David Lane

004638553

Project 5

1. This project I didn’t really have many obstacles to overcome, the first one was the bigger one which was forgetting the scope operator before the function definitions for the constructors for ZodiacReader class. This was causing Visual studio to not recognize Date as a variable within the header file for ZodiacReader. Weird. I looked over every line and eventually figured out though. I was also having some trouble writing driver code properly but I just looked at Howard’s for reference so that wasn’t that bad.
2. I actually think that Howard’s test code sufficiently tests the code in this project for my liking. He comprehensively tests the cuspsigns and stuff. If I were to really be extra careful it would essentially just be expanding his code to check for all the signs and cusps but given how I wrote the functions that would really just be to check my code for typos in the if checks for the dates.

I’m going to include Howard’s code here with my own notes and explanations

#include <iostream>

#include <string>

#include <cassert>

#include "Date.h"

#include "ZodiacReader.h"

using namespace std;

int main()

{

    // test code

Date jan1( 1, 1 ); checks constructor for Date

ZodiacReader reader( jan1 ); checks constructor for ZodiacReader

assert( reader.stringifySign( reader.checkSign() ) == "Capricorn"

); Tests the checksign function and the stringifysign function

assert( reader.checkSign( ) == ZodiacReader::CAPRICORN );

assert( !reader.onCusp( ) ); makes sure on cusp is returning false when not on cusp

Date jan20( 1, 20 );

ZodiacReader reader1( jan20 );

assert( reader1.stringifySign( reader1.checkSign() ) == "Aquarius" );

assert( reader1.checkSign( ) == ZodiacReader::AQUARIUS );

assert( reader1.onCusp( ) ); makes sure oncusp is working

assert( reader1.stringifySign( reader1.cuspSign() ) == "Capricorn" );

assert( reader1.cuspSign( ) == ZodiacReader::CAPRICORN );

Date jan21( 1, 21 );

ZodiacReader reader2( jan21 );

assert( reader2.stringifySign( reader2.checkSign() ) == "Aquarius" );

assert( reader2.checkSign( ) == ZodiacReader::AQUARIUS );

assert( reader2.onCusp( ) );

assert( reader2.stringifySign( reader2.cuspSign() ) == "Capricorn" );

assert( reader2.cuspSign( ) == ZodiacReader::CAPRICORN );

Date jan19( 1, 19 );

ZodiacReader reader3( jan19 );

assert( reader3.stringifySign( reader3.checkSign() ) == "Capricorn" );

assert( reader3.checkSign( ) == ZodiacReader::CAPRICORN);

assert( reader3.onCusp( ) );

assert( reader3.stringifySign( reader3.cuspSign() ) == "Aquarius" );

assert( reader3.cuspSign( ) == ZodiacReader::AQUARIUS);

Date feb18( 2, 18 );

ZodiacReader reader4( feb18);

assert( reader4.stringifySign( reader4.checkSign() ) == "Aquarius" );

assert( reader4.checkSign( ) == ZodiacReader::AQUARIUS );

assert( reader4.onCusp( ) );

assert( reader4.stringifySign( reader4.cuspSign() ) == "Pisces" );

assert( reader4.cuspSign( ) == ZodiacReader::PISCES );

Date feb17( 2, 17 );

ZodiacReader reader5( feb17 );

assert( reader5.stringifySign( reader5.checkSign() ) == "Aquarius" );

assert( reader5.checkSign( ) == ZodiacReader::AQUARIUS );

assert( reader5.onCusp( ) );

assert( reader5.stringifySign( reader5.cuspSign() ) == "Pisces" );

assert( reader5.cuspSign( ) == ZodiacReader::PISCES );

Date feb19( 2, 19 );

ZodiacReader reader6( feb19 );

assert( reader6.stringifySign( reader6.checkSign() ) == "Pisces" );

assert( reader6.checkSign( ) == ZodiacReader::PISCES);

assert( reader6.onCusp( ) );

assert( reader6.stringifySign( reader6.cuspSign() ) == "Aquarius" );

assert( reader6.cuspSign( ) == ZodiacReader::AQUARIUS);

             cout << "all tests passed!" << endl ;

    return 0;

}