```cpp
#include "FibonacciSequenceGenerator.h"
#include <limits>
#include <stdexcept>

// Constructor to set up a Fibonacci sequence
//sequence ID + prev 0 + cur 1 as required
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
FibonacciSequenceGenerator::operator bool() const noexcept
{
    //if current positive + next number can be calculated (same as hasNext())
    return fCurrent > 0 && std::numeric_limits<long long>::max() - fPrevious >= fCurrent;
}

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

//Tests if there is a next Fibonacci number
bool FibonacciSequenceGenerator::hasNext() const noexcept
{
    return std::numeric_limits<long long>::max() - fPrevious >= fCurrent;
}

//next Fibonacci number
void FibonacciSequenceGenerator::next() noexcept
{
    long long next = fCurrent + fPrevious;
    if (next < fCurrent) {
        //overflow assertion check
        throw std::overflow_error("Overflow in Fibonacci number calculation");
    }
    fPrevious = fCurrent;
    fCurrent = next;
}

```