Course: Programming Fundamental - ENSF 480
Lab #: Lab 2
Instructor: G. Gouri
Student Name: Daniel Rey, Aly Farouz
Lab Section: B01
Date Submitted: September 22, 2025

Exercise A

```
/*
* File: dictionaryList.cpp (inside the class definition)
*/
    friend ostream& operator << (ostream& os, DictionaryList& dl);
    bool operator !=(const DictionaryList& rhs);
    bool operator >(const DictionaryList& rhs);
    bool operator <(const DictionaryList& rhs);
    bool operator <=(const DictionaryList& rhs);
    bool operator >=(const DictionaryList& rhs);
    Datum operator[](int i) const;

/*
* File: dictionaryList.cpp
*/
ostream& operator << (ostream& os, DictionaryList& dl)
{
    os << dl.cursor_datum();
    return os;
}
bool DictionaryList::operator !=(const DictionaryList& rhs)
{
   return !(this->cursor_datum() == rhs.cursor_datum());
}
bool DictionaryList::operator >(const DictionaryList& rhs)
{
   return this->cursor_datum() > rhs.cursor_datum();
}
bool DictionaryList::operator <(const DictionaryList& rhs)
{
   return this->cursor_datum() < rhs.cursor_datum();
}
bool DictionaryList::operator <=(const DictionaryList& rhs)
{
   return !(this->cursor_datum() > rhs.cursor_datum());
}
bool DictionaryList::operator >=(const DictionaryList& rhs)
{
   return !(this->cursor_datum() < rhs.cursor_datum());
}
//above few functions were implemented by Farouz
Datum DictionaryList::operator[](int i)const{ //will cause an error if i >= sizeM
    Node* p=headM;
    while(i>0){
        p=p->nextM;
        i--;
    }
    return p->datumM;
}
```

```
Last login: Thu Sep 18 15:50:45 on ttys000
(base) alyfarouz@Alys-MacBook-Air exA % g++ -Wall -o myprog dictionaryList.cpp Lab2_exAmain.cpp

(base) alyfarouz@Alys-MacBook-Air exA % ./myprog
[
Printing list just after its creation ...
  List is EMPTY.

Printing list after inserting 3 new keys ...
  8001  Dilbert
  8002  Alice
  8003  Wally

Printing list after removing two keys and inserting PointyHair ...
  8003  Wally
  8004  PointyHair

Printing list after changing data for one of the keys ...
  8003  Sam
  8004  PointyHair

Printing list after inserting 2 more keys ...
  8001  Allen
  8002  Peter
  8003  Sam
  8004  PointyHair
***----Finished dictionary tests--------------------------***

(base) alyfarouz@Alys-MacBook-Air exA %
```

Exercise B

```
/*
* File Name: graphicsWorld.h
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#ifndef GRAPHICSWORLD_H
#define GRAPHICSWORLD_H
#include "point.h"
#include "shape.h"
#include "square.h"
#include "rectangle.h"
using namespace std;

class GraphicsWorld {
public:
  GraphicsWorld();
  ~GraphicsWorld();
  static void run();
};
#endif



/*
* File Name: graphicsWorld.cpp
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/
```

```
#include <iostream>
#include "graphicsWorld.h"
using namespace std;

void GraphicsWorld::run(){
  #if 1 // Change 0 to 1 to test Point
    Point m (6, 8);
    Point n (6,8);
    n.setx(9);
    cout << "\nExpected to dispaly the distance between m and n is: 3";
    cout << "\nThe distance between m and n is: " << m.distance(n);
    cout << "\nExpected second version of the distance function also print: 3";
    cout << "\nThe distance between m and n is again: "
         << Point::distance(m, n);
  #endif // end of block to test Point
  #if 1 // Change 0 to 1 to test Square
    cout << "\n\nTesting Functions in class Square:" <<endl;
    Square s(5, 7, 12, "SQUARE - S");
    s.display();
  #endif // end of block to test Square
  #if 1 // Change 0 to 1 to test Rectangle
    cout << "\nTesting Functions in class Rectangle:" << endl;
    Rectangle a(5, 7, 12, 15, "RECTANGLE A");
    a.display();
    Rectangle b(16 , 7, 8, 9, "RECTANGLE B");
    b.display();
    double d = a.distance(b);
    cout <<"\nDistance between square a, and b is: " << d << endl;
    Rectangle rec1 = a;
    rec1.display();
    cout << "\nTesting assignment operator in class Rectangle:" <<endl;
    Rectangle rec2 (3, 4, 11, 7, "RECTANGLE rec2");
    rec2.display();
    rec2 = a;
    a.set_side_b(200);
    a.set_side_a(100);
    cout << "\nExpected to display the following values for objec rec2: " << endl;
    cout << "Rectangle Name: RECTANGLE A\n" << "X-coordinate: 5\n" << "Y-coordinate: 7\n"
         << "Side a: 12\n" << "Side b: 15\n" << "Area: 180\n" << "Perimeter: 54\n" ;
    cout << "\nIf it doesn't there is a problem with your assignment operator.\n" << endl;
    rec2.display();

    cout << "\nTesting copy constructor in class Rectangle:" <<endl;
    Rectangle rec3 (a);
    rec3.display();
    a.set_side_b(300);
    a.set_side_a(400);
    cout << "\nExpected to display the following values for objec rec2: " << endl;
    cout << "Rectangle Name: RECTANGLE A\n" << "X-coordinate: 5\n" << "Y-coordinate: 7\n"
    << "Side a: 100\n" << "Side b: 200\n" << "Area: 20000\n" << "Perimeter: 600\n" ;
    cout << "\nIf it doesn't there is a problem with your copy constructor.\n" << endl;
    rec3.display();
  #endif // end of block to test Rectangle
  #if 1 // Change 0 to 1 to test using array of pointer and polymorphism
    cout << "\nTesting array of pointers and polymorphism:" <<endl;
    Shape* sh[4];
    sh[0] = &s;
    sh[1] = &b;
    sh [2] = &rec1;
    sh [3] = &rec3;
    sh [0]->display();
    sh [1]->display();
    sh [2]->display();
    sh [3]->display();
  #endif // end of block to test array of pointer and polymorphism
}

#if 1
main(){GraphicsWorld::run();}
// g++ -Wall graphicsWorld.cpp point.cpp shape.cpp square.cpp rectangle.cpp -o exB.exe
#endif
```

```
/*
* File Name: point.h
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#ifndef POINT_H
#define POINT_H
using namespace std;

class Point{
private:
  double x;
  double y;
  static int count;
  const int id;
public:
  Point(double x, double y);
  Point(const Point& other);
  Point& operator=(const Point& rhs);
  double getx() const;
  void setx(double a);
  double gety() const;
  void sety(double a);

  void display()const;
  const static int counter();
  static double distance(const Point& the_point, const Point& other);
  double distance(const Point& other)const;
};
#endif
```

```
/*
* File Name: point.cpp
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;

#ifndef POINT_H
#include "point.h"
int Point::count = 0;
#endif


Point::Point(double h, double v): x(h), y(v), id(++count+1000){}

Point::Point(const Point& other): x(other.getx()), y(other.gety()), id(++count+1000){}

Point& Point::operator=(const Point& rhs){
  x=rhs.getx();
  y=rhs.gety();
  return *this;
}

double Point::getx()const{return x;}
void Point::setx(double a){x=a;}
double Point::gety()const{return y;}
```

```
void Point::sety(double a){y=a;}

void Point::display()const{
  cout << "X-coordinate: ";
  cout << right << setw(9) << fixed << setprecision(2) << this->x << endl;
  cout << "Y-coordinate: ";
  cout << right << setw(9) << fixed << setprecision(2) << this->y << endl;
}

const int Point::counter(){return count;}

double Point::distance(const Point& the_point, const Point& other){
  double dx=the_point.getx()-other.getx();
  double dy=the_point.gety()-other.gety();
  return sqrt(dx*dx+dy*dy);
}
double Point::distance(const Point& other)const{
  double dx=this->x-other.getx();
  double dy=this->y-other.gety();
  return sqrt(dx*dx+dy*dy);
}
```

```
/*
* File Name: shape.h
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#ifndef SHAPE_H
#define SHAPE_H
#include "point.h"
using namespace std;

class Shape{
protected:
  Point origin;
  char* shapeName;
public:
  Shape(double x, double y, const char* name);
  Shape(const Shape& other);
  virtual ~Shape();
  Shape& operator=(const Shape& rhs);
  const Point& getOrigin() const;
  const char* getName() const;

  virtual void display();
  double distance(const Shape& other)const;
  static double distance(const Shape& the_shape, const Shape& other);
  void move(double dx, double dy);
};
#endif
```

```
/*
* File Name: shape.cpp
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#include <iostream>
#include <cstring>
#include "shape.h"
using namespace std;
```

```
Shape::Shape(double x, double y, const char* name):
origin(Point(x,y)),
shapeName(new char[strlen(name)+1]){
  strcpy(this->shapeName, name);
}

Shape::Shape(const Shape& other):
origin(Point(other.getOrigin())),
shapeName(new char[strlen(other.getName())+1]){
  strcpy(this->shapeName, other.getName());
}

Shape::~Shape(){
  delete [] shapeName;
}

Shape& Shape::operator=(const Shape& rhs){
  if (this!=&rhs){
    delete [] shapeName;
    shapeName = new char[strlen(rhs.getName())+1];
    strcpy(this->shapeName, rhs.getName());
    origin = rhs.getOrigin();
  }
  return *this;
}

const Point& Shape::getOrigin()const{
  const Point& p=origin;
  return p;
}

const char* Shape::getName()const{return shapeName;}


void Shape::display(){
  cout << "Shape Name: " << shapeName << endl;
  origin.display();
}

double Shape::distance(const Shape& other)const{
  return Point::distance(this->origin, other.getOrigin());
}
double Shape::distance(const Shape& the_shape, const Shape& other){
  return Point::distance(the_shape.getOrigin(), other.getOrigin());
}

void Shape::move(double dx, double dy){
  origin.setx(origin.getx()+dx);
  origin.sety(origin.gety()+dy);
}



/*
* File Name: square.h
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#ifndef SQUARE_H
#define SQUARE_H
#include "shape.h"
using namespace std;
```

```
class Square: public Shape{
protected:
  double side_a;
public:
  Square(double x, double y, double a, char* name);
  Square(const Square& other);
  Square& operator =(const Square& rhs);
  const double get_side_a()const;
  void set_side_a(double a);

  double area()const;
  double perimeter()const;
  virtual void display()const;
};
#endif
```

```
/*
* File Name: square.cpp
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#include <iostream>
#include "square.h"
using namespace std;

Square::Square(double x, double y, double a, char* name): Shape(x,y,name), side_a(a){}

Square::Square(const Square& other): Shape(other), side_a(other.get_side_a()){}

Square& Square::operator=(const Square& rhs){
  if (this!=&rhs){
    Shape::operator=(rhs);
    side_a = rhs.get_side_a();
  }
  return *this;
}

const double Square::get_side_a()const{return side_a;}
void Square::set_side_a(double a){side_a=a;}

double Square::area()const{return side_a*side_a;}
double Square::perimeter()const{return side_a*4;}

void Square::display()const{
  cout << "Square Name: " << shapeName << endl;
  origin.display();
  cout << "Side a: " << side_a << endl
       << "Area: " << area() << endl
       << "Perimeter: " << perimeter() << endl;
}
```

```
/*
* File Name: rectangle.h
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#ifndef RECTANGLE_H
#define RECTANGLE_H
#include "square.h"
using namespace std;
```

```
class Rectangle: public Square{
private:
  double side_b;
public:
  Rectangle(double x, double y, double a, double b, char* name);
  Rectangle(const Rectangle& other);
  Rectangle& operator =(const Rectangle& rhs);
  const double get_side_b()const;
  void set_side_b(double b);

  double area()const;
  double perimeter()const;
  void display()const;
};
#endif
```

```
/*
* File Name: rectangle.cpp
* Assignment: Lab 2, Exercise B
* Lab Section: B01
* Completed by: Daniel Rey
* Submission Date: Sept 22, 2025
*/

#include <iostream>
#include "rectangle.h"
using namespace std;

Rectangle::Rectangle(double x, double y, double a, double b, char* name):
Square(x,y,a,name),
side_b(b){}

Rectangle::Rectangle(const Rectangle& other):
Square(other),
side_b(other.get_side_b()){}

Rectangle& Rectangle::operator=(const Rectangle& rhs){
  if (this!=&rhs){
    Square::operator=(rhs);
    side_b = rhs.get_side_b();
  }
  return *this;
}

const double Rectangle::get_side_b()const{return side_b;}
void Rectangle::set_side_b(double b){side_b=b;}

double Rectangle::area()const{return side_a*side_b;}
double Rectangle::perimeter()const{return side_a*2+side_b*2;}

void Rectangle::display()const{
  cout << "Rectangle Name: " << shapeName << endl;
  origin.display();
  cout << "Side a: " << side_a << endl
       << "Side b: " << side_b << endl
       << "Area: " << area() << endl
       << "Perimeter: " << perimeter() << endl;
}
```

```
HPLaptop@DESKTOP-C6NJ9VH ~/ENSF480/lab2
$ g++ -Wall graphicsWorld.cpp point.cpp shape.cpp square.cpp rectangle.cpp -o exB.exe
graphicsWorld.cpp: In static member function ‘static void GraphicsWorld::run()’:
```

```
graphicsWorld.cpp:26:24: warning: ISO C++ forbids converting a string constant to ‘char*’
[-Wwrite-strings]
   26 |     Square s(5, 7, 12, "SQUARE - S");
      |                        ^~~~~~~~~~~~
graphicsWorld.cpp:31:31: warning: ISO C++ forbids converting a string constant to ‘char*’
[-Wwrite-strings]
   31 |     Rectangle a(5, 7, 12, 15, "RECTANGLE A");
      |                               ^~~~~~~~~~~~~
graphicsWorld.cpp:33:31: warning: ISO C++ forbids converting a string constant to ‘char*’
[-Wwrite-strings]
   33 |     Rectangle b(16 , 7, 8, 9, "RECTANGLE B");
      |                               ^~~~~~~~~~~~~
graphicsWorld.cpp:40:34: warning: ISO C++ forbids converting a string constant to ‘char*’
[-Wwrite-strings]
   40 |     Rectangle rec2 (3, 4, 11, 7, "RECTANGLE rec2");
      |                                  ^~~~~~~~~~~~~~~~

HPLaptop@DESKTOP-C6NJ9VH ~/ENSF480/lab2
$ ./exB.exe


Testing Functions in class Square:
Square Name: SQUARE - S
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 12.00
Area: 144.00
Perimeter: 48.00

Testing Functions in class Rectangle:
Rectangle Name: RECTANGLE A
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 12.00
Side b: 15.00
Area: 180.00
Perimeter: 54.00
Rectangle Name: RECTANGLE B
X-coordinate:     16.00
Y-coordinate:      7.00
Side a: 8.00
Side b: 9.00
Area: 72.00
Perimeter: 34.00

Distance between square a, and b is: 11.00
Rectangle Name: RECTANGLE A
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 12.00
Side b: 15.00
Area: 180.00
Perimeter: 54.00
```

```
Testing assignment operator in class Rectangle:
Rectangle Name: RECTANGLE rec2
X-coordinate:      3.00
Y-coordinate:      4.00
Side a: 11.00
Side b: 7.00
Area: 77.00
Perimeter: 36.00

Expected to display the following values for objec rec2:
Rectangle Name: RECTANGLE A
X-coordinate: 5
Y-coordinate: 7
Side a: 12
Side b: 15
Area: 180
Perimeter: 54

If it doesn't there is a problem with your assignment operator.

Rectangle Name: RECTANGLE A
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 12.00
Side b: 15.00
Area: 180.00
Perimeter: 54.00

Testing copy constructor in class Rectangle:
Rectangle Name: RECTANGLE A
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 100.00
Side b: 200.00
Area: 20000.00
Perimeter: 600.00

Expected to display the following values for objec rec2:
Rectangle Name: RECTANGLE A
X-coordinate: 5
Y-coordinate: 7
Side a: 100
Side b: 200
Area: 20000
Perimeter: 600

If it doesn't there is a problem with your copy constructor.

Rectangle Name: RECTANGLE A
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 100.00
Side b: 200.00
Area: 20000.00
Perimeter: 600.00

Testing array of pointers and polymorphism:
```

```
Square Name: SQUARE - S
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 12.00
Area: 144.00
Perimeter: 48.00
Rectangle Name: RECTANGLE B
X-coordinate:     16.00
Y-coordinate:      7.00
Side a: 8.00
Side b: 9.00
Area: 72.00
Perimeter: 34.00
Rectangle Name: RECTANGLE A
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 12.00
Side b: 15.00
Area: 180.00
Perimeter: 54.00
Rectangle Name: RECTANGLE A
X-coordinate:      5.00
Y-coordinate:      7.00
Side a: 100.00
Side b: 200.00
Area: 20000.00
Perimeter: 600.00
```