```
//
//  FibonacciSquenceIterator.cpp
//  Asignment2
//
//  Created by Luan Nguyen on 17/4/2024.
//

#include <stdio.h>
#include "FibonacciSequenceIterator.h"

#include <cassert>

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



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

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

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



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

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

FibonacciSequenceIterator FibonacciSequenceIterator::begin() const noexcept
{

    FibonacciSequenceGenerator lSequence =
    FibonacciSequenceGenerator(fSequenceObject.id());

    return FibonacciSequenceIterator(lSequence);
}

FibonacciSequenceIterator FibonacciSequenceIterator::end() const noexcept
{
    FibonacciSequenceGenerator lSequence =
     FibonacciSequenceGenerator(fSequenceObject.id());
    long long lIndex = 1;
    while (lSequence.hasNext() == true)
    {
        lSequence.next();
        ++lIndex;
    }
    return FibonacciSequenceIterator(lSequence,lIndex+1);
}
```