CSCI 3901 Winter 2021

Lab 4: Exceptions & Assertions

Name: Dhrumil Rakesh Shah

- **No Team Member (Doing Solo)**

Questions:

- We usually want you to re-use existing code and infrastructure whenever possible. Why might you create your own exception?
    - **Custom exceptions** provide you the flexibility to add attributes and methods that are not part of a standard Java **exception**. These can store additional information, like an application-specific error code, or provide utility methods that can be used to handle or present the **exception** to a user.

- We added parameters to the Fibonacci method. However, those parameters aren't very meaningful to a general user. What would you do to the code to make it more accessible for a general user?
    - The current implementation of the code having fiboRecursion() method takes 3 parameters as input from the user, so in order to make it more meaningful for the end user;
        - I would make a new method named fibo() that would take only take either 1 parameter (*just the number whose Fibonacci needs to be found*) or 2 parameters (*the number whose Fibonacci needs to be found & depth of recursion*) from the user.
        - This fibo() method will then call the fiboRecursion() method that takes 3 parameters as and return the desired result.
        - This way the user wouldn't have to bother about unnecessary input parameters.

- How would you recommend for someone to develop a loop invariant?
    - A good loop invariant should satisfy three properties:
        - **Initialization**: The loop invariant must be true before the first execution of the loop.
        - **Maintenance**: If the invariant is true before an iteration of the loop, it should be true also after the iteration.
        - **Termination**: When the loop is terminated the invariant should tell us something useful, something that helps us understand the algorithm.
    - A well-chosen loop invariant is useful both when designing, testing, and modifying code. It also serves as documentation and can be the foundation of a correctness proof.

- How can loop invariants help you in programming, even if you don't include them directly as assertions in your code?
    - Loop invariants help by checking the looping conditions and the value that variables hold as it is kind of a sanity check for our programs to function properly and avoid errors.
    - It should be noted that a Loop Invariant can help in the design of iterative algorithms when considered an assertion that expresses important relationships among the variables that must be true at the start of every iteration and when the loop terminates. If this holds, the computation is on the road to effectiveness.