///////////////

//Kelsey Segren

//EX03\_01

//MyPoint class 9.4

//////////////

/\*UML Diagram:

Class: MyPoint

Data Fields: double x, double y

constructors and functions: MyPoint()

MyPoint(double point1, double point2) double getx()

double gety()

double getx()

double distance (MyPoint p)

\*/

#include <iostream>

#include "MyPoint.h"

int main() {

//Test program with two points (0,0) and (10,30.5)

MyPoint p1;

MyPoint p2(10, 30.5);

double d = p1.distance(p2);

cout << d << endl;

return 0;

}

using namespace std;

#include <cmath>

#ifndef MYPOINT\_H

#define MYPOINT\_H

class MyPoint

{

private:

//Two-data fields x and y that represent the coordinates

double x;

double y;

public:

//No-Arg Constructor

MyPoint() {

x = 0;

y = 0;

}

//A constructor that constructs a point with specified coordinates

MyPoint(double point1, double point2) {

x = point1;

y = point2;

}

//Two get functions for data fields x and y

double getx() {

return x;

}

double gety() {

return y;

}

/\*A function named distance that returns the distance from this point to another point of

the MyPoint type \*/

double distance(MyPoint p) {

return sqrt((p.getx()-x)\*(p.getx() - x) + (p.gety() - y)\*(p.gety() - y));

}

};

#endif

/////////////

//Kelsey Segren

//EX03\_02

//LinearEquation

///////////////

/\*UML Diagram

Class: LinearEquation

Data FIelds: double: a,b,c,d,e,f

constructors and functions: double geta(), getb()...

bool isSolvable()

double getX(), double getY()

LinearEquation (double, double, double, double, double, double)

\*/

#include <iostream>

using namespace std;

#include "LinearEquation.h"

int main() {

//ask user for 6 values

double A, B, C, D, E, F;

cout << "Enter 6 values: " << endl;

cin >> A >> B >> C >> D >> E >> F;

cout << endl;

cout << "You entered the values: " << endl;

cout << A << " " << B << " " << C << " " << D << " " << E << " " << F << endl << endl;

//displays the results of the two linear equations

cout << "First Linear Equation: " << A << "x + " << B << "y = " << E << endl;

cout << "Second Linear Equation: " << C << "x + " << D << "y = " << F << endl << endl;

//report that the equation has no solution if ad-bc is 0

LinearEquation p2(A, B, C, D, E, F);

bool x = p2.isSolvable();

if (x == true) {

cout << "X = " << p2.getX() << endl;

cout << "Y = " << p2.getY() << endl;

}

else

cout << "The equation has no solution." << endl;

return 0;

}

#include <cmath>

#include <iostream>

using namespace std;

#ifndef LINEAREQUATION\_H

#define LINEAREQUATION\_H

class LinearEquation

{

//private data fields a,b,c,d,e,f

private:

double a, b, c, d, e, f;

public:

//a constructor with arguments for a,b,c,d,e,f

LinearEquation(double A1, double B1, double C1, double D1, double E1, double F1) {

a = A1;

b = B1;

c = C1;

d = D1;

e = E1;

f = F1;

}

//six get functions for a,b,c,d,e,f

double geta(){

return a;

}

double getb() {

return b;

}

double getc() {

return c;

}

double getd() {

return d;

}

double gete() {

return e;

}

double getf() {

return f;

}

//a function named isSolvable() that returns true if ad-bc is not 0

bool isSolvable() {

if ((a\*d) - (b\*c) == 0)

return false;

else

return true;

}

//functions getX() and getY() that return the solution for the equation

double getX() {

return ((e\*d) - (b\*f)) / ((a\*d) - (b\*c));

}

double getY() {

return ((a\*f) - (e\*c)) / ((a\*d) - (b\*c));

}

};

#endif

//////////////

//Kelsey Segren

//9.10

//EX03\_03

///////////////

#include "LinearEquation.h"

#include <iostream>

using namespace std;

//sample run: 2 2 5 -1 4 2 -1 -2

//sample run: 2 2 7 6 4 2 -1 -2

int main() {

//asks user for the endpoints

double x1, y1, x2, y2, x3, y3, x4, y4 = 0;

cout << "Enter values for x1, y1, x2, y2, x3, y3, x4, y4 " << endl;

cin >> x1 >> y1 >> x2 >> y2 >> x3 >> y3 >> x4 >> y4;

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

cout << "(x1, y1): " << "( " << x1 << " , " << y1 << " ) " << endl;

cout << "(x2, y2): " << "( " << x2 << " , " << y2 << " ) " << endl;

cout << "(x3, y3): " << "( " << x3 << " , " << y3 << " ) " << endl;

cout << "(x4, y4): " << "( " << x4 << " , " << y4 << " ) " << endl;

double a = y1 - y2;

double b = x1 - x2;

double c = y3 - y4;

double d = x3 - x4;

double e = (a \* x1 )- (b \* y1);

double f = (c \* x3) - (d \* y3);

//Finds the intersecting point

LinearEquation p2(a, b, c, d, e, f);

bool x = p2.isSolvable();

if (x == true) {

cout << "The lines intersect at: " << " ( " << p2.getX() << " , " << p2.getY() << " ) ";

}

else

cout << "The lines are parallel. " << endl;

}

#include <cmath>

#include <iostream>

using namespace std;

#ifndef LINEAREQUATION\_H

#define LINEAREQUATION\_H

class LinearEquation

{

//private data fields a,b,c,d,e,f

private:

double a, b, c, d, e, f;

public:

//a constructor with arguments for a,b,c,d,e,f

LinearEquation(double A1, double B1, double C1, double D1, double E1, double F1) {

a = A1;

b = B1;

c = C1;

d = D1;

e = E1;

f = F1;

}

//six get functions for a,b,c,d,e,f

double geta() {

return a;

}

double getb() {

return b;

}

double getc() {

return c;

}

double getd() {

return d;

}

double gete() {

return e;

}

double getf() {

return f;

}

//a function named isSolvable() that returns true if ad-bc is not 0

bool isSolvable() {

if ((a\*d) - (b\*c) == 0)

return false;

else

return true;

}

//functions getX() and getY() that return the solution for the equation

double getX() {

return ((e\*d) - (b\*f)) / ((a\*d) - (b\*c));

}

double getY() {

return ((a\*f) - (e\*c)) / ((a\*d) - (b\*c));

}

};

#endif

//////////

///Kelsey Segren

//EX04\_04

//10.4 Sort Characters in a string

///////////

#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

string sort(string& s);

int main() {

string s;

cout << "Enter a string s: ";

getline(cin, s);

cout << endl;

sort(s);

cout << "The sorted string is: " << s << endl;

}

string sort(string& s) {

sort(s.begin(), s.end());

return s;

}

////////////////

//Kelsey Segren

//EX03\_05

//10.10

/////////////

#include <iostream>

using namespace std;

#include "MyInteger.h"

/\*UML Diagram:

Class: MyInteger

Data Fields: int value

Constructors & Functions: MyInteger(int)

const int getValue

const bool isEven()

const bool isOdd()

const bool isPrime()

static bool isEven(int)

static bool isOdd(int)

static bool isPrime(int)

static bool isEven(const MyInteger&)

static bool isOff(const MyInteger&)

static bool isPrime(const MyInteger&)

const int equals(int)

const int equals(const MyInteger&)

static int parseInt(const string&)

\*/

int main() {

MyInteger p1(10);

MyInteger p2(7);

MyInteger p3(33);

MyInteger p4(22);

if (p1.isEven()) {

cout << "It is even. ";

}

if (MyInteger::isEven(10))

{

cout << "It works!" << endl;

}

if (p2.isPrime()) {

cout << "It is prime. ";

}

if (MyInteger::isPrime(7)) {

cout << "It works! " << endl;

}

if (p3.isOdd()) {

cout << "It is odd. ";

}

if (MyInteger::isOdd(33)) {

cout << "It works! " << endl;

}

return 0;

}

#ifndef MYINTEGER\_H\_

#define MYINTEGER\_H\_

#include <iostream>

#include <sstream>

using namespace std;

class MyInteger

{

//an int data field named value that stores the int value represented by this object

private:

int value;

public:

//A constructor that creates a MyInteger object for the specified int value

MyInteger(int anyValue) {

value = anyValue;

}

//a constant get function that returns the int value

const int getValue() {

return value;

}

//Constant functions isEven(), isOdd(), and isPrime() that return true if the //value is even, odd

// or prime, respectively

const bool isEven() {

if (value % 2 == 0)

return true;

else

return false;

}

const bool isOdd() {

if (value % 2 != 0)

return true;

else

return false;

}

const bool isPrime() {

for (int i = 2; i < value; i++)

{

if (value% i == 0)

return false;

else

return true;

}

}

//static functions isEven(int), isOdd(int), isPrime(int) that return true if the //specified value is even, odd, or prime, respectively

static bool isEven(int value)

{

if (value % 2 == 0)

return true;

else

return false;

}

static bool isOdd(int value) {

if (value % 2 != 0)

return true;

else

return false;

}

static bool isPrime(int value) {

for (int i = 2; i < value; i++)

{

if (value% i == 0)

return false;

else

return true;

}

}

//static functions isEven(const MyInteger&), isOdd(const MyInteger&), //isPrime(const MyInteger&) that

//return true if the value in the object is even, odd, or prime, respectively

static bool isEven(const MyInteger& I) {

return true;

}

static bool isOdd(const MyInteger& I) {

return true;

}

static bool isPrime(const MyInteger&) {

return true;

}

//constant functions equals(int) and equals(const MyInteger&) that return true if //the value

//in the object is equal to the specified value

const bool equals(int newValue) {

return value == newValue;

}

const bool equals(MyInteger& I) {

return equals(I.getValue());

}

//a static function parseInt(const string&) that converts a string to an int value

static int parseInt(const string& s)

{

stringstream ss(s);

int i;

ss >> i;

return i;

}

};

#endif