Swinburne University of Technology

Faculty of Science, Engineering and Technology

ASSIGNMENT COVER SHEET

Subject Code: COS30008
Subject Title: Data Structures & Patterns
Assignment number and title: 2 - Iterators
Due date: Monday, 22 April, 2024, 10:30
Lecturer: Dr. Markus Lumpe

Your name: Ruffin Remad **Your student id:** 103840173

Marker's comments:

Problem	Marks	Obtained
1	40	
2	70	
Total	110	

Extension certification:

This assignment has been given an extension and is now due on _____

Signature of Convener:_____

```cpp
#include "FibonacciSequenceGenerator.h"

// Constructor to set up a Fibonacci sequence
FibonacciSequenceGenerator::FibonacciSequenceGenerator(const std::string& aID) noexcept
    : fID(aID), fPrevious(0), fCurrent(1) {}

// Get sequence ID
const std::string& FibonacciSequenceGenerator::id() const noexcept {
    return fID;
}

// Get current Fibonacci number
const long long& FibonacciSequenceGenerator::operator*() const noexcept {
    return fCurrent;
}

// Type conversion to bool
// Returns true if there is a next Fibonacci number to be generated
FibonacciSequenceGenerator::operator bool() const noexcept {
    return hasNext();
}

// Reset sequence generator to first Fibonacci number
void FibonacciSequenceGenerator::reset() noexcept {
    fPrevious = 0;
    fCurrent = 1;
}

bool FibonacciSequenceGenerator::hasNext() const noexcept {
    return fCurrent <= (LLONG_MAX - fPrevious);
}

// Advance to next Fibonacci number
void FibonacciSequenceGenerator::next() noexcept {

    long long lNext = fCurrent + fPrevious;
    fPrevious = fCurrent;
    fCurrent = lNext;
}
```

```cpp
#include "FibonacciSequenceIterator.h"

FibonacciSequenceIterator::FibonacciSequenceIterator(const FibonacciSequenceGenerator& aSequenceObject,
    long long aStart) noexcept :
    fSequenceObject(aSequenceObject), fIndex(aStart)
{}

// iterator methods
const long long& FibonacciSequenceIterator::operator*() const noexcept
{
    return *fSequenceObject;
}

FibonacciSequenceIterator& FibonacciSequenceIterator::operator++() noexcept
{
    fSequenceObject.next();
    fIndex++;
    return *this;
}

FibonacciSequenceIterator FibonacciSequenceIterator::operator++(int) noexcept
{
    FibonacciSequenceIterator lOld = *this;
    ++(*this);
    return lOld;
}

bool FibonacciSequenceIterator::operator==(const FibonacciSequenceIterator& aOther) const noexcept
{
    return fIndex == aOther.fIndex;
}

bool FibonacciSequenceIterator::operator!=(const FibonacciSequenceIterator& aOther) const noexcept
{
    return !(*this == aOther);
}

// iterator auxiliary methods

// return new iterator positioned at start
FibonacciSequenceIterator FibonacciSequenceIterator::begin() const noexcept
{
    FibonacciSequenceIterator lTemp = *this;
    lTemp.fIndex = 1;
    return lTemp;
}

```

```
48  // return new iterator positioned at limit
49  FibonacciSequenceIterator FibonacciSequenceIterator::end() const
      noexcept
50  {
51      FibonacciSequenceIterator lTemp = *this;
52      lTemp.fIndex = 93;
53      return lTemp;
54  }
```