Tool Warning Analysis

Goal

To determine if a representative sample of warnings issued by the tools are false or true positives.

Methodology

Sampling:

- Our goal is to analyze a sample of 328 of the 2,237 warnings we obtained from the tools. This is a representative sample of the warnings (at 95% confidence and 5% margin of error).
- We will select the warnings per dataset proportionally to the total number of warnings we got for all the datasets (see the statistics spreadsheet).
- Instead of sampling the warnings directly, we adopt a stratified sampling procedure by dataset and then by snippet:
    - We randomly select one dataset and one snippet from the dataset and review all the warnings issued for that snippet.
    - We repeat the previous step until we reach at least 100% of the warnings for each dataset.
- We adopt the stratified procedure to facilitate/speed up the manual review of warnings, without compromising the statistical soundness of our sampling.

Procedure for 1st author:

- among the datasets that are not at or above 100% warnings (cells I3-8 in the statistics spreadsheet), choose a dataset at random
- choose a snippet at random without replacement from that dataset
- create a heading in this document for the dataset + snippet number combo
- determine the expected number of warnings on that snippet using the following command: `rg "$X,$Y," data/raw_correlation_data.csv`, where $X is the dataset code and $Y is the snippet number. Record that number in this document.
- locate the snippet in the source code. Record the file name and line numbers in this document.
- for each expected warning, locate it in the corresponding output file in the repo. Do this by searching the correct tool output .txt file for the dataset + tool for the file name + line numbers; all tools output the file name + line number for each warning.
- for each located warning, copy it into this document and determine whether it is a false positive using your own judgment and record why in this document. Warnings that could

be resolved by writing a specification/annotation count as false positives for this purpose. If the warning is a true positive, add “**TRUE POSITIVE**” in bold to the description that you record.
- record summary numbers (columns A-E) in a new row in the statistics spreadsheet. If there are one or more TPs in the snippet, you might also need to make up + add a reason code around cell F16 in the statistics spreadsheet (if the reason is different than other TPs discovered before).
- continue this procedure until we are at least 100% in the spreadsheet for each dataset (i.e., the target representative sample of warnings for each dataset)

We prototyped/developed this procedure on snippet 1,1 (selected by convenience).

Procedure for 2nd author:
- review each warning, code snippet, and judgement/justification given by the 1st author in this document
- determine if 1st author’s assessment is correct (that indeed the warning is a false or true positive and the author’s explanation is correct)
- if the assessment is correct, write “**VERIFIED**”, otherwise if it is incorrect or unclear, write “**INCORRECT_UNCLEAR**” and give explanation for this assessment.
- in case of disagreements, meet with 1st author, discuss the cases and arrive at a conclusion. Update the document (by appending the final conclusion of the discussion) and spreadsheet if needed.

Warning analysis

dataset 1, snippet 1

OpenJML:

```
cog_complexity_validation_datasets/One/Tasks.java:48: verify: The prover cannot establish an assertion (ArithmeticOperationRange) in method main1: int multiply overflow
        result = result * x;
                        ^
```

Double-checked that this matches the correlation results, which indicate that OpenJML issued 1 warning and the other tools issued zero.

This is definitely a false positive: x is 4 and reduces by 1 each loop iteration, so `result` cannot possibly overflow in an int (its size is bounded by 4^4 = 256).

VERIFIED.

dataset 3, snippet 81

1 warning, from the typestate checker:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:630: warnin\
g: Cannot assign because [this.importedPackages] is not accessible here
            importedPackages = new Vector();
                            ^
```

This warning indicates that importedPackages shouldn't be assigned in this location. I think the reason for that is that the typestate checker is attempting to enforce linearity: that is, that each object is only used (+ assigned) in a unique location. Because this field is visible from other methods than this one, the warning is issued at re-assignment.

VERIFIED:
TypeStateChecker issues this type of warning when enforcing linearity:
https://github.com/jdmota/java-typestate-checker/blob/9a90e16bb431e6fcfbf79aa8298d08110562a0f6/tests/basic/LinearityTests.java#L470

dataset f, snippet 7

no warnings

dataset 2, snippet 8

3 expected warnings, all from the typestate checker. This snippet is lines 244-252 in Tasks.java. Note that there is overlap between datasets 1 and 2, and this is in the overlap, so I've counted it for both 1 & 2.

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/One/Tasks.java:249:
warning: Cannot assign: cannot cast from Unknown to Shared{java.lang.String} | Null
            result += word.charAt(j);
```

Clearly a false positive, since chars cannot be null.

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/One/Tasks.java:244:
warning: [result[85] StringConcat ToString charAt(word, j)] did not complete its protocol (found:
Unknown)
    public static void main14(String[] args) {
                       ^
```

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/One/Tasks.java:244:
warning: [result[85] = result[85] StringConcat ToString charAt(word, j)] did not complete its
protocol (found: Unknown)
```

```
    public static void main14(String[] args) {
                       ^
```

These two errors are a bit nonsensical, since I'm not sure to what they're referring. The two are slightly different (note the "result[85]" vs "result[85] = result[85]". In general, the TS checker issues this error when a protocol isn't finished (e.g, when you forget to close a stream). I'm not sure what protocol it's trying to enforce here, but it's definitely a false positive.

VERIFIED

dataset 6, snippet 12

94 (!) warnings:

- 91 (!) from the typestate checker
- 1 from infer
- 2 from openjml
- 0 from the CF

This snippet is lines 93-144 of CarReport.java.

OpenJML:

```
./CarReport.java:114: verify: The prover cannot establish an assertion
(PossiblyNullDeReference) in method afterTextChanged
                mTxtTrustCertificate.setVisibility(View.GONE);
                                ^
./CarReport.java:115: verify: The prover cannot establish an assertion
(PossiblyNullDeReference) in method afterTextChanged
                mChkTrustCertificate.setChecked(false);
```

These are definitely false positives, because these fields are set just below those lines (and this is a callback).

Infer:

```
dataset6/src/main/java/CarReport.java:99: error: Null Dereference
  object returned by `this$0.getWindow()` could be null and is dereferenced at line 99.
   97.            super.onCreate(savedInstanceState);
   98.            setContentView(R.layout.activity_setup_webdav_sync);
   99. >          getWindow().setLayout(ViewGroup.LayoutParams.MATCH_PARENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
  100.
  101.            mEdtUrl = (EditText) findViewById(R.id.edt_url);
```

TRUE POSITIVE: This looks like an artifact of our stubbing-out process: getWindow() is defined to always return null, and I suspect Infer has deduced that and is reporting for that reason.

Typestate Checker:

- many, many “cannot access” warnings (about ⅔ of the total warnings) like these:

```
dataset6/src/main/java/CarReport.java:139: warning: Cannot access
[CarReport.Activity.RESULT_CANCELED]
            setResult(Activity.RESULT_CANCELED);
                               ^
dataset6/src/main/java/CarReport.java:114: warning: Cannot access [this.mTxtTrustCertificate]
            mTxtTrustCertificate.setVisibility(View.GONE);
            ^
dataset6/src/main/java/CarReport.java:113: warning: Cannot access [CarReport.View.GONE]
            mTxtTrustCertificateDescription.setVisibility(View.GONE);
                                                               ^
dataset6/src/main/java/CarReport.java:113: warning: Cannot access
[this.mTxtTrustCertificateDescription]
            mTxtTrustCertificateDescription.setVisibility(View.GONE);
            ^
dataset6/src/main/java/CarReport.java:116: warning: Cannot access [CarReport.View.GONE]
            mChkTrustCertificate.setVisibility(View.GONE);
                                                    ^
dataset6/src/main/java/CarReport.java:114: warning: Cannot access [CarReport.View.GONE]
            mTxtTrustCertificate.setVisibility(View.GONE);
                                                    ^
dataset6/src/main/java/CarReport.java:115: warning: Cannot access [this.mChkTrustCertificate]
            mChkTrustCertificate.setChecked(false);
            ^
```

These are caused by the typestate checker’s insistence that fields cannot be public without a human writing an annotation, which is overly conservative. Therefore, they are false positives.

- some “unsafe cast” warnings (about ⅙ of the total), like these:

```
dataset6/src/main/java/CarReport.java:120: warning: Unsafe cast
        mEdtPassword = (EditText) findViewById(R.id.edt_password);
                       ^
dataset6/src/main/java/CarReport.java:121: warning: Unsafe cast
        mTxtTrustCertificateDescription = (TextView)
findViewById(R.id.txt_trust_certificate_description);
                                          ^
dataset6/src/main/java/CarReport.java:123: warning: Unsafe cast
        mChkTrustCertificate = (CheckBox) findViewById(R.id.chk_trust_certificate);
                               ^
dataset6/src/main/java/CarReport.java:129: warning: Unsafe cast
```

```
        mBtnOk = (Button) findViewById(R.id.btn_ok);
                ^
dataset6/src/main/java/CarReport.java:101: warning: Unsafe cast
        mEdtUrl = (EditText) findViewById(R.id.edt_url);
                 ^
dataset6/src/main/java/CarReport.java:122: warning: Unsafe cast
        mTxtTrustCertificate = (TextView) findViewById(R.id.txt_trust_certificate);
                              ^
dataset6/src/main/java/CarReport.java:119: warning: Unsafe cast
        mEdtUserName = (EditText) findViewById(R.id.edt_user_name);
                      ^
```

These casts are perfectly safe, so these are false positives, too.

- some “cannot assign” warnings, like these (about ⅙ of the total):

```
dataset6/src/main/java/CarReport.java:119: warning: Cannot assign because
[this.mEdtUserName] is not accessible here
        mEdtUserName = (EditText) findViewById(R.id.edt_user_name);
                    ^
dataset6/src/main/java/CarReport.java:120: warning: Cannot assign because
[this.mEdtPassword] is not accessible here
        mEdtPassword = (EditText) findViewById(R.id.edt_password);
```

These are false positives: the checker is enforcing an ownership model that is stronger than the standard Java one, but which this code does not (and should not) obey. It is safe to assign a field, and these fields are accessible under standard Java rules.

There are no other warnings.

VERIFIED.

dataset f, snippet 13

2 warnings from OpenJML:

```
fMRI_Study_Classes/RecursiveFibonacciVariant.java:14: verify: The prover cannot establish an
assertion (ArithmeticOperationRange) in method compute: underflow in int sum
      return compute(number - 2) + compute(number - 4);
                                 ^
fMRI_Study_Classes/RecursiveFibonacciVariant.java:14: verify: The prover cannot establish an
assertion (ArithmeticOperationRange) in method compute: overflow in int sum
      return compute(number - 2) + compute(number - 4);
```

The underflow is obviously a false positive, since compute() of any number less than 1 is 1 (with no recursive call).

The overflow warning is trickier. I think technically this code can overflow, if a sufficiently-large number is given as input. However, one could easily write a specification that limits the inputs to numbers that would not result in an overflow, and that spec would be verifiable. So, I think we can regard this as a case of a missing annotation.

VERIFIED.

dataset 9, snippet 2

10 warnings, 5 each from the typestate checker and openJML. The snippet is lines 458-471 of CodeSnippets.java.

OpenJML:

```
./CodeSnippets.java:466: verify: The prover cannot establish an assertion (InvariantLeaveCaller:
/home/authors/openjml/specs/java/io/PrintStream.jml:35:) in method
logAndEmailSeriousProblemS112: (Caller:
CodeSnippets.logAndEmailSeriousProblemS112(java.lang.Throwable,javax.servlet.http.HttpSer
vletRequest), Callee: java.io.PrintStream.println(java.lang.String))
   System.out.println("SERIOUS PROBLEM OCCURRED.");// changed to allow compilation
                    ^
./CodeSnippets.java:466: verify: The prover cannot establish an assertion (InvariantLeaveCaller:
/home/authors/openjml/specs/java/io/PrintStream.jml:42:) in method
logAndEmailSeriousProblemS112: (Caller:
CodeSnippets.logAndEmailSeriousProblemS112(java.lang.Throwable,javax.servlet.http.HttpSer
vletRequest), Callee: java.io.PrintStream.println(java.lang.String))
   System.out.println("SERIOUS PROBLEM OCCURRED.");// changed to allow compilation
                    ^
./CodeSnippets.java:467: verify: The prover cannot establish an assertion (InvariantLeaveCaller:
/home/authors/openjml/specs/java/io/PrintStream.jml:42:) in method
logAndEmailSeriousProblemS112: (Caller:
CodeSnippets.logAndEmailSeriousProblemS112(java.lang.Throwable,javax.servlet.http.HttpSer
vletRequest), Callee: java.io.PrintStream.println(java.lang.String))
   System.out.println(troubleTicket.toString());// changed to allow compilation
                    ^
./CodeSnippets.java:467: verify: The prover cannot establish an assertion (InvariantLeaveCaller:
/home/authors/openjml/specs/java/io/PrintStream.jml:35:) in method
logAndEmailSeriousProblemS112: (Caller:
CodeSnippets.logAndEmailSeriousProblemS112(java.lang.Throwable,javax.servlet.http.HttpSer
vletRequest), Callee: java.io.PrintStream.println(java.lang.String))
   System.out.println(troubleTicket.toString());// changed to allow compilation
```

```
                  ^
./CodeSnippets.java:470: verify: The prover cannot establish an assertion (InvariantEntrance:
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method
logAndEmailSeriousProblemS112: (Caller:
CodeSnippets.logAndEmailSeriousProblemS112(java.lang.Throwable,javax.servlet.http.HttpSer
vletRequest), Callee:
javax.servlet.ServletContext.setAttribute(java.lang.String,java.lang.Object))
   setAttribute(MOST_RECENT_TROUBLE_TICKET, troubleTicket);
```

The first four errors are all saying that System.out's print stream might not be closed. Obviously false positives!

The last error is referring to the internal specification of java.lang.CharSequence, suggesting that the internal character array hasn't been allocated. That is obviously false, since the constant string here is actually a string constant. So this is also an FP.

typestate:

```
dataset9/src/main/java/CodeSnippets.java:464: warning: Cannot access
[CodeSnippets.fLogger]
   fLogger.severe(troubleTicket.toString());
   ^

dataset9/src/main/java/CodeSnippets.java:463: warning: Cannot access
[CodeSnippets.fLogger]
   fLogger.severe("TOP LEVEL CATCHING Throwable.");
   ^

dataset9/src/main/java/CodeSnippets.java:469: warning: Cannot call setAttribute on null
   aRequest.getSession().getServletContext().
                                    ^

dataset9/src/main/java/CodeSnippets.java:470: warning: Cannot access
[CodeSnippets.MOST_RECENT_TROUBLE_TICKET]
   setAttribute(MOST_RECENT_TROUBLE_TICKET, troubleTicket);
             ^

dataset9/src/main/java/CodeSnippets.java:469: warning: Cannot call getServletContext on null
   aRequest.getSession().getServletContext().
                   ^
```

The 3 "cannot access" warnings are obviously FPs. The "cannot call" warnings are a bit trickier: effectively, they are claiming that aRequest and aRequest.getSession() might both be null. getSession() cannot return null according to its docs. aRequest is a method parameter, so if it were nullable a human would need to indicate that, but the comments don't do that. We can definitely regard both as false positives.

VERIFIED.

dataset 3, snippet 25

5 warnings:

- 4 from the CF
- 1 from the typestate checker

Lines 839-847 in Tasks1.java.

CF:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:840: warning: [not.interned] attempting to use a non-@Interned comparison operand
        if (p.getTile().getSettlement() != null && p.getTile().getSettlement().getOwner() == player
                                                             ^
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:840: warning: [not.interned] attempting to use a non-@Interned comparison operand
        if (p.getTile().getSettlement() != null && p.getTile().getSettlement().getOwner() == player
                                                                ^
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:841: warning: [not.interned] attempting to use a non-@Interned comparison operand
                && p.getTile().getSettlement() != inSettlement) {
                                  ^
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:841: warning: [not.interned] attempting to use a non-@Interned comparison operand
                && p.getTile().getSettlement() != inSettlement) {
```

These warnings are all because objects are being compared using the == operator. Presumably doing so is safe in this context (making these FPs - the warnings would be issued even if the code is correct), but with the context missing here there is no way to be 100% sure. These are borderline, but I'd judge them still to be false positives: this looks like videogame code, which is typically heavily optimized, which is one of the few places in Java where == over .equals() makes sense.

Another interpretation of this is that this snippet is wrong, and these really should be .equals() calls. If that were the case, these would be true positive warnings. Regardless, the use of ==

here rather than .equals() is definitely making this snippet more complicated for any competent Java programmer, so I don't feel these threaten our conclusions.

typestate:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:844:
warning: Cannot call add on null
        destinations.add(new ChoiceItem(s.toString() + " (" + turns + ")", s));
                ^
```

destinations is a field, and there is no indication that it could be nullable. Looks like an FP to me (or at least a missing annotation).

VERIFIED.

dataset 3, snippet 65

4 warnings:

- 1 from OpenJML
- 3 from the typestate checker

lines 712-718 in Tasks_2.java

typestate checker:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:713:
warning: Cannot access [this.messages.length]
     String[] texts = new String[messages.length];
                              ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:714:
warning: Cannot access field [length] of null
     ImageIcon[] images = new ImageIcon[messages.length];
                                  ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:716:
warning: Cannot call getMessageID on null
        String ID = messages[i].getMessageID();
                       ^
```

All of these warnings indicate that messages might be or contain null. The original snippet doesn't give any specification for messages, so these are presumably false positives or indicate an annotation is required.

openjml:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:716: verify: The prover cannot
establish an assertion (PossiblyNegativeIndex) in method s65
            String ID = messages[i].getMessageID();
                                 ^
```

Obviously a false positive, since i ranges from 0 to messages.length. This is the simplest for loop for an index checking tool to verify I've ever seen…

VERIFIED.

dataset 3, snippet 1

6 warnings, all from the TS checker

lines 529-542 in Tasks_1.java

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:537:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_1.RETURN]
                case RETURN:
                     ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:530:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_1.callstack]
        Object ret = body.eval(callstack, interpreter);
                               ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:530:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_1.body]
        Object ret = body.eval(callstack, interpreter);
                     ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:530:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_1.interpreter]
        Object ret = body.eval(callstack, interpreter);
                                          ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:535:
warning: Cannot access [ret.kind]
            switch(((ReturnControl)ret).kind )
                                        ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:541:
warning: Incompatible return value: cannot cast from Shared{java.lang.Object} | Null to
Shared{java.lang.Object}
        return ret; // had to be added to allow compilation
        ^
```

These are all false positives. The 5 "cannot access" errors indicate the need for ownership annotations, I think. The last error indicates that ret could be null, but there is no reason to believe that it might be unless body.eval() could return null, which the original snippet doesn't give an indication of. A charitable interpretation of this last warning is that an @Nullable annotation or similar is needed on the method's return, but that would also be a false positive by our definition.

VERIFIED.

dataset 1, snippet 8

2 warnings, both from openjml

lines 159-168 in Tasks.java. Also DS2 snippet 4.

```
cog_complexity_validation_datasets/One/Tasks.java:164: verify: The prover cannot establish an
assertion (ArithmeticOperationRange) in method main8: overflow in int sum
            result = result + number % 10;
                           ^
cog_complexity_validation_datasets/One/Tasks.java:164: verify: The prover cannot establish an
assertion (ArithmeticOperationRange) in method main8: underflow in int sum
            result = result + number % 10;
                           ^
```

Both are false positives, since the loop is bounded and over/underflow never occurs.

VERIFIED.

dataset 3, snippet 90

2 warnings, both from the TS checker

lines 754-766 in Tasks_3

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:755:
warning: Cannot access
[cog_complexity_validation_datasets.Three.Tasks_3.ViewMode.VIEW_TERRAIN_MODE]
        if (currentMode == ViewMode.VIEW_TERRAIN_MODE) {
                           ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:757:
warning: Cannot call getSelectedTile on null
            Position selectedTilePos = gui.getSelectedTile();
                                          ^
```

First one is clearly a false positive (the TS checker's insistence that fields not be accessed).

Second one is also an FP, as it is asserting that gui is nullable, but there is no evidence of that.

VERIFIED.

dataset 1, snippet 17

no warnings. Also dataset 2, snippet 9

dataset 1, snippet 11

also dataset 2, snippet 6

one warning, from openjml

lines 203-211 in Tasks.java

```
cog_complexity_validation_datasets/One/Tasks.java:208: verify: The prover cannot establish an
assertion (ArithmeticOperationRange) in method main11: int multiply overflow
            result = result * num1;
                            ^
```

False positive, since the values being multiplied are small constants.

VERIFIED.

dataset 6, snippet 6

29 warnings:

- 25 from the typestate checker
- 3 from the CF
- 1 from openjml

lines 75-106 in Pom.java

CF:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/dataset6/src/main/jav
a/Pom.java:75: warning: [initialization.fields.uninitialized] the constructor does not initialize
fields: localizibleDescription
        public HealthReport(int score, String iconUrl, Localizable description) {
               ^
```

False positive: a setter is called unconditionally on line 104.

```
/home/authors/Code-Complexity-Research/complexity-verification-project/dataset6/src/main/jav
a/Pom.java:103: warning: [assignment] incompatible types in assignment.
        this.description = null;
                           ^
  found   : null (NullType)
  required: @Initialized @NonNull String
```

False positive: missing field annotation.

```
/home/authors/Code-Complexity-Research/complexity-verification-project/dataset6/src/main/jav
a/Pom.java:104: warning: [method.invocation] call to setLocalizibleDescription(Pom.Localizable)
not allowed on the given receiver.
        setLocalizibleDescription(description);
                                 ^
  found   : @Initialized @NonNull Pom.@UnderInitialization @NonNull HealthReport
  required: @Initialized @NonNull Pom.@Initialized @NonNull HealthReport
```

False positive: missing @UnderInitialization annotation on setLocalizableDescription()'s receiver.

openjml:

```
./Pom.java:103: verify: The prover cannot establish an assertion (PossiblyNullAssignment) in
method HealthReport
        this.description = null;
                         ^
```

False positive (field is clearly intended to be nullable).

typestate checker:

```
dataset6/src/main/java/Pom.java:78: warning: Cannot access [Pom.HEALTH_0_TO_20]
            this.iconClassName = HEALTH_0_TO_20;
                                 ^
dataset6/src/main/java/Pom.java:78: warning: Cannot access [Pom.HEALTH_0_TO_20]
            this.iconClassName = HEALTH_0_TO_20;
                                 ^
dataset6/src/main/java/Pom.java:82: warning: Cannot access [Pom.HEALTH_41_TO_60]
            this.iconClassName = HEALTH_41_TO_60;
                                 ^
dataset6/src/main/java/Pom.java:86: warning: Cannot access [Pom.HEALTH_OVER_80]
            this.iconClassName = HEALTH_OVER_80;
                                 ^
```

… many more of these. All of these are false positives, because these fields are clearly intended to be accessed from here.

```
dataset6/src/main/java/Pom.java:103: warning: Cannot assign: cannot cast from Null to
Shared{java.lang.String}
        this.description = null;
                           ^
```
False positive: field is clearly intended to be nullable. Missing annotation?

VERIFIED.

dataset f, snippet 16

six warnings:

- 1 from the CF
- 5 from the typestate checker

lines 10-25 of YesNo.java

CF:
```
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/fMRI_Study_Classes/YesNo.java:25: warning: [return] incompatible types in return.
    return null;
           ^
  type of expression: null (NullType)
  method return type: @Initialized @NonNull Boolean
```

Clear FP, indicating a missing @Nullable annotation.

Typestate checker:
```
simple-datasets/src/main/java/fMRI_Study_Classes/YesNo.java:20: warning: Incompatible
return value: cannot cast from Primitive{boolean} to Shared{java.lang.Boolean}
        return true;
        ^
simple-datasets/src/main/java/fMRI_Study_Classes/YesNo.java:14: warning: Incompatible
return value: cannot cast from Primitive{boolean} to Shared{java.lang.Boolean}
        return false;
        ^
simple-datasets/src/main/java/fMRI_Study_Classes/YesNo.java:22: warning: Incompatible
return value: cannot cast from Primitive{boolean} to Shared{java.lang.Boolean}
        return true;
        ^
simple-datasets/src/main/java/fMRI_Study_Classes/YesNo.java:16: warning: Incompatible
return value: cannot cast from Primitive{boolean} to Shared{java.lang.Boolean}
```

```
            return false;
            ^
simple-datasets/src/main/java/fMRI_Study_Classes/YesNo.java:25: warning: Incompatible
return value: cannot cast from Null to Shared{java.lang.Boolean}
        return null;
        ^
```

All FPs as well, indicating missing annotations. The first four warnings are saying that Boolean and boolean aren't assignable (they are); the last is similar to the CF warning.

VERIFIED.

dataset 3, snippet 66

10 warnings:

- 1 from infer
- 5 from the typestate checker
- 4 from openjml

lines 722-729 of Tasks_2.java

Infer:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:727:
error: Null Dereference
  object returned by `Tasks_2.devplugin.Plugin.getPluginManager()` could be null and is
dereferenced at line 727.
  725.           Program p = (Program) mProgramTableModel.getValueAt(row, 1);
  726.
  727. >         JPopupMenu menu =
devplugin.Plugin.getPluginManager().createPluginContextMenu(p,
CapturePlugin.getInstance());
  728.
  729.       } // Added to allow compilation
```

TRUE POSITIVE: This is another instance of a stubbed-out null class causing an infer warning.

OpenJML:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:725: verify: The prover cannot
establish an assertion (PossiblyNullDeReference) in method s66
        Program p = (Program) mProgramTableModel.getValueAt(row, 1);
                                          ^
cog_complexity_validation_datasets/Three/Tasks_2.java:725: verify: The prover cannot
establish an assertion (NullFormal:
```

```
cog_complexity_validation_datasets/Three/Tasks_2.java:1712:) in method s66: row in
getValueAt(java.lang.Boolean[],int)
        Program p = (Program) mProgramTableModel.getValueAt(row, 1);
                                                            ^
cog_complexity_validation_datasets/Three/Tasks_2.java:725: verify: The prover cannot
establish an assertion (PossiblyBadCast) in method s66: a java.lang.Object cannot be proved to
be a cog_complexity_validation_datasets.Three.Tasks_2.Program
        Program p = (Program) mProgramTableModel.getValueAt(row, 1);
                    ^
cog_complexity_validation_datasets/Three/Tasks_2.java:727: verify: The prover cannot
establish an assertion (PossiblyNullDeReference) in method s66
        JPopupMenu menu = devplugin.Plugin.getPluginManager().createPluginContextMenu(p,
CapturePlugin.getInstance());
                                    ^
cog_complexity_validation_datasets/Three/Tasks_2.java:722: verify: Validity is unknown - time
or memory limit reached: : Aborted proof: timeout
    public void s66() {
                ^
```

The last is a timeout, so it is ignored. The first two are warnings about possible nullability in a field and parameter, respectively. These look like false positives/lack of annotations, to me: the code clearly doesn't expect them to be nullable. The third warning is a warning that the cast might fail at run time, which is obvious (the purpose of a cast is to suppress a false positive from Java's type system, so this warning is redundant with Java itself and clearly an FP). The fourth is an odd choice to complain about nullability, given that this is a static class and therefore guaranteed to be non-null, but a false positive regardless.

Typestate checker:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:725:
warning: Unsafe cast
        Program p = (Program) mProgramTableModel.getValueAt(row, 1);
                    ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:725:
warning: Cannot call getValueAt on null
        Program p = (Program) mProgramTableModel.getValueAt(row, 1);
                                                ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:727:
warning: Cannot access [this.devplugin.Plugin]
        JPopupMenu menu = devplugin.Plugin.getPluginManager().createPluginContextMenu(p,
CapturePlugin.getInstance());
                                    ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:723:
warning: Cannot call changeSelection on null
        mProgramTable.changeSelection(row, 0, false, false);
```

```
                  ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:723:
warning: Incompatible parameter: cannot cast from Shared{java.lang.Boolean[]} | Null to
Shared{java.lang.Boolean[]}
        mProgramTable.changeSelection(row, 0, false, false);
                                     ^
```

The first two are duplicates of OpenJML warnings. The third is a false positive indicating the need for an annotation. The last two are spurious nullness warnings: the code seems to pretty clearly assume these values are not nullable.

VERIFIED.

dataset 3, snippet 51

lines 554-561 of Tasks_2

1 warning, from the typestate checker:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:560:
warning: Incompatible return value: cannot cast from Shared{java.lang.String} | Null to
Shared{java.lang.String}
        return className.substring(i+1);
        ^
```

Definitely an FP, as substring cannot return null.

VERIFIED.

dataset 6, snippet 38

9 warnings:

- 1 each from the CF, infer, openjml
- 6 from the typestate checker

lines 144-177 of SpringBatch.java

CF:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/dataset6/src/main/jav
a/SpringBatch.java:177: warning: [return] incompatible types in return.
        return null;
               ^
  type of expression: null (NullType)
  method return type: T extends @Initialized @Nullable Object
```

FP, needs annotation. The method is a mock for different types of objects.

Infer:

```
dataset6/src/main/java/SpringBatch.java:148: error: Null Dereference
  object `ds` last assigned on line 146 could be null and is dereferenced at line 148.
  146.          DataSource ds = mock(DataSource.class);
  147.
  148. >         when(ds.getConnection()).thenReturn(con); // con1
  149.          con.close();
  150.          when(ds.getConnection()).thenReturn(con); // con2
```

TRUE POSITIVE: this is also a consequence of stubbing out classes/methods using "return null" rather than "throw new Error()" or similar.

OpenJML:

```
./SpringBatch.java:167: verify: The prover cannot establish an assertion
(PossiblyNullAssignment) in method testOperationWithDirectCloseCall
        con1_1 = null;
               ^
```

This is pretty clearly intended to be nullable, so I'm not sure what OpenJML is getting at here. OpenJML did encounter some internal errors while processing this class, which might be responsible for the oddly-incomplete output: for example, I'd expect to get the same warning on the next line (which also re-assigns a variable with null as the RHS), but we don't.

Typestate:

```
dataset6/src/main/java/SpringBatch.java:145: warning: Cannot assign: cannot cast from
Shared{java.lang.Object} | Null to Shared{java.sql.Connection} | Null
        Connection con = mock(Connection.class);
                   ^
dataset6/src/main/java/SpringBatch.java:145: warning: Cannot access
[java.sql.Connection.class]
        Connection con = mock(Connection.class);
                              ^
dataset6/src/main/java/SpringBatch.java:146: warning: Cannot assign: cannot cast from
Shared{java.lang.Object} | Null to Shared{javax.sql.DataSource} | Null
        DataSource ds = mock(DataSource.class);
                   ^
dataset6/src/main/java/SpringBatch.java:148: warning: Incompatible parameter: cannot cast
from Shared{java.sql.Connection} | Null to Shared{java.sql.Connection}
        when(ds.getConnection()).thenReturn(con); // con1
                         ^
dataset6/src/main/java/SpringBatch.java:146: warning: Cannot access
[javax.sql.DataSource.class]
```

```
        DataSource ds = mock(DataSource.class);
                                ^
dataset6/src/main/java/SpringBatch.java:150: warning: Incompatible parameter: cannot cast
from Shared{java.sql.Connection} | Null to Shared{java.sql.Connection}
        when(ds.getConnection()).thenReturn(con); // con2
                          ^
```

These are all false positives. 4 of the warnings are about nullness, and indicate missing annotations. The other two "cannot access" warnings are for some reason attempting to restrict the use of class literals, because they are in different packages. This is a nonsensical idea, and the checker should have permitted it (they are interned, so there is only ever one immutable .class object for each class loaded by the JVM).

VERIFIED.

dataset 3, snippet 59

5 warnings:

- 3 from typestate
- 1 from infer
- 1 from openjml

lines 647-654 in Tasks_2

typestate:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:649:
warning: Cannot access
[cog_complexity_validation_datasets.Three.Tasks_2.Session.INFO_CONNECTION_READONL
Y]
        Object info = getAttribute(Session.INFO_CONNECTION_READONLY);
                                  ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:651:
warning: Unsafe cast
        isReadOnly = ((Boolean) info).booleanValue();
                      ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:651:
warning: Cannot assign because [this.isReadOnly] is not accessible here
        isReadOnly = ((Boolean) info).booleanValue();
                   ^
```

First and third indicate missing annotations and are therefore FPs. 2nd is redundant with a javac FP.

openjml:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:651: verify: The prover cannot
establish an assertion (PossiblyBadCast) in method isReadOnly: a java.lang.Object cannot be
proved to be a java.lang.Boolean
        isReadOnly = ((Boolean) info).booleanValue();
                      ^
```

FP, redundant with javac.

infer:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:651:
error: Null Dereference
  object `info` last assigned on line 649 could be null and is dereferenced at line 651.
  649.          Object info = getAttribute(Session.INFO_CONNECTION_READONLY);
  650.
  651. >         isReadOnly = ((Boolean) info).booleanValue();
  652.
  653.          return isReadOnly;
```

TRUE POSITIVE: getAttribute is stubbed to always return null.

VERIFIED.

dataset 3, snippet 48

4 warnings, all from OpenJML

lines 507-519 in Tasks_2

```
cog_complexity_validation_datasets/Three/Tasks_2.java:514: verify: The prover cannot
establish an assertion (InvariantExceptionExit:
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method ComparisonFailure:
cog_complexity_validation_datasets.Three.Tasks_2.ComparisonFailure.ComparisonFailure(java
.lang.String,java.lang.String,java.lang.String) (parameter actual)
        public ComparisonFailure (String message, String expected, String actual) {
                                                                          ^
cog_complexity_validation_datasets/Three/Tasks_2.java:514: verify: The prover cannot
establish an assertion (InvariantExceptionExit:
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method ComparisonFailure:
cog_complexity_validation_datasets.Three.Tasks_2.ComparisonFailure.ComparisonFailure(java
.lang.String,java.lang.String,java.lang.String) (parameter expected)
        public ComparisonFailure (String message, String expected, String actual) {
                                                         ^
cog_complexity_validation_datasets/Three/Tasks_2.java:514: verify: The prover cannot
establish an assertion (InvariantExit:
```

```
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method ComparisonFailure:
cog_complexity_validation_datasets.Three.Tasks_2.ComparisonFailure.ComparisonFailure(java
.lang.String,java.lang.String,java.lang.String) (parameter expected)
        public ComparisonFailure (String message, String expected, String actual) {
                                                      ^
cog_complexity_validation_datasets/Three/Tasks_2.java:514: verify: The prover cannot
establish an assertion (InvariantExit:
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method ComparisonFailure:
cog_complexity_validation_datasets.Three.Tasks_2.ComparisonFailure.ComparisonFailure(java
.lang.String,java.lang.String,java.lang.String) (parameter actual)
        public ComparisonFailure (String message, String expected, String actual) {
                                                                      ^
```

These are all clearly false positives, seemingly related to an issue in how OpenJML handles strings. In particular, the problem's root cause seems to be an invariant in CharSequence.jml in their distributed specs.

VERIFIED.

dataset 3, snippet 18

3 warnings, all from openjml

lines 758-764 in Tasks_1

```
cog_complexity_validation_datasets/Three/Tasks_1.java:758: verify: Validity is unknown - time
or memory limit reached: : Aborted proof: timeout
    public Result runMain(String... args) {
                  ^
cog_complexity_validation_datasets/Three/Tasks_1.java:759: verify: The prover cannot
establish an assertion (InvariantEntrance:
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method runMain: (Caller:
cog_complexity_validation_datasets.Three.Tasks_1.runMain(java.lang.String...), Callee:
java.lang.String.concat(@org.jmlspecs.annotation.Nullable
java.lang.String,@org.jmlspecs.annotation.Nullable java.lang.String))
        System.out.println("JUnit version " + Version.id());
                                            ^

cog_complexity_validation_datasets/Three/Tasks_1.java:759: verify: The prover cannot
establish an assertion (InvariantLeaveCaller:
/home/authors/openjml/specs/java/io/PrintStream.jml:35:) in method runMain: (Caller:
cog_complexity_validation_datasets.Three.Tasks_1.runMain(java.lang.String...), Callee:
java.io.PrintStream.println(java.lang.String))
        System.out.println("JUnit version " + Version.id());
```

```
                      ^
```

```
cog_complexity_validation_datasets/Three/Tasks_1.java:759: verify: The prover cannot
establish an assertion (InvariantLeaveCaller:
/home/authors/openjml/specs/java/io/PrintStream.jml:42:) in method runMain: (Caller:
cog_complexity_validation_datasets.Three.Tasks_1.runMain(java.lang.String...), Callee:
java.io.PrintStream.println(java.lang.String))
        System.out.println("JUnit version " + Version.id());
                      ^
```

One instance of the string problem, and two instances of the printstream problem. All are obvious FPs. Also, a timeout.

VERIFIED.

dataset 3, snippet 67

no warnings

dataset 3, snippet 53

10 warnings:

- 1 from infer
- 1 from the CF
- 5 from the typestate checker
- 3 from openjml

lines 576-587 in Tasks_2

infer:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:579:
error: Null Dereference
  object `Tasks_2.suiteMethod` last assigned on line 578 could be null and is dereferenced at
line 579.
  577.          try {
  578.              suiteMethod= klass.getMethod("suite");
  579. >            if (! Modifier.isStatic(suiteMethod.getModifiers())) {
  580.                  throw new Exception(klass.getName() + ".suite() must be static");
  581.              }
```

TRUE POSITIVE: stubbed null returns

CF:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_2.java:582: warning: [argument]
incompatible argument for parameter o of invoke.
        suite= (Test) suiteMethod.invoke(null); // static method
                                         ^
  found   : null (NullType)
  required: @Initialized @NonNull Object
```

FP: indicates missing annotation on library method

typestate:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:578:
warning: Cannot assign because [this.suiteMethod] is not accessible here
        suiteMethod= klass.getMethod("suite");
                   ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:578:
warning: Cannot call getMethod on null
        suiteMethod= klass.getMethod("suite");
                          ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:582:
warning: Unsafe cast
        suite= (Test) suiteMethod.invoke(null); // static method
              ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:582:
warning: Cannot assign because [this.suite] is not accessible here
        suite= (Test) suiteMethod.invoke(null); // static method
             ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:582:
warning: Incompatible parameter: cannot cast from Null to Shared{java.lang.Object}
        suite= (Test) suiteMethod.invoke(null); // static method
                                         ^
```

All false positives:

- the first is an access error, which indicates a missing annotation
- the second also indicates a missing annotation: klass is clearly intended to be nonnull. This checker is modular, so (unlike infer) it cannot tell that klass is actually null because of stubbing: it would warn regardless of the implementation.
- the unsafe cast warning is redundant with javac and therefore obviously intentional (and so an FP)
- the fourth warning again indicates the need for human annotation

- the fifth is the same, but in a library: invoke clearly is intended to be null safe

openjml:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:582: verify: The prover cannot
establish an assertion (NullFormal:
cog_complexity_validation_datasets/Three/Tasks_2.java:1575:) in method s53: o in
invoke(java.lang.Object)
          suite= (Test) suiteMethod.invoke(null); // static method
                                                      ^

cog_complexity_validation_datasets/Three/Tasks_2.java:582: verify: The prover cannot
establish an assertion (PossiblyNullDeReference) in method s53
          suite= (Test) suiteMethod.invoke(null); // static method
                                             ^
cog_complexity_validation_datasets/Three/Tasks_2.java:580: verify: The prover cannot
establish an assertion (PossiblyNullDeReference) in method s53
             throw new Exception(klass.getName() + ".suite() must be static");
                                                ^
```

The first of these is similar to the warning from the CF: openjml requires the called method to have a spec indicating it can accept null. That indicates a missing annotation. The other two occur because OpenJML is assuming that suiteMethod and klass could be null, because no spec on the methods that produced them said otherwise. Again, annotation problem ergo FP.

VERIFIED.

dataset 3, snippet 30

5 warnings:
- 1 from openjml
- 4 from the typestate checker

lines 902-908 in Tasks_1

openjml:

```
cog_complexity_validation_datasets/Three/Tasks_1.java:904: verify: The prover cannot
establish an assertion (PossiblyBadCast) in method setMapTransform: a java.lang.Object
cannot be proved to be a cog_complexity_validation_datasets.Three.MapControlsAction
        MapControlsAction mca = (MapControlsAction)
freeColClient.getActionManager().getFreeColAction(MapControlsAction.ID);
                                         ^
```

Redundant with the javac cast, so this is clearly intentional and therefore an FP.

typestate:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:903:
warning: Cannot assign because [this.currentMapTransform] is not accessible here
        currentMapTransform = mt;
                    ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:904:
warning: Cannot access [cog_complexity_validation_datasets.Three.MapControlsAction.ID]
        MapControlsAction mca = (MapControlsAction)
freeColClient.getActionManager().getFreeColAction(MapControlsAction.ID);
                                                                    ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:904:
warning: Unsafe cast
        MapControlsAction mca = (MapControlsAction)
freeColClient.getActionManager().getFreeColAction(MapControlsAction.ID);
                       ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:904:
warning: Cannot call getActionManager on null
        MapControlsAction mca = (MapControlsAction)
freeColClient.getActionManager().getFreeColAction(MapControlsAction.ID);
                                       ^
```

First and second indicate need for access annotations; fourth indicates need for nullability annotation. 3rd is redundant with javac. All FPs.

VERIFIED.

dataset 3, snippet 64

11 warnings:

- 5 from openjml
- 6 from the typestate checker

lines 702-708 in Tasks_2

openjml:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:705: verify: The prover cannot
establish an assertion (InvariantEntrance:
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method s64: (Caller:
```

```
cog_complexity_validation_datasets.Three.Tasks_2.s64(), Callee:
cog_complexity_validation_datasets.Three.Tasks_2.Ns.getCatalogName(java.lang.String))
        clsCat = ns.getCatalogName(clsName);
                                  ^

cog_complexity_validation_datasets/Three/Tasks_2.java:704: verify: The prover cannot
establish an assertion (PossiblyBadCast) in method s64: a java.lang.Object cannot be proved to
be a java.lang.String
        clsName = (String) classNames.next();
                  ^

cog_complexity_validation_datasets/Three/Tasks_2.java:706: verify: The prover cannot
establish an assertion (InvariantEntrance:
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method s64: (Caller:
cog_complexity_validation_datasets.Three.Tasks_2.s64(), Callee:
cog_complexity_validation_datasets.Three.Tasks_2.Ns.getSchemaName(java.lang.String))
        clsSchem = ns.getSchemaName(clsName);
                                   ^

cog_complexity_validation_datasets/Three/Tasks_2.java:706: verify: The prover cannot
establish an assertion (PossiblyNullDeReference) in method s64
        clsSchem = ns.getSchemaName(clsName);
                     ^

cog_complexity_validation_datasets/Three/Tasks_2.java:706: verify: The prover cannot
establish an assertion (NullFormal:
cog_complexity_validation_datasets/Three/Tasks_2.java:1401:) in method s64: clsName in
getSchemaName(java.lang.String)
        clsSchem = ns.getSchemaName(clsName);
                                   ^
```

The first and third warnings are the string invariant issue, and therefore obvious FPs. The second is a cast and therefore redundant with javac, and therefore also an FP. The fourth and fifth both indicate the need for annotations/specs about nullability.

typestate:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:704:
warning: Unsafe cast
        clsName = (String) classNames.next();
                  ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:706:
warning: Cannot assign because [this.clsSchem] is not accessible here
```

```
          clsSchem = ns.getSchemaName(clsName);
                   ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:704:
warning: Cannot assign because [this.clsName] is not accessible here
          clsName = (String) classNames.next();
                  ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:703:
warning: Cannot call hasNext on null
      while (classNames.hasNext()) {
                       ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:705:
warning: Cannot call getCatalogName on null
          clsCat = ns.getCatalogName(clsName);
                     ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:705:
warning: Cannot assign because [this.clsCat] is not accessible here
          clsCat = ns.getCatalogName(clsName);
                 ^
```

The first is about a cast; the others all clearly indicate the need for specs/annotations about accessibility/nullability.

VERIFIED

dataset 3, snippet 42

no warnings

dataset 3, snippet 7

lines 616-627 in Tasks_1

1 warning from the Checker Framework:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:622: warning: [dep-ann]
deprecated item is not annotated with @Deprecated
    public void Date(int daysSince1970) { // return type void added to allow compilation
                ^
```

javac also gives this warning, so it's definitely not a real Checker Framework warning. I've added an @Deprecated annotation to the code, so when we re-run the Checker Framework this will go away (and it did: it's not in our final version). The actual cause of this warning is the @deprecated tag in the Javadoc comment just before the method.

VERIFIED

dataset 3, snippet 60

3 warnings:

- 2 from typestate
- 1 from openjml

lines 658-662 in Tasks_2

openjml:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:660: verify: The prover cannot
establish an assertion (NullFormal:
cog_complexity_validation_datasets/Three/Tasks_2.java:1680:) in method s60:
warehouseDialog in remove(cog_complexity_validation_datasets.Three.Tasks_2.WareHouse)
        remove(warehouseDialog);
               ^
```

FP: missing nullability spec on formal parameter

typestate:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:659:
warning: Cannot call getResponseBoolean on null
        boolean response = warehouseDialog.getResponseBoolean();
                                          ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:661:
warning: Incompatible return value: cannot cast from Primitive{boolean} to
Shared{java.lang.Object}
        return response;
        ^
```

both FPs: first indicates a missing nullability spec, second is just flat wrong (that is a legal cast in Java)

VERIFIED

dataset 3, snippet 79

16 warnings:

- 9 from the CF
- 7 from typestate

lines 594-605 of Tasks_3

CF:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:596: warning:
[array.access.unsafe.low] Potentially unsafe array access: the index could be negative.
        int i = Column.compare(session.database.collation, a[cols[j]],
                                                             ^
  found   : @LowerBoundUnknown int
  required: an integer >= 0 (@NonNegative or @Positive)
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:596: warning:
[array.access.unsafe.high] Potentially unsafe array access: the index could be larger than the
array's bound
        int i = Column.compare(session.database.collation, a[cols[j]],
                                                             ^
  found   : @UpperBoundUnknown int
  required: @IndexFor("this.a") or @LTLengthOf("this.a") -- an integer less than this.a's length
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:596: warning:
[array.access.unsafe.high.range] Potentially unsafe array access: the index could be larger than
the array's bound
        int i = Column.compare(session.database.collation, a[cols[j]],
                                                               ^
  index type found: @IntRange(from=-2147483648) int
  array type found: @UnknownVal int @UnknownVal []
  required        : index of type @IndexFor("this.cols") or @LTLengthOf("this.cols"), or array of
type @MinLen(-9223372036854775808)
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597: warning:
[array.access.unsafe.low] Potentially unsafe array access: the index could be negative.
                b[cols[j]], coltypes[cols[j]]);
                  ^
  found   : @LowerBoundUnknown int
  required: an integer >= 0 (@NonNegative or @Positive)
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597: warning:
```

```
[array.access.unsafe.high] Potentially unsafe array access: the index could be larger than the
array's bound
                  b[cols[j]], coltypes[cols[j]]);
                       ^
  found   : @UpperBoundUnknown int
  required: @IndexFor("this.b") or @LTLengthOf("this.b") -- an integer less than this.b's length
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597: warning:
[array.access.unsafe.high.range] Potentially unsafe array access: the index could be larger than
the array's bound
                  b[cols[j]], coltypes[cols[j]]);
                       ^
  index type found: @IntRange(from=-2147483648) int
  array type found: @UnknownVal int @UnknownVal []
  required        : index of type @IndexFor("this.cols") or @LTLengthOf("this.cols"), or array of
type @MinLen(-9223372036854775808)
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597: warning:
[array.access.unsafe.low] Potentially unsafe array access: the index could be negative.
                  b[cols[j]], coltypes[cols[j]]);
                                      ^
  found   : @LowerBoundUnknown int
  required: an integer >= 0 (@NonNegative or @Positive)
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597: warning:
[array.access.unsafe.high] Potentially unsafe array access: the index could be larger than the
array's bound
                  b[cols[j]], coltypes[cols[j]]);
                                      ^
  found   : @UpperBoundUnknown int
  required: @IndexFor("this.coltypes") or @LTLengthOf("this.coltypes") -- an integer less than
this.coltypes's length
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597: warning:
[array.access.unsafe.high.range] Potentially unsafe array access: the index could be larger than
the array's bound
                  b[cols[j]], coltypes[cols[j]]);
                                       ^
  index type found: @IntRange(from=-2147483648) int
  array type found: @UnknownVal int @UnknownVal []
  required        : index of type @IndexFor("this.cols") or @LTLengthOf("this.cols"), or array of
type @MinLen(-9223372036854775808)
```

These warnings all come from four missing facts:

- fieldcount is equal to the length of the cols[] array
- elements of the cols array are indices for the a, b, and coltypes arrays

Both of those are easy to express with the Index Checker, which would remove the warnings, so we can consider them FPs. Tbh, I'm a bit surprised that OpenJML didn't warn on this. I bet it timed out. (I checked, it did. Nice.)

typestate:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597:
warning: Cannot call [#helpers.arrayAccess] on Shared{int[]} | Null
                b[cols[j]], coltypes[cols[j]]);
                 ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:596:
warning: Cannot call [#helpers.arrayAccess] on Shared{int[]} | Null
          int i = Column.compare(session.database.collation, a[cols[j]],
                                                              ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597:
warning: Cannot call [#helpers.arrayAccess] on Shared{int[]} | Null
                b[cols[j]], coltypes[cols[j]]);
                   ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597:
warning: Cannot call [#helpers.arrayAccess] on Shared{int[]} | Null
                b[cols[j]], coltypes[cols[j]]);
                                    ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:596:
warning: Cannot access field [database] of null
          int i = Column.compare(session.database.collation, a[cols[j]],
                                        ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:596:
warning: Cannot call [#helpers.arrayAccess] on Shared{int[]} | Null
          int i = Column.compare(session.database.collation, a[cols[j]],
                                                                 ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:597:
warning: Cannot call [#helpers.arrayAccess] on Shared{int[]} | Null
                b[cols[j]], coltypes[cols[j]]);
                                 ^
```

All are false positives: all result from missing specifications about nullability.

dataset 3, snippet 29

3 warnings, all from typestate

lines 888-893 of Tasks_1

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:891:
warning: Cannot assign because [this.outlen] is not accessible here
        outlen = parameters.length;
        ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:892:
warning: Cannot assign because [this.offset] is not accessible here
        offset = 0;
        ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:891:
warning: Cannot access [this.parameters.length]
        outlen = parameters.length;
                           ^
```

All false positives due to ownership model. Needs annotations.

VERIFIED

dataset 3, snippet 45

6 warnings, all from typestate checker

lines 469-474 of Tasks_2

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:472:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.COLUMNS]
        final JTextField inputWidth = new JTextField(Integer.toString(DEFAULT_WIDTH),
COLUMNS);
                                                                                     ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:472:
warning: Incompatible parameter: cannot cast from Shared{java.lang.String} | Null to
Shared{java.lang.String}
        final JTextField inputWidth = new JTextField(Integer.toString(DEFAULT_WIDTH),
COLUMNS);
                                                                    ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:473:
warning: Incompatible parameter: cannot cast from Shared{java.lang.String} | Null to
Shared{java.lang.String}
        final JTextField inputHeight = new JTextField(Integer.toString(DEFAULT_HEIGHT),
COLUMNS);
                                                                     ^
```

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:472:
warning: Cannot access
[cog_complexity_validation_datasets.Three.Tasks_2.DEFAULT_WIDTH]
     final JTextField inputWidth = new JTextField(Integer.toString(DEFAULT_WIDTH),
COLUMNS);
                                           ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:473:
warning: Cannot access
[cog_complexity_validation_datasets.Three.Tasks_2.DEFAULT_HEIGHT]
     final JTextField inputHeight = new JTextField(Integer.toString(DEFAULT_HEIGHT),
COLUMNS);
                                            ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:473:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.COLUMNS]
     final JTextField inputHeight = new JTextField(Integer.toString(DEFAULT_HEIGHT),
COLUMNS);
                                                      ^
```

All access errors are false positives and just require annotations.

VERIFIED

dataset 3, snippet 62

5 warnings:
- 3 from TS
- 2 from OpenJML

lines 679-686 of Tasks_2

TS:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:683:
warning: Unsafe cast
       dataServiceId = (String) in.readObject();
                 ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:683:
warning: Cannot call readObject on null
       dataServiceId = (String) in.readObject();
                        ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:683:
warning: Cannot assign because [this.dataServiceId] is not accessible here
       dataServiceId = (String) in.readObject();
                ^
```

All three are false positives: the cast is redundant with javac; in is a field that needs a nonnull annotation; and dataServiceId is clearly intended to be accessible here (and therefore needs an annotation).

VERIFIED

OpenJML:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:683: verify: The prover cannot
establish an assertion (PossiblyBadCast) in method s62: a java.lang.Object cannot be proved to
be a java.lang.String
        dataServiceId = (String) in.readObject();
                        ^

cog_complexity_validation_datasets/Three/Tasks_2.java:684: verify: The prover cannot
establish an assertion (PossiblyNullDeReference) in method s62
        channelId = "" + in.readInt();
                           ^
```

Both FPs: redundant with javac on the cast, in needs a spec on the second one.

VERIFIED

dataset 3, snippet 72

lines 502-520 in Tasks_3

3 warnings:

- 2 from TS
- 1 from OpenJML

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:509:
warning: Unsafe cast
        JLabel label = (JLabel) super.getListCellRendererComponent(list, value, index,
isSelected, cellHasFocus);
                       ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:509:
warning: Unsafe cast
        JLabel label = (JLabel) super.getListCellRendererComponent(list, value, index,
isSelected, cellHasFocus);
                       ^
cog_complexity_validation_datasets/Three/Tasks_3.java:509: verify: The prover cannot
establish an assertion (PossiblyBadCast) in method getListCellRendererComponent: a
java.lang.Object cannot be proved to be a javax.swing.JLabel
```

```
        JLabel label = (JLabel) super.getListCellRendererComponent(list, value, index,
isSelected, cellHasFocus);
                        ^
```

All false positives: these casts are javac FPs.

VERIFIED

dataset 3, snippet 80

1 warning, from OpenJML

lines 612-620 of Tasks_3

```
cog_complexity_validation_datasets/Three/Tasks_3.java:615: verify: The prover cannot
establish an assertion (UndefinedCalledMethodPrecondition:
/home/authors/openjml/specs/java/lang/String.jml:447:) in method s99
      // we're still in the game then log an error because at this
                                                         ^
Note: Call stack
  /home/authors/openjml/specs/java/lang/String.jml:458: java.lang.String.equals
  /home/authors/openjml/specs/java/lang/StringBuffer.jml:35: java.lang.String.equals
  cog_complexity_validation_datasets/Three/Tasks_3.java:885: java.lang.StringBuffer.<init>
/home/authors/openjml/specs/java/lang/String.jml:447: verify: Associated declaration:
cog_complexity_validation_datasets/Three/Tasks_3.java:615:
    @ public static model helper boolean equals(nullable String s1, nullable String s2);
                                    ^
/home/authors/openjml/specs/java/lang/String.jml:436: verify: Precondition conjunct is false:
s1.charArray != null
    @   //-RAC@ requires s1.charArray != null & s2.charArray != null; // OPENJML: TODO
Needs the invariant that charArray is not null
                                  ^
/home/authors/openjml/specs/java/lang/String.jml:439: verify: Precondition conjunct is false: s1
== s2
    @   requires s1 == s2;
                  ^
/home/authors/openjml/specs/java/lang/String.jml:442: verify: Precondition conjunct is false: s1
== null || s2 == null || (java.lang.String.isInterned(s1) && java.lang.String.isInterned(s2))
    @   requires s1 == null || s2 == null ||
                                    ^
```

This is pretty obviously a false positive, because the implicated line is a comment? Not sure what's going on here. This may be a bug? Notice that the warning says: in method s99. Since we aren't sure how to interpret this one, we decided to keep it at the location where it's reported (i.e., in this snippet, not s99) and treat it as a false positive (as it clearly isn't reporting about a real problem in s80).

dataset 3, snippet 63

lines 691-698

5 warnings:

- 1 from each tool, except TS, which has 2

TS:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:694:
warning: Cannot assign: cannot cast from Shared{java.lang.Object} | Null to
Shared{cog_complexity_validation_datasets.Three.Tasks_2.Method} | Null
        for (Method method : testMethods)
                    ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:697:
warning: Incompatible return value: cannot cast from Null to
Shared{cog_complexity_validation_datasets.Three.Tasks_2.Description}
        return null; // Added to allow compilation
        ^
```

The first of these is definitely an FP. The other, and the warnings from OpenJML and the CF (after this comment), are more murky: the method does actually return null. The warnings could be avoided with annotations, but the cause is the `return null` that we added to enable compilation. Because they can be avoided by adding an annotation or spec, we consider them FPs.

OpenJML:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:697: verify: Associated method exit
        return null; // Added to allow compilation
        ^
```

CF:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_2.java:697: warning: [return]
incompatible types in return.
        return null; // Added to allow compilation
               ^
  type of expression: null (NullType)
  method return type: @Initialized @NonNull Description
```

Infer:
```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:695:
error: Null Dereference
  object `spec` last assigned on line 692 could be null and is dereferenced at line 695.
  693.          List<Method> testMethods = fTestMethods;
  694.          for (Method method : testMethods)
  695. >            spec.addChild(methodDescription(method));
  696.
  697.          return null; // Added to allow compilation
```

TRUE POSITIVE: spec is only null because of our stub

dataset 3, snippet 57

8 warnings:

- 3 from TS
- 1 from infer
- 4 from openJML

lines 629-634 in Tasks_2

TS:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:630:
warning: Cannot call readObject on null
        String classname = (String) in.readObject();
                                    ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:630:
warning: Unsafe cast
        String classname = (String) in.readObject();
                           ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:631:
warning: Unsafe cast
        String devname = (String)in.readObject();
                         ^
```

All fps: in is clearly supposed to be nonnull (and needs to be annotated as such); and the other two are javac FPs.

Infer:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:633:
error: Null Dereference
  object returned by `getInstance()` could be null and is dereferenced at line 633.
```

```
631.           String devname = (String)in.readObject();
632.
633. >         DeviceIf dev = DriverFactory.getInstance().createDevice(classname, devname);
634.       } // Added to allow compilation
635.
```

TRUE POSITIVE: stubbing

OpenJML:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:630: verify: The prover cannot
establish an assertion (PossiblyBadCast) in method s57: a java.lang.Object cannot be proved to
be a java.lang.String
        String classname = (String) in.readObject();
                           ^
cog_complexity_validation_datasets/Three/Tasks_2.java:631: verify: The prover cannot
establish an assertion (PossiblyNullDeReference) in method s57
        String devname = (String)in.readObject();
                                   ^
cog_complexity_validation_datasets/Three/Tasks_2.java:631: verify: The prover cannot
establish an assertion (PossiblyBadCast) in method s57: a java.lang.Object cannot be proved to
be a java.lang.String
        String devname = (String)in.readObject();
                         ^
cog_complexity_validation_datasets/Three/Tasks_2.java:633: verify: The prover cannot
establish an assertion (InvariantEntrance:
/home/authors/openjml/specs/java/lang/CharSequence.jml:30:) in method s57: (Caller:
cog_complexity_validation_datasets.Three.Tasks_2.s57(), Callee:
cog_complexity_validation_datasets.Three.Tasks_2.DriverFactory.createDevice(java.lang.String
,java.lang.String))
        DeviceIf dev = DriverFactory.getInstance().createDevice(classname, devname);
                                                               ^
```

All are FPs. The first and third warnings are redundant with javac. The fourth is the string FP from OpenJML. The second indicates a missing spec.

VERIFIED

dataset 3, snippet 28

3 warnings, all from TS.

lines 876-884 in Tasks_1

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:877:
warning: Cannot call iterator on null
        for (Iterator<Runner> iter= fRunners.iterator(); iter.hasNext();) {
                                    ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:878:
warning: Cannot assign: cannot cast from Shared{java.lang.Object} | Null to
Shared{cog_complexity_validation_datasets.Three.Tasks_1.Runner} | Null
            Runner runner = iter.next();
                   ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:877:
warning: Cannot call hasNext on null
        for (Iterator<Runner> iter= fRunners.iterator(); iter.hasNext();) {
                                                     ^
```

There is no indication that any of these might be nullable. So, I think this is a need for specifications indicating that they're nonnull.

VERIFIED

dataset 3, snippet 2

9 warnings:

- 2 from the CF
- 2 from openjml
- 5 from the typestate checker

lines 546-560 of Tasks_1

CF:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:551: warning: [return]
incompatible types in return.
                return null;
                       ^
  type of expression: null (NullType)
  method return type: @Initialized @NonNull Object

/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:555: warning:
[array.access.unsafe.high.constant] Potentially unsafe array access: the constant index 0 could
be larger than the array's bound
                Action action = menu.getSubItems()[0].getAction();
                                                   ^
  found   : @UnknownVal AbstractButton @UnknownVal []
  required: @MinLen(1) -- an array guaranteed to have at least 1 elements
```

Both are FPs. The first is caused because there is no @Nullable annotation on the return type. The second is caused by the CF not being aware that the getSubitems() method is pure.

OpenJML:

```
cog_complexity_validation_datasets/Three/Tasks_1.java:555: verify: The prover cannot
establish an assertion (PossiblyTooLargeIndex) in method s2
            Action action = menu.getSubItems()[0].getAction();
                                              ^

cog_complexity_validation_datasets/Three/Tasks_1.java:551: verify: Associated method exit
            return null;
            ^
```

These warnings are exactly the same as the CF warnings, and have the same causes, so they're also FPs.

TS:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:555:
warning: Cannot call getAction on null
            Action action = menu.getSubItems()[0].getAction();
                                                ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:551:
warning: Incompatible return value: cannot cast from Null to Shared{java.lang.Object}
            return null;
            ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:548:
warning: Cannot assign: cannot cast from Shared{java.lang.Object} | Null to
Shared{cog_complexity_validation_datasets.Three.Tasks_1.ActionMenu} | Null
         ActionMenu menu = actionList.get(0);
                    ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:548:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_1.actionList]
         ActionMenu menu = actionList.get(0);
                           ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:547:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_1.actionList]
      if (actionList.size() == 1) {
         ^
```

These are all false positives. The first is caused by a lack of nullness specs for getSubitems. The second is like the warnings on the same line from the other two checkers. The third is totally nonsensical and seems to be caused by type erasure. The last two are about accessing fields, which indicate missing annotations.

VERIFIED

dataset 3, snippet 19

3 warnings, all from TS

lines 771-777 of Tasks_1

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:776:
warning: Cannot assign because [this.constType] is not accessible here
        constType         = type;
                  ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:774:
warning: Cannot assign because [this.core] is not accessible here
        core              = new ConstraintCore();
                  ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:775:
warning: Cannot assign because [this.constName] is not accessible here
        constName         = name;
                  ^
```

All FPs indicating missing annotations to allow mutation.

VERIFIED

dataset 3, snippet 98

3 warnings:

- 2 from TS
- 1 from OpenJML

lines 870-875 in Tasks_3

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:871:
warning: Incompatible parameter: cannot cast from Shared{java.lang.Object} | Null to
Shared{java.lang.Object}
```

```
        Description description= Description.createSuiteDescription(name);
```

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:872:
warning: Cannot call testCount on null
        int n= ts.testCount();
                 ^
```

Both FPs indicating missing nullability annotations.

OpenJML:

```
cog_complexity_validation_datasets/Three/Tasks_3.java:872: verify: The prover cannot
establish an assertion (PossiblyNullDeReference) in method s98
        int n= ts.testCount();
                 ^
```

Same.

VERIFIED

dataset 3, snippet 54

4 warnings, all from TS

lines 591-598 in Tasks_2

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:595:
warning: Incompatible parameter: cannot cast from
Shared{cog_complexity_validation_datasets.Three.Tasks_2.Ns} | Null to
Shared{cog_complexity_validation_datasets.Three.Tasks_2.Ns}
        addColumn(t, "PROCEDURE_CAT", Types.VARCHAR);
                  ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:595:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.Types.VARCHAR]
        addColumn(t, "PROCEDURE_CAT", Types.VARCHAR);
                                           ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:597:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.Types.VARCHAR]
        addColumn(t, "PROCEDURE_NAME", Types.VARCHAR, false);    // not null
                                            ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:596:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.Types.VARCHAR]
        addColumn(t, "PROCEDURE_SCHEM", Types.VARCHAR);
```

```
                                ^
```

All FPs. The first appears to be concerned about the receiver parameter being nullable? Or maybe t? The former is 100% impossible; the latter there is no indication of (and would be solvable with annotations regardless). The other three are access errors, which could easily be solved via annotation.

VERIFIED

dataset 3, snippet 73

2 warnings, both from TS

lines 524-533 in Tasks_3

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:526:
warning: Incompatible parameter: cannot cast from Shared{java.lang.String} | Null to
Shared{java.lang.String}
        return Assert.format(message, fExpected, fActual);
                                      ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:526:
warning: Incompatible parameter: cannot cast from Shared{java.lang.String} | Null to
Shared{java.lang.String}
        return Assert.format(message, fExpected, fActual);
                             ^
```

These two warnings are about fExpected and fActual, which definitely are nullable at this point. However, the code *explicitly* checks for their nullability before making this call, so this code must be assuming that Assert#format can take nullable parameters. The TS checker is missing that info: that is, it needs more annotations. The CF not warning about this guarantees that fact, actually, because it would warn by default: however, it must have a built-in model for Assert#format that permits nullability.

VERIFIED

dataset 3, snippet 84

6 warnings, all from TS

lines 671-679 in Tasks_3

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:678:
warning: Cannot assign because [this.core.refColArray] is not accessible here
```

```
        core.refColArray  = refCols;
                          ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:677:
warning: Cannot access [this.core.colLen]
        core.colLen       = core.mainColArray.length;
            ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:678:
warning: Cannot access [this.core.refColArray]
        core.refColArray  = refCols;
            ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:676:
warning: Cannot assign because [this.core.mainColArray] is not accessible here
        core.mainColArray = mainCols;
                          ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:676:
warning: Cannot access [this.core.mainColArray]
        core.mainColArray = mainCols;
            ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_3.java:677:
warning: Cannot assign because [this.core.colLen] is not accessible here
        core.colLen       = core.mainColArray.length;
                          ^
```

All FPs related to accessibility and missing annotations about it.

VERIFIED

dataset 3, snippet 20

1 warning from TS

lines 781-786

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:782:
warning: Cannot assign: cannot cast from Null | Primitive{int} to Primitive{int}
        int eventId = active? ON : OFF;
            ^
```

ON and OFF are primitives and cannot be null, so this is obviously an FP.

VERIFIED

dataset 3, snippet 41

6 warnings, all from TS

lines 426-434 of Tasks_2

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:429:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.EQ]
            t = jj_consume_token(EQ);
                                 ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:429:
warning: Cannot assign because [this.t] is not accessible here
            t = jj_consume_token(EQ);
            ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:428:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.EQ]
          case EQ:
               ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:431:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.NE]
          case NE:
               ^

simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:432:
warning: Cannot access [cog_complexity_validation_datasets.Three.Tasks_2.NE]
            t = jj_consume_token(NE);
                                 ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:432:
warning: Cannot assign because [this.t] is not accessible here
            t = jj_consume_token(NE);
            ^
```

All are access warnings, which are all FPs.

VERIFIED

dataset 3, snippet 55

6 warnings, all from TS

lines 602-612 of Tasks_2

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:606:
warning: Incompatible parameter: cannot cast from Null | Primitive{boolean} to
Primitive{boolean}
        loadMissionChip(gc, color, expertMission);
                                   ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:606:
warning: Incompatible parameter: cannot cast from Shared{java.awt.GraphicsConfiguration} |
Null to Shared{java.awt.GraphicsConfiguration}
        loadMissionChip(gc, color, expertMission);
                        ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:605:
warning: Cannot call getDefaultConfiguration on null
             .getDefaultConfiguration();
             ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:609:
warning: Cannot assign because [this.missionChip] is not accessible here
           missionChip = expertMissionChips.get(color);
                       ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:604:
warning: Cannot call getDefaultScreenDevice on null
        GraphicsConfiguration gc =
GraphicsEnvironment.getLocalGraphicsEnvironment().getDefaultScreenDevice()
                                                 ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:606:
warning: Incompatible parameter: cannot cast from Shared{java.lang.Object} | Null to
Shared{java.lang.Object}
        loadMissionChip(gc, color, expertMission);
                            ^
```

All FPs. The nullability warnings all indicate missing annotations. The access warnings do, too.

VERIFIED

dataset 3, snippet 60

3 warnings:

- 2 from TS
- 1 from OpenJML

lines 658-662 of Tasks_2

TS:

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:659:
warning: Cannot call getResponseBoolean on null
        boolean response = warehouseDialog.getResponseBoolean();
                                          ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_2.java:661:
warning: Incompatible return value: cannot cast from Primitive{boolean} to
Shared{java.lang.Object}
        return response;
        ^
```

Both FPs. First is a missing nullability annotation. Second is just false: you certainly can cast from boolean to Object (such casts are automatic and safe).

OpenJML:

```
cog_complexity_validation_datasets/Three/Tasks_2.java:660: verify: The prover cannot
establish an assertion (NullFormal:
cog_complexity_validation_datasets/Three/Tasks_2.java:1680:) in method s60:
warehouseDialog in remove(cog_complexity_validation_datasets.Three.Tasks_2.WareHouse)
        remove(warehouseDialog);
               ^
```

Missing nonnull spec on warehouseDialog; resolvable by annotation. FP.

VERIFIED

dataset 3, snippet 26

31 warnings:

- 9 from the CF
- 5 from the TS checker
- 17 from openJML

lines 851-859 in Tasks_1

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:853:
warning: Incompatible parameter: cannot cast from Shared{java.lang.Object} | Null to
Shared{java.lang.Object}
            System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
                                                      ^
```

```
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:855:
warning: Incompatible parameter: cannot cast from Shared{java.lang.Object} | Null to
Shared{java.lang.Object}
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                                                              ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:853:
warning: Incompatible parameter: cannot cast from Null | Primitive{int} to Primitive{int}
        System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
                                                           ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:853:
warning: Incompatible parameter: cannot cast from Shared{java.lang.Object} | Null to
Shared{java.lang.Object}
        System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
                         ^
simple-datasets/src/main/java/cog_complexity_validation_datasets/Three/Tasks_1.java:855:
warning: Incompatible parameter: cannot cast from Shared{java.lang.Object} | Null to
Shared{java.lang.Object}
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                         ^
```

All FPs. Nullability is not a problem here, especially for ints: ints cannot be null, and for the rest, annotations could easily resolve them. The TS checker is really missing the complexity of this method.

CF:

```
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:853: warning: [argument]
incompatible argument for parameter arg1 of arraycopy.
        System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
                                        ^
  found   : @LowerBoundUnknown int
  required: @NonNegative int
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:853: warning: [argument]
incompatible argument for parameter arg4 of arraycopy.
        System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
                                                           ^
  found   : @LowerBoundUnknown int
  required: @NonNegative int
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:853: warning: [argument]
incompatible argument for parameter arg4 of arraycopy.
        System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
```

```
                                              ^
  found   : @UpperBoundUnknown int
  required: @LTLengthOf(offset={"this.bufpos - this.len + 1 - 1", "0 - 1"}, value={"this.buffer",
"this.ret"}) int
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:855: warning: [argument]
incompatible argument for parameter arg1 of arraycopy.
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                                      ^
  found   : @LowerBoundUnknown int
  required: @NonNegative int
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:856: warning: [argument]
incompatible argument for parameter arg4 of arraycopy.
              len - bufpos - 1);
                      ^
  found   : @LowerBoundUnknown int
  required: @NonNegative int
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:856: warning: [argument]
incompatible argument for parameter arg4 of arraycopy.
              len - bufpos - 1);
                      ^
  found   : @UpperBoundUnknown int
  required: @LTLengthOf(offset={"this.bufsize - (len - bufpos - 1) - 1", "0 - 1"},
value={"this.buffer", "this.ret"}) int
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:857: warning: [argument]
incompatible argument for parameter arg3 of arraycopy.
        System.arraycopy(buffer, 0, ret, len - bufpos - 1, bufpos + 1);
                                            ^
  found   : @LowerBoundUnknown int
  required: @NonNegative int
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:857: warning: [argument]
incompatible argument for parameter arg4 of arraycopy.
        System.arraycopy(buffer, 0, ret, len - bufpos - 1, bufpos + 1);
                                                    ^
  found   : @LowerBoundUnknown int
  required: @NonNegative int
/home/authors/Code-Complexity-Research/complexity-verification-project/simple-datasets/src/m
ain/java/cog_complexity_validation_datasets/Three/Tasks_1.java:857: warning: [argument]
incompatible argument for parameter arg4 of arraycopy.
        System.arraycopy(buffer, 0, ret, len - bufpos - 1, bufpos + 1);
```

```
                                                ^
  found   : @UpperBoundUnknown int
  required: @LTLengthOf(offset={"0 - 1", "this.len - this.bufpos - 1 - 1"}, value={"this.buffer",
"this.ret"}) int
```

All FPs: all of these warnings are caused by missing annotations on the various integer values involved (`len`, `bufpos`). System.arraycopy has a very difficult-to-satisfy specification (because it is a direct wrapper of the C standard library's memcopy!), so given arbitrary integers as the lengths, as here, it is unsurprising that the CF warns.

OpenJML:

```
cog_complexity_validation_datasets/Three/Tasks_1.java:855: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: overflow in int difference
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                               ^
cog_complexity_validation_datasets/Three/Tasks_1.java:855: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: overflow in int difference
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                                    ^
cog_complexity_validation_datasets/Three/Tasks_1.java:855: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: underflow in int difference
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                                          ^
cog_complexity_validation_datasets/Three/Tasks_1.java:855: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: underflow in int difference
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                               ^
cog_complexity_validation_datasets/Three/Tasks_1.java:856: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: overflow in int difference
             len - bufpos - 1);
                ^
cog_complexity_validation_datasets/Three/Tasks_1.java:855: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: underflow in int difference
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                                    ^
cog_complexity_validation_datasets/Three/Tasks_1.java:855: verify: The prover cannot
establish an assertion (Precondition: /home/authors/openjml/specs/java/lang/System.jml:312:) in
method s26
        System.arraycopy(buffer, bufsize - (len - bufpos - 1), ret, 0,
                   ^
cog_complexity_validation_datasets/Three/Tasks_1.java:856: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: underflow in int difference
             len - bufpos - 1);
```

```
                            ^
cog_complexity_validation_datasets/Three/Tasks_1.java:852: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: overflow in int sum
        if ((bufpos + 1) >= len)
                  ^
cog_complexity_validation_datasets/Three/Tasks_1.java:853: verify: The prover cannot
establish an assertion (Precondition: /home/authors/openjml/specs/java/lang/System.jml:312:) in
method s26
            System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
                            ^
cog_complexity_validation_datasets/Three/Tasks_1.java:853: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: overflow in int difference
            System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
                                            ^
cog_complexity_validation_datasets/Three/Tasks_1.java:856: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: underflow in int difference
                    len - bufpos - 1);
                        ^
cog_complexity_validation_datasets/Three/Tasks_1.java:857: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: underflow in int difference
            System.arraycopy(buffer, 0, ret, len - bufpos - 1, bufpos + 1);
                                                          ^
cog_complexity_validation_datasets/Three/Tasks_1.java:857: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: overflow in int difference
            System.arraycopy(buffer, 0, ret, len - bufpos - 1, bufpos + 1);
                                                 ^
cog_complexity_validation_datasets/Three/Tasks_1.java:853: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: overflow in int sum
            System.arraycopy(buffer, bufpos - len + 1, ret, 0, len);
                                                  ^
cog_complexity_validation_datasets/Three/Tasks_1.java:857: verify: The prover cannot
establish an assertion (Precondition: /home/authors/openjml/specs/java/lang/System.jml:312:) in
method s26
            System.arraycopy(buffer, 0, ret, len - bufpos - 1, bufpos + 1);
                            ^
cog_complexity_validation_datasets/Three/Tasks_1.java:857: verify: The prover cannot
establish an assertion (ArithmeticOperationRange) in method s26: overflow in int sum
            System.arraycopy(buffer, 0, ret, len - bufpos - 1, bufpos + 1);
                                                                     ^
cog_complexity_validation_datasets/Three/Tasks_1.java:851: verify: Validity is unknown - time
or memory limit reached: : Aborted proof: timeout
    public void s26() {
                ^
```

All are FPs:

- the warnings about System.jml correspond to the warnings from the CF.
- the warnings about over- and under-flow are borderline, but definitely resolvable with annotations. They cannot be true positives, since java arrays are 32-bit addressable but ints are 64 bit on modern machines.

VERIFIED