Deque Test Suite Description

./DQtest:

NOTE: Elements that are dequeued or ejected will remain in the circular array until they are overwritten.

TEST-1: A basic display of Deque methods *(enqueue, dequeue, jump, eject, reserve, clear)*

- Invoking enqueue and jump methods on an empty Deque to demonstrate basic addition of elements
- Invoke dequeue and eject operations on the already filled Deque to demonstrate basic removal of elements
- Invoke the reserve method to display the expansion property of the array (assuming the requested size is larger than the current) and the clear method to reset

TEST-2: Demonstrating the "circularity" of the array with both front and back wrapping of elements when the indices exceeds the array boundaries

- Create a Deque where elements are in the "middle" of the array then proceed to enqueue elements pass the array's end bound to demonstrate the insert "wrapping" from the back to front
- Dequeue the current array to demonstrate the removal "unwrapping" from the front to back
- The steps are above are repeated but with inverse operations to demonstrate "wrapping" from front to back and "unwrapping" of elements from the back to front

TEST-3: Show the expanding property of the Deque when its size reaches its capacity and the rearrangement of elements after expansion

- Demonstrate the auto-expanding property of the Deque by: enqueueing a full Deque, jumping a full Deque (while array is in a wrapped or unwrapped state)
- Invoke enqueue/jump operations when the Deque is full (AND the array is in its "wrapped" state) to demonstrate the reorganization of elements after expansion (order of elements become non-circular again)
- NOTE: there will likely be a lot garbage values from the previous Deque prior to clearing

TEST-4: Demonstrating the use of the Deque with different data types *(char and string)* as a template class

- Perform a the basic operations (enqueue, dequeue, jump, eject) on the char and string to demonstrate the Deque as a templated class

./DQarraytest:

NOTE: Tests with the rand() function will give different results every time the test program is ran

TEST-1: A demonstration of the retrieval of values in the Deque using the array bracket notation [i] with different indices

- Generate random integers to using the rand() function to insert into the Deque
- Generate random indices to access and retrieve from the Deque using [] notation

TEST-2: A demonstration of overwriting values in the Deque using the array bracket notation [i] and assignment operator with different indices

- Generate random integers to using the rand() function to insert into the Deque
- Generate random indices to overwrite values at those indices in the Deque
- The overwritten values are also randomly generated

TEST-3: A demonstration of retrieving and overwriting values in a Deque with different data types *(char, string)*

- Perform the same actions as above with retrieving and overwriting at random indices with random chars.