21d7	TRUE	N/A	Input Validation	TRUE	N/A	NONE	INTERMEDIATE
flink-closingfsdataoutputstream-wrapsafe-61d28987	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-bufferwritingsulpartition-setup-4b5ca08a	TRUE	N/A	Input Validation	FALSE	N/A	NONE	INTERMEDIATE
jackrabbit-oak-anonymous-06d24382-5fb9-4402-9f63-ca3b6cd67fc1-readsegment-40e90863	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
lucene-solr-xysshape-newboxquery-66aa8600	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-bufferbuilder-append-c128e0c2	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
commons-math-baseabstractunivariate-solver-solve-3a277d17	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
swingx-componenttreetablemodel-convertnumtotext-l439175e	FALSE	TRUE	N/A	N/A	N/A	POOR	COMPLEX
logging-log4j2-log4jmarker-remove-cfa42a2e	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-resourceprofile-fromresources-10662a10	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
accumulo-kerberosrenewali-stopkdc-d6ab8e47	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
lucene-solr-typostingsenum-getpayload-574001d1	FALSE	N/A	Hybrid Unstable	N/A	N/A	NONE	INTERMEDIATE
jfreechart-defaultkeyedvalues-insetValue-3919be65	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
commons-text-strsubstitutor-replace-d50ecd1f	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
poi-spmutil-geotppri-68ae5ed	TRUE	N/A	Input Validation	TRUE	N/A	NONE	PURE
flink-builder-column-1c2b3d35	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
lucene-solr-restore-getparams-cfc2b790	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
jackrabbit-derbyfilesystem-close-a13db187	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
logging-log4j2-eventresolversinterceptors-populateinterceptors-f455bece	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
byte-buddy-withoutmapping-bindlambda-7ef0c638	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX

id	wt Correctness	miner Correctness	source Checkability	wt Checkability	miner Checkability	documentation	check Effort
xmlgraphics-fop-pdfcidssysteminfo-topdfstring-f3ca215d	FALSE	N/A		N/A	N/A	NONE	COMPLEX
lucene-solr-blockpack-kedreaderiterator-blockpackedreaderiterator-d5e3bc08	TRUE	N/A	Input Validation	TRUE	N/A	WORD	PURE
flink-streamtableenvironmentimpl-fromchangelogstream-eb4d22de	TRUE	N/A	Hybrid Testable	FALSE	N/A	NONE	PURE
poi-ptg-readtokens-8920e99d	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-intvaluearray-add-7dc2ac39	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-binarymapdata-pointtto-1f68be4c	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
lucene-solr-geosouthrectangle-geosouthrectangle-4119d437	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-rowtypeinfo-gettypepeat-40794da5	FALSE	N/A	Hybrid Untestable	N/A	N/A	NONE	COMPLEX
itext7-pdfinspect-createembeddedfilespec-d50e9f01	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
jmonkeyengine-lwjglgltext-glbufferuserdata-d0ba7cd9	FALSE	N/A	Hybrid Testable	N/A	N/A	NONE	PURE
camel-joorlanguage-createexpression-54a09838	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-binaryarraydata-getboolean-37b7d725	FALSE	N/A	Hybrid Untestable	N/A	N/A	NONE	PURE
flink-statefulsequencesource-initializestate-f02cf46c	TRUE	N/A	Hybrid Testable	FALSE	N/A	NONE	PURE
dubbo-ioutils-geurl-a069ba6e	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
closure-compiler-anonymous-fac23e1e-b74f-4b00-8c1e-fd8d6b6cc8f8-enterscope-9d310c1b	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
jmonkeyengine-lwjgl-gluniforrmatrix4-7a2ef860	TRUE	N/A	Hybrid Testable	FALSE	N/A	NONE	PURE
lucene-solr-pathhierarchyoptimizer-pathhierarchyoptimizer-8a995940	TRUE	N/A	Input Validation	TRUE	N/A	NONE	COMPLEX
jackrabbit-impersonationimpl-getimpersonators-40d3a066	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
flink-kuberulesleaderretrievaldriverfactory-createleaderretrievaldriver-17657528	FALSE	TRUE	N/A	N/A	N/A	NONE	COMPLEX
commons-configuration-filehandler-setfilelocator-d92d1a53	TRUE	TRUE	Input Validation	TRUE	TRUE	VAR	PURE
jmonkeyengine-mesh-gettrianglecount-1c172a23	TRUE	TRUE	Hybrid Untestable	FALSE	FALSE	NONE	COMPLEX
camel-timerendpoint-createproducer-fffc00d	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
activiti-timerjobqueryimpl-messages-c8b11f98	TRUE	TRUE	Private state	FALSE	TRUE	NONE	PURE
flink-equijoin-equijoin-9192449c	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
commons-lang-objectutils-const-short-7ef1a952	TRUE	TRUE	Input Validation	TRUE	TRUE	VAR	PURE
spoon-substitutor-visictmethod-c4fde28d	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
commons-bcel-pass3bverifier-invalidreturnypeerror-a6e8136c	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
flink-fscheckpointstreamfactory-fscheckpointstreamfactory-a29af122	TRUE	TRUE	Input Validation	TRUE	TRUE	POOR	PURE
dubbo-javabeandescrptor-getprimitiveproperty-f8807c3d	TRUE	TRUE	Public Visible State	FALSE	TRUE	NONE	PURE
spoon-patternbuilder-visictannotation-85b05e5f	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
lucene-solr-stopfilterfactory-stopfilterfactory-fce70556	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	POOR	PURE
jackrabbit-versioncontrolledresourceimpl-update-c37b25e2	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
spoon-patternbuilder-visictannotationtype-06458832	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
camel-readerinputstream-read-0427b449	TRUE	TRUE	Input Validation	TRUE	TRUE	POOR	PURE
lucene-solr-querycommand-sefilterlist-eebe7f0e	TRUE	TRUE	Input Validation	FALSE	TRUE	WORD	PURE
jackrabbit-abstractsession-setnamespacesprefix-10e5c314	TRUE	TRUE	Input Validation	TRUE	TRUE	POOR	PURE

id	wit Correctness	miner Correctness	source Checkability	wit Checkability	miner Checkability	documentation	check Effort
lucene-solr-doubledocvaluesfield-setlongvalue-72495416	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
poi-styl-estable-setmaximumnumberofdataformats-c060f647	TRUE	TRUE	Hybrid Testable	TRUE	TRUE	VAR	COMPLEX
jackson-databind-beamserializerbuilder-settypeid-c6d6d1cf	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
flink-genericdatasourcebase-genericdatasourcebase-a6971365	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
jackrabbit-staleproperty-issame-f386f2c2	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE
lucene-solr-uniformdistributionevaluator-dowork-fda7bf5	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
flink-genericinputformat-createinputsplits-c2be0403	TRUE	TRUE	Input Validation	TRUE	TRUE	NONE	PURE
commons-lang-strtokenizer-add-0671cddb	TRUE	TRUE	Unconditionally Throw	TRUE	TRUE	NONE	PURE