Natalie Wintour | Testing Oracles

Oracle 1:

My first oracle selection was to create random generators for square and rectangle that are inverse of the functions which categorize them. I chose this approach because squares and rectangles are the specific to identify. Any of the remaining shapes can be created from small transformation to either a square or rectangle.

I did however use the c standard library version of rand() rather than creating my own seed. This made the random numbers slightly less random as rand() is not truly random.

Once I could assert that I could transform a square to another square and a rectangle to another rectangle, I could transform inputs to make all additional shapes for testing.

Oracle 2:

My second oracle was the transformation to the same and other known shapes. This oracle still relies on the random number generation in my square and rectangle functions so to maintain some amount of randomization with a known transformation. Example, a rectangle can be transformed into a less specific category (such as a parallelogram) by moving two of the four points in unison. With a few simple transformations, I can create all the desired shapes. This created known outputs based on semi-random inputs and transformations.

Oracle 3:

My last oracle was the least random. A random string generator could generate errors with a string I knew would be mostly likely rejected in order to trigger a wide range of random input errors. I then choose to insert alphabetic characters and symbols to trigger additional errors.

In order to trigger all of the errors and continue, I saved the errors and printed to match expected input rather than exit.