SE/CS 6367 Software

Project 3: Effective Software Fault Location

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

- **Part II (400 points) Suspiciousness Ranking-based Fault Localization**
  Design your own heuristics to identify suspicious code

<u>README</u>

1) Create a working directory and download the files from
    http://www.utdallas.edu/~ewong/SE6367/01-Project/project-03-Data.rar

    Do not modify the following directories

    - testplans.alt   // contains test scripts
    - testplans   // used for test script execution
    - outputs   // contains saved execution outputs
    - inputs.alt   // contains test cases
    - inputs   // used for test case execution

    Test oracle is saved in the directory *gzip-Correct-Version*.
    Source code of the faulty version is in the directory *gzip-Faulty-Version*.

    Run this command to give the READ permission to files in your working directory:

    $ chmod -R 755 ./*name-of-your-working-directory*

2) Login to *cs4.utdallas.edu* via *Xmanager* and set the *experiment_root* to your working directory

    Example:

    Assuming your NetID is *XYZ* and the name of your working directory is *Project-3*, run this command:

    $ export experiment_root=/people/cs/g/gxr116020/Project-3

3) To compile the program
        $ cd gzip-Faulty-Version
        $ make

    To erase results from the previous compilation/run
        $ make clean

    To execute test cases
        $ ./run-all.sh ./gzip

Project Submission

Part I: Submit a tar file (*Project-3A-FirstName-LastName.tar*) containing

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 and the heuristics used to identify suspicious code

	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 the top 30 most suspicious lines of code
2nd run				
. . . . .				

[†]You need to submit a paper copy with suspicious code circled in red

Part II: Submit a tar file (*Project-3B-FirstName-LastName.tar*) containing

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

	Failed tests used	Successful tests used	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 = $\alpha - \beta$)	Report the top 30 most suspicious lines of code
2nd run				
. . . . .				

[†]You need to submit a paper copy with suspicious code circled in red

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