CS 362 Software Engineering Fall 2018
Student: Daniel Krueger
Oct 30, 2018
Random Testing Quiz

After looking at the code it was quickly determined that the subset of values that would create an error were in the ASCII subset of characters between ASCII 32 to ACI 126 (‘ ’ to ‘~’). Other characters were avoided because this is a console based program and non printing characters can create unusual behavior with the console display.

The First think I did after creating the random character generator was to make sure that it was generating the range of characters expected. I used an iterative loop of 10,000 iterations expecting that probalisticaly it was very low that every character would not have been hit by that time by the generator. I also wanted to make sure that the random character generator would not create values outside the set range {‘ ‘ to ‘~’ }.

Then I made sure the string “reset” would actually create an exit code of 200. But this was a non iterative test and just checking to make sure the condition was physically hittable.

Since it was impossible to tell the length of string that would be tested without looking into the code, the examination of the code provided that only 5 characters were utilized for the error code of ‘reset’. The permutations are 735 Billion with 95 different characters. Limiting the subset to {a-Z} would be 380 million permutations and just using lowercase letters {a-z} would be 11.8 million. It was decided that the lower subset would be run first and based on running time of that test it would be determined if the larger character populations would be tested.

My computer was running about 1M test cases / 30 sec. So without the subset whittled down, it would take approximately 3.2 hours to run all cases { a-Z } and 271 days to run the other subset {‘ ‘ – ‘~’}.

Results:

94.2 % coverage

Coverage was about as expected. I was surprised that the if (s[0] ==...) counted as separated lines. It demonstrates that with the && statements, the execution stops on first && failure. This could be exploited to break apart the criteria if we felt that a certain part of the ‘if block’ was at fault for some bug.