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: Avery Flannery **Your student id:** 104416957

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:_____

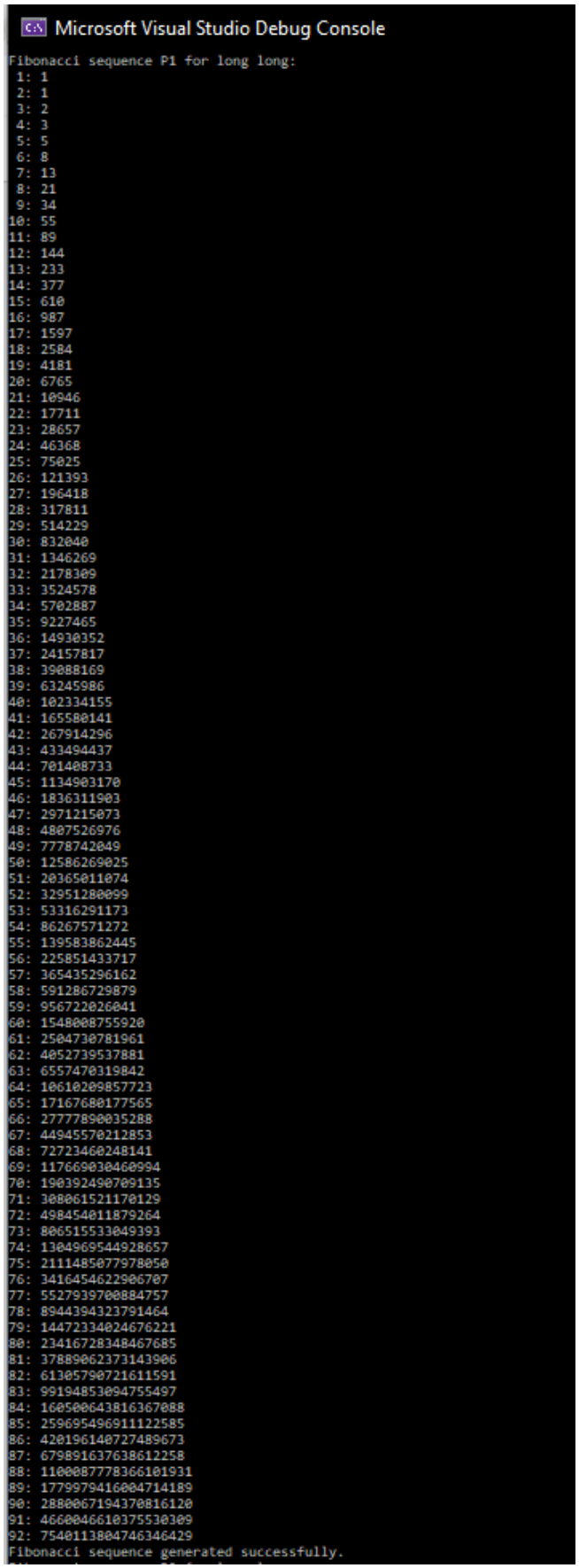
Problem 1 – FibonacciSequenceGenerator.cpp

```cpp
//COS30008 - 104416957 - Avery Flannery. Semester 1 2024
//Fibonacci Sequence Generator


#include "FibonacciSequenceGenerator.h"

#include <cassert>

// constructor
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;
}

//deference operator to get current Fibonacci number
const long long& FibonacciSequenceGenerator::operator*() const noexcept {
    return fCurrent; // returning the current number
}

// converting operator to bool
FibonacciSequenceGenerator::operator bool() const noexcept {
    return hasNext(); // check if there is a next Fibonacci number
}

// used to reset the sequence generator
void FibonacciSequenceGenerator::reset() noexcept {
    fPrevious = 0; // return previous
    fCurrent = 1; // return current
}

// used to check if there is a next Fibonacci number
bool FibonacciSequenceGenerator::hasNext() const noexcept {
    long long next = fPrevious + fCurrent; // calculates the next number
    return next >= 0; // checks if the number is negative
}

// used to go on to the next Fibonacci number in the sequence
void FibonacciSequenceGenerator::next() noexcept {
    long long next = fPrevious + fCurrent; // calculates the number
    fPrevious = fCurrent; // updates the previous Fibonacci number
    fCurrent = next; // updates the current Fibonacci number
}
```

Problem 2 – FibonacciSequenceIterator.cpp

```cpp
//COS30008 - 104416957 - Avery Flannery. Semester 1 2024
//Fibonacci Sequence Iterator

#include "FibonacciSequenceIterator.h"

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


// deference operator
const long long& FibonacciSequenceIterator::operator*() const noexcept {
    return *fSequenceObject; // deferring the object
}

// prefix increment operator
FibonacciSequenceIterator& FibonacciSequenceIterator::operator++() noexcept {
    ++fIndex; // increments iterator position
    fSequenceObject.next(); //move to the next Fibonacci number
    return *this;
}

// postfix increment operator
FibonacciSequenceIterator FibonacciSequenceIterator::operator++(int) noexcept {
    FibonacciSequenceIterator temp = *this; // create a copy of the current
iterator
    ++(*this); // increment the iterator position
    return temp; // return the copy
}

// == equals comparison operator
bool FibonacciSequenceIterator::operator==(const FibonacciSequenceIterator&
aOther) const noexcept {
    // comparing the sequence objects' ID and iterator positions
    return (fSequenceObject.id() == aOther.fSequenceObject.id()) && (fIndex ==
aOther.fIndex);
}

// != does not equal comparison operator
bool FibonacciSequenceIterator::operator!=(const FibonacciSequenceIterator&
aOther) const noexcept {
    return !(*this == aOther);
}

FibonacciSequenceIterator FibonacciSequenceIterator::begin() const noexcept {
    FibonacciSequenceIterator beginIterator = *this;
    beginIterator.fIndex = 1; // sets the iterator to 1 to start
    beginIterator.fSequenceObject.reset();
    return beginIterator; // returns the iterator
}

FibonacciSequenceIterator FibonacciSequenceIterator::end() const noexcept {
    FibonacciSequenceIterator endIterator = *this;
    endIterator.fIndex = 93; // sets the iterator position to 93 at the end
    return endIterator; // returns the iterator
}
```

Output


Microsoft Visual Studio Debug Console
Fibonacci sequence P1 for long long:
1: 1
2: 1
3: 2
4: 3
5: 5
6: 8
7: 13
8: 21
9: 34
10: 55
11: 89
12: 144
13: 233
14: 377
15: 610
16: 987
17: 1597
18: 2584
19: 4181
20: 6765
21: 10946
22: 17711
23: 28657
24: 46368
25: 75025
26: 121393
27: 196418
28: 317811
29: 514229
30: 832040
31: 1346269
32: 2178309
33: 3524578
34: 5702887
35: 9227465
36: 14930352
37: 24157817
38: 39088169
39: 63245986
40: 102334155
41: 165580141
42: 267914296
43: 433494437
44: 701408733
45: 1134903170
46: 1836311903
47: 2971215073
48: 4807526976
49: 7778742049
50: 12586269025
51: 20365011074
52: 32951280099
53: 53316291173
54: 86267571272
55: 139583862445
56: 225851433717
57: 365435296162
58: 591286729879
59: 956722026041
60: 1548008755920
61: 2504730781961
62: 4052739537881
63: 6557470319842
64: 10610209857723
65: 17167680177565
66: 27777890035288
67: 44945570212853
68: 72723460248141
69: 117669030460994
70: 190392490709135
71: 308061521170129
72: 498454011879264
73: 806515533049393
74: 1304969544928657
75: 2111485077978050
76: 3416454622906707
77: 5527939700884757
78: 8944394323791464
79: 14472334024676221
80: 23416728348467685
81: 37889062373143906
82: 61305790721611591
83: 99194853094755497
84: 160500643816367088
85: 259695496911122585
86: 420196140727489673
87: 679891637638612258
88: 1100087778366101931
89: 1779979416004714189
90: 2880067194370816120
91: 4660046610375530309
92: 7540113804746346429
Fibonacci sequence generated successfully.

```
Fibonacci sequence P2 for long long:
 1: 1
 2: 1
 3: 2
 4: 3
 5: 5
 6: 8
 7: 13
 8: 21
 9: 34
10: 55
11: 89
12: 144
13: 233
14: 377
15: 610
16: 987
17: 1597
18: 2584
19: 4181
20: 6765
21: 10946
22: 17711
23: 28657
24: 46368
25: 75025
26: 121393
27: 196418
28: 317811
29: 514229
30: 832040
31: 1346269
32: 2178309
33: 3524578
34: 5702887
35: 9227465
36: 14930352
37: 24157817
38: 39088169
39: 63245986
40: 102334155
41: 165580141
42: 267914296
43: 433494437
44: 701408733
45: 1134903170
46: 1836311903
47: 2971215073
48: 4807526976
49: 7778742049
50: 12586269025
51: 20365011074
52: 32951280099
53: 53316291173
54: 86267571272
55: 139583862445
56: 225851433717
57: 365435296162
58: 591286729879
59: 956722026041
60: 1548008755920
61: 2504730781961
62: 4052739537881
63: 6557470319842
64: 10610209857723
65: 17167680177565
66: 27777890035288
67: 44945570212853
68: 72723460248141
69: 117669030460994
70: 190392490709135
71: 308061521170129
72: 498454011879264
73: 806515533049393
74: 1304969544928657
75: 2111485077978050
76: 3416454622906707
77: 5527939700884757
78: 8944394323791464
79: 14472334024676221
80: 23416728348467685
81: 37889062373143906
82: 61305790721611591
83: 99194853094755497
84: 160500643816367088
85: 259695496911122585
86: 420196140727489673
87: 679891637638612258
88: 1100087778366101931
89: 1779979416004714189
90: 2880067194370816120
91: 4660046610375530309
92: 7540113804746346429
Fibonacci sequence generated successfully.
2 test(s) run.
```