Fantastic-Four_deliverable5.md

Chapter 5: Fault Injection

Executive Summary:

We have completed the task of injecting faults into our source code. A one-symbol change to emulate a typo was made in the return statement for each of the following methods:

1. `add(x, y)` : "+" was replaced with "-", to simulate an adjacent key typo
2. `sub(x, y)` : "-" was replaced with "+", to simulate an adjacent key typo
3. `mul(x, y)` : an additional "*" was added, to simulate a double keystroke typo
4. `div(x, y)` : an additional "/" was added, to simulate a double keystroke typo
5. `pow(x, y)` : a "*" was removed, to simulate a missed keystroke typo

Small tweaks were made to a few test cases to ensure that not all tests will fail when the faults are injected.

Technical Summary:

Fault Injection Process

Faults were injected locally but not pushed to the repository so that a fresh clone of the repo will have clean source code. Directions for duplicating our faults can be found in the `faults.txt` file, which can be found in the directory `Fantastic-Four/TestAutomation/docs` .

An example report generated by our framework while faults are present, `faultReport.html` , can be found in the directory `Fantastic-Four/TestAutomation/reports` .

Fault Injection Results

Expected Results:

All previously passing test cases now fail, with the following exceptions, which are unaffected:

Test ID	Requirement	Component	Method	Test Inputs	Oracles
11	Multiplication of two whole numbers	`functions.py`	mul(x, y)	(2,1)	2
12	Multiplication of negative and positive integer	`functions.py`	mul(x, y)	(-1,1)	-1
21	Exponentiation of two whole numbers	`functions.py`	pow(x, y)	(2,1)	2
22	Exponentiation of negative and positive integer	`functions.py`	pow(x, y)	(-1,1)	-1

Unexpected Results:

The following test cases are failing, with or without fault injection, but should pass with or without faults:

Test ID	Requirement	Component	Method	Test Inputs	Oracles
16	Division of two whole numbers	`functions.py`	div(x, y)	(2,1)	2
17	Division of negative and positive integer	`functions.py`	div(x, y)	(-1,1)	-1.0