SE 6367: Software Testing

Project 3: Effective Software Fault Location

- **Part I (100 points): Execution dice-based Fault Localization**
  Use χSlice to highlight some suspicious code

  *** READ THIS BEFORE YOU CONTINUE ***

  Note that in this part, you will be using *x*Slice to automatically locate location(s) of the bug(s). However, since the subject program (i.e., gzip) is very large, *x*Slice can be slow, which may make you think it froze. (e.g., open xSlice with gzip source file opened may take about 1-2 minutes). Here are a few tips that can help you make *x*Slice faster:

  1. don't drag any scroll bar in the tool
  2. after you decide which test cases to be selected, and you are ready to use *x*Slice to locate the bug, make sure your trace file only contains the execution information of the test cases you want to select.
     (e.g., you have 100 test cases, you only want to use 4th, 25th, 60th, and 99th test cases for a run, delete your existing trace file if there is any, only run 4th, 25th, 60th, and 99th test cases, then launch the tool and proceed)
  3. don't randomly decide which test cases should be selected and launch the tool to see what statements are highlighted. This is not how this project should be done. Think carefully about which test cases should be selected and why they should be selected.

  *** PLEASE CONTINUE ***

  Test Case Execution:

  1) Go to the folder of project 3 under your home directory.

```
test2019fall@xsuds:~$ ls
Desktop  examples.desktop  LastOpenedFiles.xml  Project-1  Project-3  sqatool_workspace  xRegress-Tutorial
test2019fall@xsuds:~$
```

  2) Go to the folder that contains the correct version of gzip.

```
test2019fall@xsuds:~/Project-3$ ls
gzip-Correct-Version  gzip-Faulty-Version  inputs  inputs.alt  outputs  outputs-correct  Project-3-20191018  testplans  testplans.alt
test2019fall@xsuds:~/Project-3$
```

  3) Execute *run-all.sh* to execute the 100 test cases against the **correct** version of gzip.

```
test2019fall@xsuds:~/Project-3/gzip-Correct-Version$ ls
gzip  run-all.sh
test2019fall@xsuds:~/Project-3/gzip-Correct-Version$ ./run-all.sh
```

  4) Go to the folder that contains the faulty version of gzip.

```
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$ cd ..
test2019fall@xsuds:~/Project-3$ ls
gzip-Correct-Version  gzip-Faulty-Version  inputs  inputs.alt  outputs  outputs-correct  Project-3-20191018  testplans  testplans.alt
test2019fall@xsuds:~/Project-3$ cd gzip-Faulty-Version/
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$
```

  5) Execute *run-all.sh* to execute the 100 test cases against the **faulty** version of gzip.

```
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$ l
Coverages/  getopt.h*  gzip.AElock*  gzip.atacp*  gzip.h*  gzip.trace*  Makefile*  run-all.sh*
crypt.h*    gzip*      gzip.atac*    gzip.c*      gzip.o*  lzw.h*       revision.h*  tailor.h*
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$ ./run-all.sh
```

Output Verification:

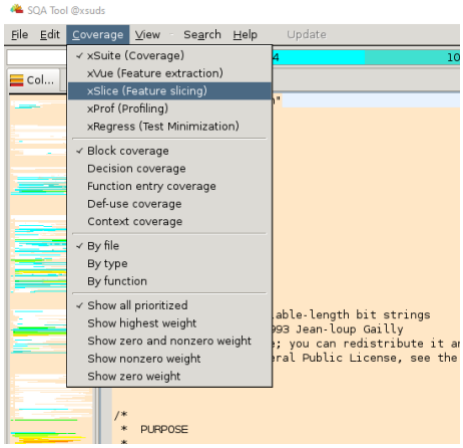
6) Use your own method to compare the outputs of faulty version to the outputs of correct version. Find out which test cases of faulty version were failed. A test case of the faulty version can be failed if its output is different with the correct one.

Use χSlice to Locate the Bug Location:

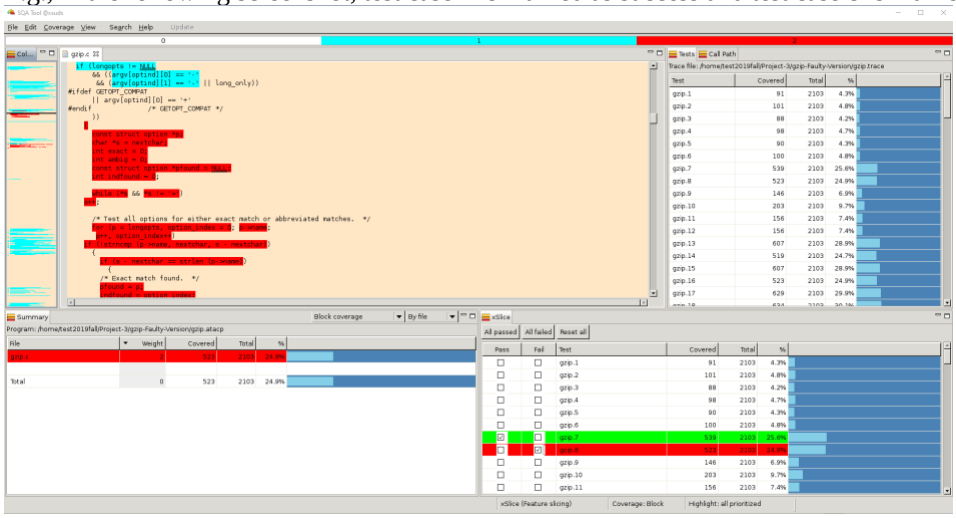
7) Launch χSlice, open project files, and load the trace file.

```
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$ ls
Coverages  crypt.h  getopt.h  gzip  gzip.AElock  gzip.atac  gzip.atacp  gzip.c  gzip.h  gzip.o  gzip.trace  lzw.h  Makefile  revision.h  run-all.sh  tailor.h
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$ xsuite *.atacp *.trace
Picked up _JAVA_OPTIONS: -Xms512m -Xmx1400m
```

8) Switch to χSlice. **From this point, χSlice can be slow. Please be patient.**



9) Use χSlice to help identify the suspicious code.
E.g., in the following screenshot, test case 7 is marked as success and test case 8 is marked as fail.



- **Part II Suspiciousness Ranking-based Fault Localization**
Design your own heuristics to identify suspicious code.

Project Submission

Part I: Submit a tar/zip file including the following two items.
Name your file to *Project-3A-FirstName-LastName.tar* (or zip)
Submit your file to eLearning

1) a table reporting the result of each test case

	Execution result (1: failed and 0: successful)
test-01	
test-02	
. . . . .	
test-100	

2) a table with failed and successful tests you select for each run and the most suspicious lines (the code highlighted in red) identified in each run

	Failed tests used	Successful tests used	Heuristics (via χSlice)	Suspicious code†
1st run	test-01, test-33, ...	test-10, test-21, ...	⋂ (execution slices of all failed tests) - ⋃ (execution slices of all successful tests)	Report line numbers of the code that is highlighted in red
2nd run			⋂ (execution slices of all failed tests) - ⋃ (execution slices of all successful tests)	
3rd run			⋂ (execution slices of all failed tests) - ⋃ (execution slices of all successful tests)	
4th run			⋂ (execution slices of all failed tests) - ⋃ (execution slices of all successful tests)	
5th run			⋂ (execution slices of all failed tests) - ⋃ (execution slices of all successful tests)	

Part II: Submit a tar/zip file including the following item
Name your file to *Project-3B-FirstName-LastName.tar* (or zip)
Submit your file to eLearning

3) a table with failed and successful tests and the formula used to identify suspicious code

	Failed tests used	Successful tests used	Your own formula to compute the suspiciousness	Suspicious code†
1st run	test-01, test-33, ...	test-10, test-21, ...	The formula (in terms of α, β, γ and η) used to compute the suspiciousness of each statement (e.g., suspiciousness = α – β)	Report the top 15 most suspicious lines of code (e.g., lines 15 to 20 and lines 35 to 45)
2nd run				
3rd run				
4th run				
5th run				

For a given statement *s*, parameters α, β, γ and η are the respective number of failed (or successful) tests that execute (or not execute) it.

	s is covered	*s* is not covered
Number of failed tests	α	γ
Number of successful tests	β	η

The coverage information of all test cases can be found in the "Coverage" folder under "gzip-Faulty-Version" folder.

```
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$ ls
Coverages  crypt.h  getopt.h  gzip  gzip.AElock  gzip.atac  gzip.atacp  gzip.c  gzip.h  gzip.o  gzip.trace  lzw.h  Makefile  revision.h  run-all.sh  tailor.h
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$ ls Coverages/
Coverage-vector-100.coverage  Coverage-vector-28.coverage  Coverage-vector-46.coverage  Coverage-vector-64.coverage  Coverage-vector-82.coverage
Coverage-vector-10.coverage   Coverage-vector-29.coverage  Coverage-vector-47.coverage  Coverage-vector-65.coverage  Coverage-vector-83.coverage
Coverage-vector-11.coverage   Coverage-vector-2.coverage   Coverage-vector-48.coverage  Coverage-vector-66.coverage  Coverage-vector-84.coverage
Coverage-vector-12.coverage   Coverage-vector-30.coverage  Coverage-vector-49.coverage  Coverage-vector-67.coverage  Coverage-vector-85.coverage
Coverage-vector-13.coverage   Coverage-vector-31.coverage  Coverage-vector-4.coverage   Coverage-vector-68.coverage  Coverage-vector-86.coverage
Coverage-vector-14.coverage   Coverage-vector-32.coverage  Coverage-vector-50.coverage  Coverage-vector-69.coverage  Coverage-vector-87.coverage
Coverage-vector-15.coverage   Coverage-vector-33.coverage  Coverage-vector-51.coverage  Coverage-vector-6.coverage   Coverage-vector-88.coverage
Coverage-vector-16.coverage   Coverage-vector-34.coverage  Coverage-vector-52.coverage  Coverage-vector-70.coverage  Coverage-vector-89.coverage
Coverage-vector-17.coverage   Coverage-vector-35.coverage  Coverage-vector-53.coverage  Coverage-vector-71.coverage  Coverage-vector-8.coverage
Coverage-vector-18.coverage   Coverage-vector-36.coverage  Coverage-vector-54.coverage  Coverage-vector-72.coverage  Coverage-vector-90.coverage
Coverage-vector-19.coverage   Coverage-vector-37.coverage  Coverage-vector-55.coverage  Coverage-vector-73.coverage  Coverage-vector-91.coverage
Coverage-vector-1.coverage    Coverage-vector-38.coverage  Coverage-vector-56.coverage  Coverage-vector-74.coverage  Coverage-vector-92.coverage
Coverage-vector-20.coverage   Coverage-vector-39.coverage  Coverage-vector-57.coverage  Coverage-vector-75.coverage  Coverage-vector-93.coverage
Coverage-vector-21.coverage   Coverage-vector-3.coverage   Coverage-vector-58.coverage  Coverage-vector-76.coverage  Coverage-vector-94.coverage
Coverage-vector-22.coverage   Coverage-vector-40.coverage  Coverage-vector-59.coverage  Coverage-vector-77.coverage  Coverage-vector-95.coverage
Coverage-vector-23.coverage   Coverage-vector-41.coverage  Coverage-vector-5.coverage   Coverage-vector-78.coverage  Coverage-vector-96.coverage
Coverage-vector-24.coverage   Coverage-vector-42.coverage  Coverage-vector-60.coverage  Coverage-vector-79.coverage  Coverage-vector-97.coverage
Coverage-vector-25.coverage   Coverage-vector-43.coverage  Coverage-vector-61.coverage  Coverage-vector-7.coverage   Coverage-vector-98.coverage
Coverage-vector-26.coverage   Coverage-vector-44.coverage  Coverage-vector-62.coverage  Coverage-vector-80.coverage  Coverage-vector-99.coverage
Coverage-vector-27.coverage   Coverage-vector-45.coverage  Coverage-vector-63.coverage  Coverage-vector-81.coverage  Coverage-vector-9.coverage
test2019fall@xsuds:~/Project-3/gzip-Faulty-Version$
```

- If you have any questions, please contact the TA.