SV_CodeTest

Overview of the solution

The solution is developed as a C# .net framework console application which can be run as executable from the following path: /SV_CodeTest/bin/Debug/SV_CodeTest.exe

The application consumes a static method `Game(int n,int k)` which performs the circular iteration on a collection of child elements with a skip factor K as defined in the game rules. The **Iteration behavioural design pattern** is used to develop the solution with an iterator that works on the aggregator. The rules of the iterator are defined in such a way that it moves through the collection in a clockwise circular fashion. The children who goes out of the game are removed from the collection so that the game will resume from the child immediately after the child who goes out. These elements are collected in a different collection in the order they are going out of the game. This helps to maintain track of the sequence.

The Iterator is defined to stop (IsDone) when the collection becomes reduced to only one participant child element in the game. The last child element in the collection is defined as the winner and added to the results object along with the list of children who are out of the game.

The result object holds the following information for the client that consumes the static method:

1. Success - Set false if the game cannot be started with the given parameters
2. Error Message - Error messages if any
3. Winner - the winner child's ID
4. Out List - collection of children who are out of the game in sequence

Assumptions

If the k factor of the game exceeds the number of participating children(n), the game will continue until the child who counts the number k will go out of the game.

Test Cases

The solution contains visual studio unit test project which assets the following test cases.

1. The game cannot start if the number of participating children is 0.
2. The game cannot start if the k factor of the game if 0.
3. The number of children who are out of the game is n-1
4. There will be one and only one winner
5. Catch out of range exceptions

The solution was developed in a Test Driven Development **(TDD)** methodology were the code was developed to fail and the code was refactored until all test cases are passed (green).