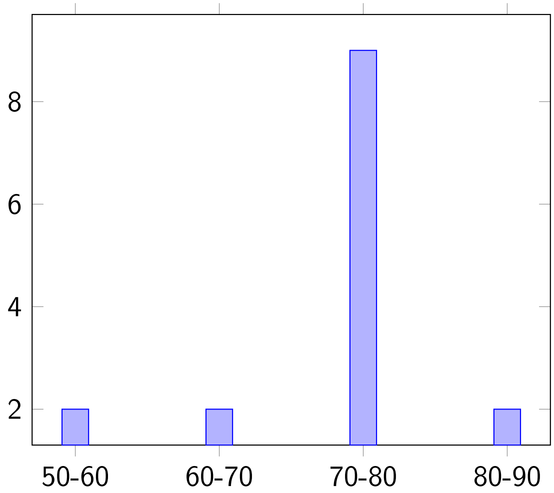
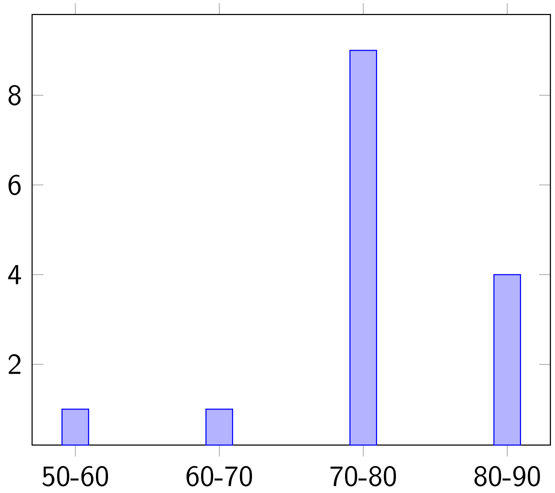
Midterm: grade distribution



Average: 73%

Overall grade distribution



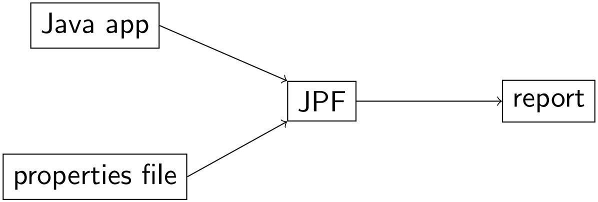
Average: 77%

Testing JPF

EECS 4315

`www.eecs.yorku.ca/course/4315/`

Java PathFinder (JPF)



Testing JPF

Question

In order to test a part of JPF, how many pieces of input do we need to provide and what type of input?

Testing JPF

Question

In order to test a part of JPF, how many pieces of input do we need to provide and what type of input?

Answer

Two: a Java app and an application properties file.

Failed assertions

We want to test whether JPF can detect failed assertions.

Question

Write a Java app that can be used to test whether JPF can detect failed assertions.

Failed assertions

We want to test whether JPF can detect failed assertions.

Question

Write a Java app that can be used to test whether JPF can detect failed assertions.

Answer

```
public class FailedAssertion {
  public static void main(String[] args) {
    assert false;
  }
}
```

Failed assertions

We want to test whether JPF can detect failed assertions.

Question

Write the corresponding application properties file.

Failed assertions

We want to test whether JPF can detect failed assertions.

Question

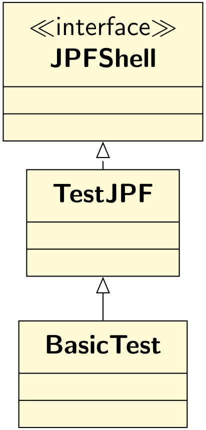
Write the corresponding application properties file.

Answer

```
target=FailedAssertion
classpath=.
```

JPF testing framework

JPF provides a framework very similar to JUnit so that the body of the `main` method of the Java app and the content of the application properties file can be combined into a single unit test.



Skeleton of a test case

```
import gov.nasa.jpf.util.test.TestJPF;
import org.junit.Test;

public class ...Test extends TestJPF {

  /**
   * Runs the test methods with the given names.
   * If no names are given, all test methods are run.
   *
   * @param testMethods the names of the test methods.
   */
  public static void main(String[] testMethods) {
    TestJPF.runTestsOfThisClass(testMethods);
  }
}
```

Skeleton of a unit test

```
@Test
public void ...Test() {
  if (this.verify...(application properties)) {
       // body of main method of app
  }
}
```

Example of a unit test

```
@Test
public void assertTest() {
  if (this.verifyAssertionError("+classpath=.")) {
    assert false;
  }
}
```

Example of a unit test

```
@Test
public void assertTest() {
  if (this.verifyAssertionError(    "+classpath=."    )) {
                                  application properties
    assert false;
  }
}
```

Example of a unit test

```
@Test
public void assertTest() {
  if (this.verifyAssertionError("+classpath=.")) {
```

$$\underbrace{\texttt{assert false;}}_{\texttt{body}}$$

```
  }
}
```

Running a test case

To run the `BasicTest` from the command line, use

```
java -cp <path to>/jpf.jar:. BasicTest
```

It produces output similar to the following.

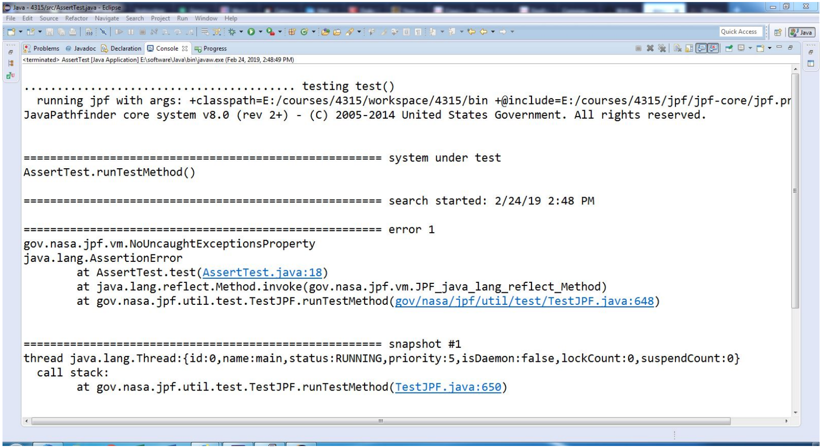
```
................................... testing assertTes
running jpf with args: +classpath=.
JavaPathfinder core system v8.0 (rev ..) - (C) 2005-2014 Un
...

========================================== sear
................... test: Ok

................... execution of testsuite: AssertTest
.... [1] test: Ok
................... tests: 1, failures: 0, errors: 0
```

Running a test case

The class can also be run as an app in eclipse. In that case, you may have to add

`"+@include=<path to>jpf/jpf-core/jpf.properties"`

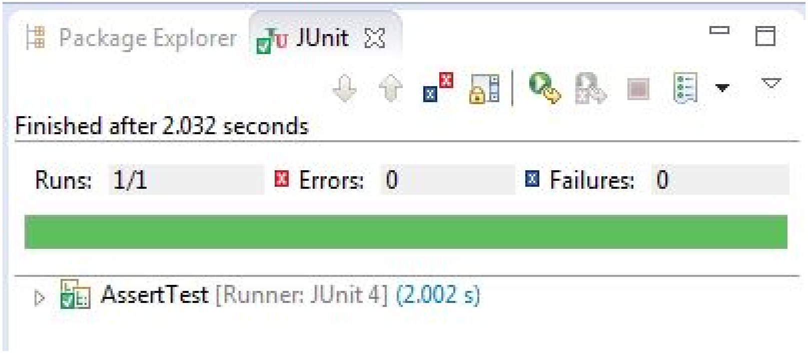
as a second argument of the invocation of `verifyAssertionError`.


Java - 4315/src/AssertTest.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java
Problems @ Javadoc Declaration Console Progress

```
<terminated> AssertTest [Java Application] E:\software\Java\bin\javaw.exe (Feb 24, 2019, 2:48:49 PM)

......................................... testing test()
  running jpf with args: +classpath=E:/courses/4315/workspace/4315/bin +@include=E:/courses/4315/jpf/jpf-core/jpf.pr
JavaPathfinder core system v8.0 (rev 2+) - (C) 2005-2014 United States Government. All rights reserved.


====================================================== system under test
AssertTest.runTestMethod()

====================================================== search started: 2/24/19 2:48 PM

====================================================== error 1
gov.nasa.jpf.vm.NoUncaughtExceptionsProperty
java.lang.AssertionError
        at AssertTest.test(AssertTest.java:18)
        at java.lang.reflect.Method.invoke(gov.nasa.jpf.vm.JPF_java_lang_reflect_Method)
        at gov.nasa.jpf.util.test.TestJPF.runTestMethod(gov/nasa/jpf/util/test/TestJPF.java:648)


====================================================== snapshot #1
thread java.lang.Thread:{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
  call stack:
        at gov.nasa.jpf.util.test.TestJPF.runTestMethod(TestJPF.java:650)
```

Running a test case

The class can also be run as a JUnit test in eclipse.



Finished after 2.032 seconds

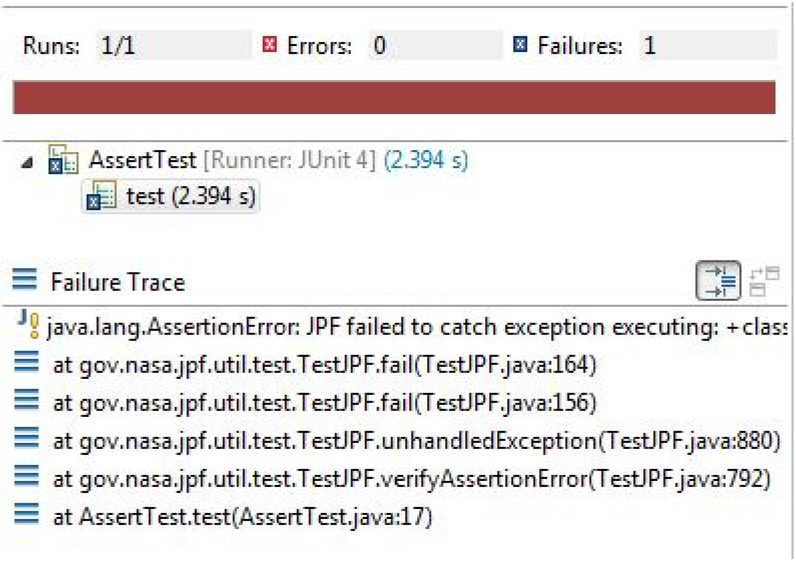
Runs: 1/1 Errors: 0 Failures: 0

AssertTest [Runner: JUnit 4] (2.002 s)

Example of a failing unit test

```
@Test
public void assertTest() {
  if (this.verifyAssertionError("+classpath=.")) {
    assert true;
  }
}
```

Example of a failing unit test



```
Runs: 1/1     Errors: 0     Failures: 1

AssertTest [Runner: JUnit 4] (2.394 s)
    test (2.394 s)

Failure Trace
java.lang.AssertionError: JPF failed to catch exception executing: +class
at gov.nasa.jpf.util.test.TestJPF.fail(TestJPF.java:164)
at gov.nasa.jpf.util.test.TestJPF.fail(TestJPF.java:156)
at gov.nasa.jpf.util.test.TestJPF.unhandledException(TestJPF.java:880)
at gov.nasa.jpf.util.test.TestJPF.verifyAssertionError(TestJPF.java:792)
at AssertTest.test(AssertTest.java:17)
```

Unhandled exception

```
@Test
public void exceptionTest() {
  if (this.verifyUnhandledException(
        "java.lang.NullPointerException",
        "+classpath=.")) {
    throw new NullPointerException();
  }
}
```

Unhandled exception

```
@Test
public void exceptionTest() {
  if (this.verifyUnhandledException(
        "java.lang.NullPointerException",
           type of unhandled exception
        "+classpath=.")) {
    throw new NullPointerException();
  }
}
```

Property violation

```
@Test
public void propertyViolationTest() {
  if (this.verifyPropertyViolation(
        new TypeRef("gov.nasa.jpf.vm.NoUncaughtExceptionsPr
        "+classpath=.")) {
     throw new RuntimeException();
  }
}
```

The class `TypeRef` is part of the package `gov.nasa.jpf.util`.

Property violation

```
@Test
public void propertyViolationTest() {
  if (this.verifyPropertyViolation(
        new TypeRef("gov.nasa.jpf.vm.NoUncaughtExceptionsPr
                           type of violated property
        "+classpath=.")) {
    throw new RuntimeException();
  }
}
```

No property violation

```
@Test
public void noPropertyViolationTest() {
  if (this.verifyNoPropertyViolation("+classpath=.")) {
    // nothing
  }
}
```

Testing the listener CallMonitor

The listener `gov.nasa.jpf.listener.CallMonitor` monitors method invocations. When JPF finishes, it publishes for each method invocation, the ID of the thread that executed the method invocation, the depth of the stack, the name of the class, the name of the method, and its arguments.

Testing the listener CallMonitor

The listener `gov.nasa.jpf.listener.CallMonitor` monitors method invocations. When JPF finishes, it publishes for each method invocation, the ID of the thread that executed the method invocation, the depth of the stack, the name of the class, the name of the method, and its arguments.

Problem

How to test the listener CallMonitor.

Sample app

```
public class Example {
  public static void main(String [] args) {
    first(1, true);
  }

  private static void first(int i, boolean b) {
    second(i + 1);
  }

  private static void second(int i) {
    // do nothing
  }
}
```

Sample application properties file

```
target=Example
classpath=.
listener=gov.nasa.jpf.listener.CallMonitor
```

Sample output

```
========================================= method invocations
0:              java.lang.Boolean.<clinit>()
...
0:   Example.main([Ljava.lang.String;@bb)
0:    Example.first(1,true)
0:     Example.second(2)
```

Testing the listener CallMonitor

Question

What is the simplest app on which we can run the listener CallMonitor?

Testing the listener CallMonitor

Question

What is the simplest app on which we can run the listener CallMonitor?

Answer

An app with an empty main method.

Testing the listener CallMonitor

Question

To test the CallMonitor listener, which of the following `verify` methods can we use for the empty app?

- `verifyAssertionError`
- `verifyNoPropertyViolation`
- `verifyPropertyViolation`
- `verifyUnhandledException`

Testing the listener CallMonitor

Question

To test the CallMonitor listener, which of the following `verify` methods can we use for the empty app?

- `verifyAssertionError`
- `verifyNoPropertyViolation`
- `verifyPropertyViolation`
- `verifyUnhandledException`

Answer

`verifyNoPropertyViolation`

Testing the listener CallMonitor

Question

What is the corresponding application properties file?

Testing the listener CallMonitor

Question

What is the corresponding application properties file?

Answer

```
target=CallMonitorTest
classpath=.
listener=gov.nasa.jpf.listener.CallMonitor
```

Testing the listener CallMonitor

Question

Write the corresponding unit test.

Testing the listener CallMonitor

Question

Write the corresponding unit test.

Answer

```
@Test
public void emptyTest() {
  if (this.verifyNoPropertyViolation(
        "+listener=gov.nasa.jpf.listener.CallMonitor",
        "+classpath=.")) {
    // do nothing
  }
}
```

Testing the listener CallMonitor

Question

Consider the following method.

```
private static void staticMethod() {}
```

Write a unit test that invokes this method.

Testing the listener CallMonitor

Question

Consider the following method.

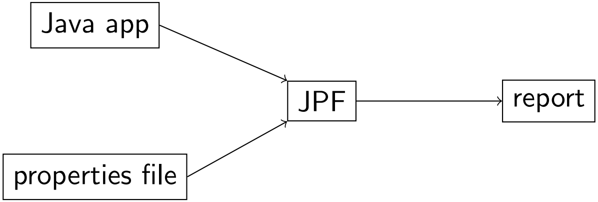
```
private static void staticMethod() {}
```

Write a unit test that invokes this method.

Answer

```
@Test
public void staticMethodTest() {
  if (this.verifyNoPropertyViolation(
        "+listener=gov.nasa.jpf.listener.CallMonitor",
        "+classpath=.")) {
    CallMonitorTest.staticMethod();
  }
}
```

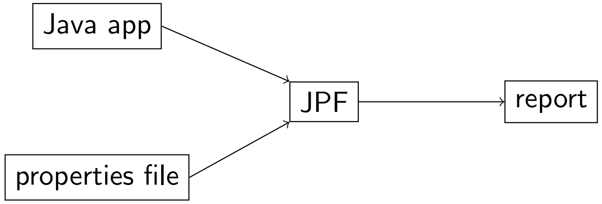
Testing the listener CallMonitor



Question

To test the CallMonitor listener in this way, for what in the report should we look?

Testing the listener CallMonitor



Question

To test the CallMonitor listener in this way, for what in the report should we look?

Answer

The report should contain the string `CallMonitorTest.staticMethod()`.

Testing the listener CallMonitor

Question

To where is the report written?

Testing the listener CallMonitor

Question

To where is the report written?

Answer

The console.

Testing the listener CallMonitor

Question

How can we write to a “string” instead of the console?

Testing the listener CallMonitor

Question

How can we write to a “string” instead of the console?

Answer

```
ByteArrayOutputStream stream = new ByteArrayOutputStream()
System.setOut(new PrintStream(stream));
```

Testing the listener CallMonitor

The class TestJPF contains static methods assert... similar to those found in JUnit.

```
TestJPF.assertTrue("Invocation of staticMethod incorrect",
  stream.toString().contains("CallMonitorTest.staticMethod(
```

Testing the listener CallMonitor

Question

How do we ensure that `TestJPF.assertTrue` write to the console (and not to the `stream`)?

Testing the listener CallMonitor

Question

How do we ensure that `TestJPF.assertTrue` write to the console (and not to the `stream`)?

Answer

```
PrintStream out = System.out; // remember for later
ByteArrayOutputStream stream = new ByteArrayOutputStream()
System.setOut(new PrintStream(stream));
...
System.setOut(out); // reset
TestJPF.assertTrue("Invocation of staticMethod incorrect",
  stream.toString().contains("CallMonitorTest.staticMethod
```

Testing the listener CallMonitor

```
@Test
public void order() {
  System.out.println("1");
  if (this.verifyNoPropertyViolation(...)) {
    System.out.println("2");
  } else {
    System.out.println("3");
  }
}
```

Testing the listener CallMonitor

```
.............................. testing order()
1
running jpf with args: ...
JavaPathfinder core system v8.0 (rev ...) - (C) 2005-2014 U
================================= system under test
BasicTest.runTestMethod()
================================= search started: 03/03/1
1
2
================================= results
no errors detected
================================= search finished: 03/03/
3
.............................. order: Ok
```

Testing the listener CallMonitor

Step 1: black part of executed by the host JVM.

```
@Test
public void order() {
  System.out.println("1");
  if (this.verifyNoPropertyViolation(...))  {
    System.out.println("2");
  } else {
    System.out.println("3");
  }
}
```

Testing the listener CallMonitor

Step 2: black part is model checked by JPF.

```
@Test
public void order() {
  System.out.println("1");
  if (this.verifyNoPropertyViolation(...))  {
    System.out.println("2");
  } else {
    System.out.println("3");
  }
}
```

Testing the listener CallMonitor

Step 3: black part is executed by host JVM.

```
@Test
public void order() {
  System.out.println("1");
  if (this.verifyNoPropertyViolation(...)) {
    System.out.println("2");
  } else {
    System.out.println("3");
  }
}
```

Testing the listener CallMonitor

```
@Test
public void staticMethodTest() {
  PrintStream out = System.out;
  ByteArrayOutputStream stream = new ByteArrayOutputStream(
  System.setOut(new PrintStream(stream));
  if (this.verifyNoPropertyViolation(
        "+listener=gov.nasa.jpf.listener.CallMonitor",
        "+classpath=.")) {
    CallMonitorTest.staticMethod();
  } else {
    System.setOut(out);
    TestJPF.assertTrue("Invocation of staticMethod incorrec
      stream.toString().contains("CallMonitorTest.staticMet
  }
}
```

Testing the listener CallMonitor

To avoid that

```
PrintStream out = System.out;
ByteArrayOutputStream stream = new ByteArrayOutputStream()
System.setOut(new PrintStream(stream));
```

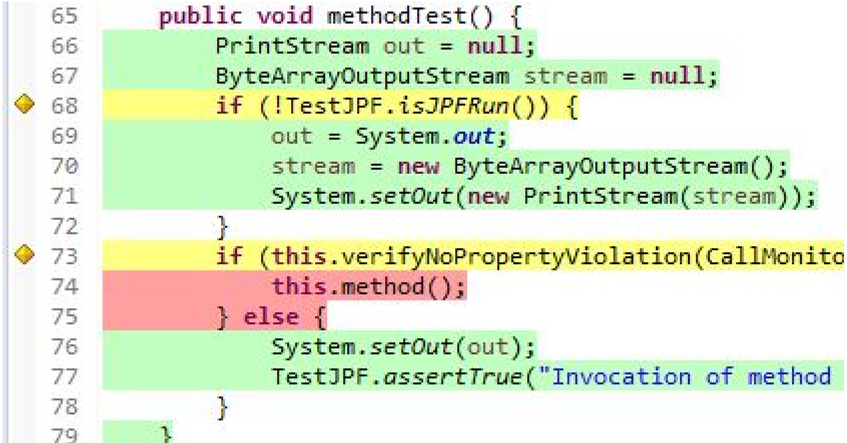
is model checked, we can use

```
PrintStream out = null;
ByteArrayOutputStream stream = null;
if (!TestJPF.isJPFRun()) {
  out = System.out;
  stream = new ByteArrayOutputStream();
  System.setOut(new PrintStream(stream));
}
```

Testing the listener CallMonitor

Use techniques and tools, as discussed in the course EECS 4313 Software Engineering Testing, to develop the test case for the `CallMonitor` listener.

Note that tools such as EclEmma may not provide accurate feedback, as the code is executed in different modes.



```
    public void methodTest() {
        PrintStream out = null;
        ByteArrayOutputStream stream = null;
        if (!TestJPF.isJPFRun()) {
            out = System.out;
            stream = new ByteArrayOutputStream();
            System.setOut(new PrintStream(stream));
        }
        if (this.verifyNoPropertyViolation(CallMonito
            this.method();
        } else {
            System.setOut(out);
            TestJPF.assertTrue("Invocation of method
        }
    }
```