Pair Programming 7 Turn In

Name: Matthew Krahel\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Username: c1020a12\_\_\_\_\_\_\_\_\_\_

Partner name: Hiram Kamau\_\_\_\_\_\_\_\_\_\_\_\_ Partner username: c1020a11\_\_\_\_\_

\_X\_ I certify that my partner worked with me on this assignment.

SCORE: \_\_\_\_\_\_\_\_\_\_\_\_ (to be filled in by instructor)

7a (5 points)

Graphical user interface, text

Description automatically generated

7b (5 points)

Graphical user interface, text, application

Description automatically generated

**/\* date.cpp**

**\* Author: Matthew Krahel and Hiram Kamau**

**\* Description: contains constants for dates**

**\*/**

**#include <iostream>**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <ctime>**

**#include <iomanip>**

**#include "date.h"**

**const int Date::MIN\_MONTH = JANUARY;**

**const int Date::MAX\_MONTH = DECEMBER;**

**const int Date::MIN\_YEAR = 1;**

**const int Date::MIN\_DAY = 1;**

**/\* Date: default constructor sets date to today**

**\*/**

**Date::Date()**

**{**

**time\_t t = time(NULL);**

**tm\* tPtr = localtime( &t );**

**month = tPtr->tm\_mon;**

**day = tPtr->tm\_mday;**

**year = tPtr->tm\_year + 1900;**

**}**

**/\* 3-arg constructor**

**\* Parameters:**

**\* mm month, 1 - 12**

**\* dd day in month**

**\* yyyy year**

**\*/**

**Date::Date( int mm, int dd, int yyyy ) {**

**setMonth( mm );**

**setYear( yyyy );**

**setDay( dd );**

**}**

**/\* 2-arg constructor**

**\* Parameters:**

**\* mm month, 1 - 12**

**\* yyyy year**

**\*/**

**Date::Date( int mm, int yyyy ) { // month, year parameters**

**setMonth( mm );**

**setYear( yyyy );**

**setDay( MIN\_DAY );**

**}**

**/\* 1-arg constructor**

**\* Parameters:**

**\* yyyy year**

**\*/**

**Date::Date( int yyyy ) {// year parameter**

**setMonth( MIN\_MONTH);**

**setYear( yyyy );**

**setDay( MIN\_DAY );**

**}**

**Date::~Date() {**

**printDate();**

**cout << " destroyed\n";**

**}**

**Date operator+( const Date& d1, const Date& d2 ) {**

**int month = d1.month, day = d1.day, year = d1.year;**

**// The code here just adds the days. It doesn’t**

**// check to see if it’s a valid day. This is fine**

**// for the purpose of the pair programming.**

**day += d2.day;**

**return Date( month, day, year );**

**}**

**bool operator==( const Date& d1, const Date& d2 ) {**

**if ( d1.day == d2.day && d1.month == d2.month &&**

**d1.year == d2.year ) return true;**

**else return false;**

**}**

**Date Date::operator-( ) const {**

**return Date( abs(this->month-6), this->day, this->year );**

**}**

**ostream& operator<<( ostream& outStream, const Date& d ) {**

**outStream << setfill( '0' ) << setw(2) <<**

**(d.month+1) << '/' << d.day << '/' << setw(4) << d.year;**

**return outStream;**

**}**

**istream& operator>>( istream& inStream, Date& d ) {**

**inStream >> d.month >> d.day >> d.year;**

**d.setMonth( d.month-1 ); // since user enters 1 for JAN**

**return inStream;**

**}**

**Date& Date::operator=( const Date& d1 ) {**

**this->day = d1.day;**

**this->month = d1.month;**

**this->year = d1.year;**

**return \*this;**

**}**

**/\* daysInMonth: returns number of days in month based upon year**

**\* Pre-condition object month and year have been set**

**\* Returns maximum valid day given a particular month/year**

**\*/**

**int Date::daysInMonth() const {**

**int max; // max number of days in month**

**switch( month ) {**

**case JANUARY: case MARCH: case MAY: case JULY:**

**case AUGUST: case OCTOBER: case DECEMBER:**

**max = 31; break;**

**case APRIL: case JUNE: case SEPTEMBER: case NOVEMBER:**

**max = 30; break;**

**case FEBRUARY: // FEB**

**// leap year?**

**if ( (year % 4 == 0 && year % 100 != 0) ||**

**( year %400 == 0) )**

**max = 29;**

**else**

**max = 28;**

**}**

**return max;**

**}**

**/\* printDate: prints date in form mm/dd/yyyy**

**\*/**

**void Date::printDate() const {**

**cout << setfill( '0' ) << setw(2) <<**

**(month+1) << '/' << day << '/' << setw(4) << year;**

**}**

**/\* setYear: sets year**

**\* Parameter y year**

**\* Returns: nothing**

**\*/**

**void Date::setYear ( int y ){**

**if ( y < MIN\_YEAR ) year = MIN\_YEAR;**

**else year = y;**

**}**

**/\* setMonth: sets month**

**\* Parameter m month**

**\* Returns: nothing**

**\*/**

**void Date::setMonth( int m){**

**if ( m < MIN\_MONTH ) month = MIN\_MONTH;**

**else month = m;**

**}**

**/\* setDay: sets day**

**\* Parameter d day**

**\* Returns: nothing**

**\*/**

**void Date::setDay( int d){**

**if ( d < MIN\_DAY ) day = MIN\_DAY;**

**else day = d;**

**}**

**/\* setDate: calls setMonth, setYear, and setDay functions**

**\* to then set the entire date at once**

**\* Parameter m month, d day, y year**

**\* Returns: nothing**

**\*/**

**void Date::setDate( int m, int d, int y ){**

**setMonth(m);**

**setYear(y);**

**setDay(d);**

**}**

**/\* date.h**

**\* Author: Matthew Krahel and Hiram Kamau**

**\* Description: header for date class**

**\*/**

**#ifndef DATE\_H**

**#define DATE\_H**

**#include <iostream>**

**using namespace std;**

**enum month\_t { JANUARY, FEBRUARY, MARCH, APRIL, MAY, JUNE,**

**JULY, AUGUST, SEPTEMBER, OCTOBER, NOVEMBER, DECEMBER };**

**class Date**

**{**

**public:**

**// default constructor is the constructor**

**// with no parameters**

**Date();**

**Date( int, int, int ); // month, day, year**

**Date( int, int ); // month, year parameters**

**Date( int ); // year parameter**

**virtual ~Date();**

**friend Date operator+( const Date& d1, const Date& d2 );**

**friend bool operator==( const Date& d1, const Date& d2 );**

**Date operator-( ) const;**

**friend ostream& operator<<( ostream& outStream, const Date& d );**

**friend istream& operator>>( istream& inStream, Date& d );**

**Date& operator=( const Date& d1 );**

**static const int MIN\_MONTH;**

**static const int MAX\_MONTH;**

**static const int MIN\_YEAR;**

**static const int MIN\_DAY;**

**void setYear( int );**

**int getYear() const { return year; }**

**void setMonth( int );**

**int getMonth() const { return month; }**

**void setDay( int );**

**int getDay() const { return day; }**

**void setDate( int, int, int );**

**void printDate() const ;**

**private:**

**int daysInMonth() const;**

**int month;**

**int day;**

**int year;**

**};**

**#endif**

**/\* File: main.cpp**

**\* Author: Cindy Arnold**

**\* This program tests a Date class**

**\*/**

**#include <iostream>**

**#include "date.h"**

**using namespace std;**

**int main(int argc, char \*\*argv)**

**{**

**// Code to test PP 7a start here**

**Date d1;**

**cout << "Minimum year: " << Date::MIN\_YEAR << endl;**

**//d1.month = -5;**

**// test setYear, getYear**

**d1.setYear(1999);**

**cout << "d1 year: " << d1.getYear() << endl; // 1999**

**d1.setYear(-5); // shoulld set to min year = 0**

**cout << "d1 year: " << d1.getYear() << endl; // 0**

**// test printDate and other set/get functions**

**d1.setMonth(DECEMBER);**

**d1.setDay(31);**

**cout << "d1: "; d1.printDate(); cout << endl; // 12/31/000**

**d1.setDate(JANUARY, 1, 2000);**

**cout << "d1: "; d1.printDate(); cout << endl; // 1/1/2000**

**d1.setMonth(FEBRUARY);**

**d1.setDay( 29 ); // test not a leap year**

**cout << "d1: "; d1.printDate(); cout << endl;**

**d1.setDate(-1, -1, -1); // test invalid everything**

**cout << "d1: "; d1.printDate(); cout << endl;**

**// test constructors**

**Date d2; // calls default constructor =**

**d2 = Date( ); // also calls default constructor**

**cout << "d2: "; d2.printDate(); cout << endl;**

**Date \*d2Ptr = new Date(); // default constructor**

**cout << "d2Ptr->: "; d2Ptr->printDate(); cout <<endl; // note -> operator**

**Date d3( 1999 ); // calls 1-arg conestructor**

**cout << "d3: "; d3.printDate(); cout << endl;**

**Date d4( APRIL, 2001 ); // calls 2-arg constructor**

**cout << "d4: "; d4.printDate(); cout << endl;**

**Date d5( MAY, 1, 2000); // calls 3-arg constuctor**

**cout << "d5: "; d5.printDate(); cout << endl;**

**Date d6( d5 ); // calls copy constructor**

**cout << "d6: "; d6.printDate(); cout << endl;**

**//End of code to test 7a**

**//The code below is for PP 7b**

**// test destructor**

**// don't forget to de-allocate memory when done with it! Otherwise program has a memory leak!**

**delete d2Ptr; // this calls destructor**

**// test overloading operators" +, =**

**d6 = d5 + d4; // tests + and = actually**

**cout << "d6 = d5 + d4: ";**

**d6.printDate(); cout << endl;**

**// test ==**

**d4.setDate( d5.getMonth(), d5.getDay(), d5.getYear());**

**if ( d5 == d4 ) { // should be true**

**cout << "d5 and d4 contain the same date\n";**

**}**

**// test - (negate)**

**d5 = -d4; // tests - and = actually**

**cout << "d5 = -d4: "; d5.printDate(); cout << endl;**

**// test >> and <<**

**cout << "Enter month, day, year: ";**

**cin >> d5;**

**cout << "You entered: " << d5 << endl;**

**// test an array of Date objects**

**Date dates[3];**

**dates[1].setYear( 1999 );**

**for( int i = 0; i < 3; i++ ) {**

**cout << dates[i] << endl;**

**}**

**return 0;**

**}**