**Assignment 4**

**Restoring a Deleted File in Git: BlackBoxGiven.java**

**1. What would happen if we merged Whitebox into master after deleting BlackBoxGiven.java?**

If we merged the Whitebox branch (where BlackBoxGiven.java was deleted) into the master branch, the file would also be deleted from master. This is not what we want, since we want to keep our test files. This demonstrates how changes in one branch (like deletions) can propagate to other branches during merges, potentially causing unwanted data loss.

**2. Steps to Restore BlackBoxGiven.java in StaticAnalysis Branch**

**a. Switch to the Whitebox branch and create StaticAnalysis branch**

**git checkout Whitebox**

**A black background with white text

AI-generated content may be incorrect.**

**git checkout -b StaticAnalysis**

**A black background with white text

AI-generated content may be incorrect.**

**b. Find the commit where the file was deleted**

**git log --diff-filter=D –summary**

A black background with white text

AI-generated content may be incorrect.****

**c. Restore the file from the commit before deletion**

**git checkout Whitebox~1 src/test/java/BlackBoxGiven.java**

**A black screen with white text

AI-generated content may be incorrect.**

**d. Add and commit the restored file**

**git add src/test/java/BlackBoxGiven.java**

****

**git commit -m "Restore BlackBoxGiven.java in StaticAnalysis branch"**

**A black screen with white text

AI-generated content may be incorrect.**

**Not Running All Test Files**

**Steps Completed:**

1. Modified build.gradle to exclude BlackBoxGiven tests:

test {

exclude '\*\*/BlackBoxGiven.class'

}

1. Successfully ran gradle clean and gradle build on the Blackbox branch

* A screen shot of a computer code

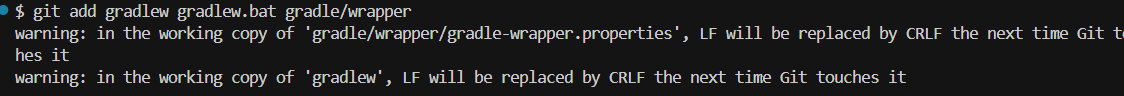
  AI-generated content may be incorrect.The build was successful, showing that BlackBoxGiven tests were properly excluded

1. Merged Blackbox into StaticAnalysis branch

* The merge showed "Already up to date" since both branches had the same changes

A black background with white text

AI-generated content may be incorrect.



A black background with white text

AI-generated content may be incorrect.

**Which branches fail on GitHub Actions? Explain why!**

1. **Whitebox Branch**:

* This branch will fail because it has the BlackBoxGiven.java file deleted
* The build will fail during compilation because the test class is missing

1. **Review Branch**:

* This branch might fail if it doesn't have the Gradle wrapper files
* It might also fail if it has the original BlackBoxGiven.java with references to Game0-Game4 classes

A screenshot of a computer

AI-generated content may be incorrect.**Task 1: Style checking using Checkstyle**

**Task 2: Static Analysis using Spotbug**A screenshot of a computer

AI-generated content may be incorrect.After solving the errors:

A screenshot of a computer

AI-generated content may be incorrect. A screen shot of a computer

AI-generated content may be incorrect.

**Task 4: Putting it together**

**a) What do you get now for these branches?**

The **current branch** (after corrections) shows the following:

* **SpotBugs:** 0 warnings (both high and medium priority).
* **Checkstyle:** 12 warnings remain (mostly minor style issues, like long lines or missing Javadoc periods).

b) Number of Checkstyle and SpotBugs Violations/Bugs (Current vs. Initial StaticAnalysis Branch)

| **Tool** | **StaticAnalysis (Initial)** | **Current Branch** |
| --- | --- | --- |
| **SpotBugs** | 4 total (1 High, 3 Medium) | 0 |
| **Checkstyle** | 12 (10 in Game.java, 2 in Main.java) | 12 (same warnings remain) |

c) Comparison to StaticAnalysis Branch *After* Corrections

| **Tool** | **StaticAnalysis (After Fixes)** | **Current Branch** |
| --- | --- | --- |
| **SpotBugs** | 0 | 0 |
| **Checkstyle** | 12 | 12 |

The **SpotBugs issues were resolved**, indicating functional improvements. However, the **Checkstyle violations remained unchanged**, likely because they were related to style/documentation and not fixed.

d) Table Summary of Violations Across Branches

| **Branch** | **SpotBugs (High/Medium)** | **SpotBugs Total** | **Checkstyle Total** | **Notes** |
| --- | --- | --- | --- | --- |
| **StaticAnalysis (Before Fixes)** | 1 / 3 | 4 | 12 | Functional + Style issues present |
| **StaticAnalysis (After Fixes)** | 0 / 0 | 0 | 12 | Functional bugs fixed only |
| **Current Branch (Post-Fix)** | 0 / 0 | 0 | 12 | Same as after fixes |
| **Blackbox** | Not shown | Not shown | Not shown | Not evaluated here |
| **Review** | Not shown | Not shown | Not shown | Not evaluated here |

**e) Did it get better or worse? Can you explain why?**

* **Better** in terms of **SpotBugs** — all critical and medium issues have been resolved, indicating improvement in the logic and reliability of the code.
* **Same** in terms of **Checkstyle** — cosmetic and documentation warnings were not addressed, so this area hasn't improved.

**Why?**  
The improvements were likely targeted at **functional bugs (SpotBugs)** rather than **style (Checkstyle)**. The development effort was focused on correctness and performance rather than formatting and documentation compliance.