Team Member:
Zach Ryan
Adam Salyers
Luke Favret
Bao Tran
Kaiwen Chen

Random Number Generator

Test Scenarios:

1. Image Processing
    a. Input same image into OpenCV multiple times to make sure the same output is generated
        i. Checks for contour angle generation for consistent results
    b. Input different image in OpenCV to see if output is different
        i. Verify that angle output are not the same for each picture and visually verify the difference through observation
    c. Input multiple images from different frames of a video/gif to verify results are consistently different.
2. Checking random numbers:
    a. Generate 20 random strings of the same length to check for adequate differences
        i. Test for all types of strings that can be used(8-digits,10-digits,16-digits,128-digits)
    b. Generate an extremely long string i.e > 10000 characters to make sure there are no repeating sections
    c. Repeat generation process across visually similar images to ensure enough difference in randomness, even in similar test scenarios.
    d. Use random number generation with multiple pseudo-random generators to verify that the numbers generated work as seeds.
    e. Run a dieharder test on the generation to catch any possible patterns that are not immediately prevalent in the data.
3. Testing user login system
    a. Create account
        i. Username: PaulB Password:m4llN!NJ4
    b. Try to create account w/ username already taken
        i. i.e. PaulB, d0nutS#
    c. Login and refresh the page, then logout
    d. Login, change pages multiple times, logout
    e. Login, go to my profile, logout, go to my profile URL
    f. Enter incorrect login info and try to login
    g. Leave a field blank and try to login
        i. Repeat this for every field

4. User Random Number Storage
    a. Generate a large amount of single number sets and store them, then try to retrieve them
    b. Enter the URL of a stored set/ the stored set page of another user
        i. Ensure users can’t see other’s numbers
    c. Try to retrieve sets of a user with 0 sets
    d. Try to retrieve sets of a user with an obscene amount of sets
    e. Generate a large number of sets each with a large number of elements and store them, then try to retrieve them
    f. Generate enough sets/numbers to cause memory issues, then ensure software deals with it properly

Github repo:https://github.com/kaiwen31/3308-Project-Milestone