ENSF 614 – Fall 2021

Lab 5 – Tuesday, October 26

Student Name: Aastha Patel and Bhavyai Gupta

Submission date: October 26, 2021

Exercise B – Source file point.h

```
/*
 * File Name:               point.h
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise B
 * Lab section:             B01
 * Completed by:            Aastha Patel, Bhavyai Gupta
 * Submission Date:         October 26, 2021
 */

#ifndef POINT_H
#define POINT_H

class Point
{
private:
    double x;
    double y;
    int id;
    static int num_of_objects;

public:
    Point(double a, double b);
    // REQUIRES
    //    two arguments of type double
    // PROMISES
    //    creates Point object with arguments a and b of double type
    //    assigns appropriate id to the object created
    //    increments num_of_objects

    ~Point();
    // PROMISES
    //    destroys the Point object and decrements num_of_objects

    Point(const Point &P);
    // REQUIRES
    //    reference of Point object as argument P
    // PROMISES
    //    creates Point object with deep copy of data members of P
    //    assigns appropriate id to the object created
    //    increments num_of_objects

    Point &operator=(const Point &rhs);
```

```cpp
    // REQUIRES
    //    reference of Point object on right hand side of =
    // PROMISES
    //    deep copy of data members of rhs to object being created
    //    assigns appropriate id to the object being created
    //    increments num_of_objects

    void display() const;
    // PROMISES
    //    prints the Point object on stdout

    double getx() const;
    // PROMISES
    //    returns the x co-ordinate of Point

    double gety() const;
    // PROMISES
    //    returns the y co-ordinate of Point

    void setx(double a);
    // REQUIRES
    //    an argument of type double
    // PROMISES
    //    sets the x co-ordinate of Point as a

    void sety(double b);
    // REQUIRES
    //    an argument of type double
    // PROMISES
    //    sets the y co-ordinate of Point as b

    int counter() const;
    // PROMISES
    //    returns the num_of_objects

    double distance(const Point &P) const;
    // REQUIRES
    //    reference to Point object
    // PROMISES
    //    returns the distance between this Point and P on the cartesian plane

    static double distance(const Point &P, const Point &Q);
    // REQUIRES
    //    two references to Point objects as arguments
    // PROMISES
```

```
    //      returns the distance between P and Q on the cartesian plane
};

#endif
```

Exercise B – Source file point.cpp

```cpp
/*
 * File Name:               point.cpp
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise B
 * Lab section:             B01
 * Completed by:            Aastha Patel, Bhavyai Gupta
 * Submission Date:         October 26, 2021
 */

#include "point.h"
#include <stdio.h>
#include <math.h>
using namespace std;

int Point::num_of_objects = 0;

Point::Point(double x, double y)
{
    this->x = x;
    this->y = y;
    this->id = ++num_of_objects + 1000;
}

Point::Point(const Point &P)
{
    this->x = P.getx();
    this->y = P.gety();
    this->id = ++num_of_objects + 1000;
}

Point &Point::operator=(const Point &rhs)
{
    if (this != &rhs)
    {
        this->x = rhs.getx();
        this->y = rhs.gety();
        this->id = ++num_of_objects + 1000;
    }

    return *this;
}
```

```cpp
Point::~Point()
{
    --num_of_objects;
}

void Point::display() const
{
    printf("X-coordinate: %.2f\n", getx());
    printf("Y-coordinate: %.2f\n", gety());
}

double Point::getx() const
{
    return this->x;
}

double Point::gety() const
{
    return this->y;
}

void Point::setx(double x)
{
    this->x = x;
}

void Point::sety(double y)
{
    this->y = y;
}

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

double Point::distance(const Point &P) const
{
    double dx2 = pow((this->getx() - P.getx()), 2);
    double dy2 = pow((this->gety() - P.gety()), 2);

    return sqrt(dx2 + dy2);
}

double Point::distance(const Point &P, const Point &Q)
```

```
{
    double dx2 = pow((P.getx() - Q.getx()), 2);
    double dy2 = pow((P.gety() - Q.gety()), 2);

    return sqrt(dx2 + dy2);
}
```

Exercise B – Source file shape.h

```cpp
/*
 * File Name:               shape.h
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise B
 * Lab section:             B01
 * Completed by:            Aastha Patel, Bhavyai Gupta
 * Submission Date:         October 26, 2021
 */

#include "point.h"

#ifndef SHAPE_H
#define SHAPE_H

class Shape
{
protected:
    Point origin;
    char *shapeName;

public:
    Shape(double x, double y, const char *shapeName);
    // REQUIRES
    //     two arguments of type double and a pointer to built-in string
    // PROMISES
    //     creates Shape object with the supplied arguments

    virtual ~Shape();
    // PROMISES
    //     destroys the Shape object
    //     deallocates the memory referenced by shapeName

    Shape(const Shape &s);
    // REQUIRES
    //     reference of Shape object as argument s
    // PROMISES
    //     creates Shape object with deep copy of data members of s

    Shape& operator=(const Shape &rhs);
    // REQUIRES
    //     reference of Shape object on right hand side of =
    // PROMISES
```

```cpp
    //     deep copy of data members of rhs to object being created

    const Point &getOrigin() const;
    // PROMISES
    //     returns reference to origin

    const char *getName() const;
    // PROMISES
    //     returns pointer to shapeName

    virtual void display() const;
    // PROMISES
    //     prints the Shape object on stdout

    virtual double distance(Shape &S) const;
    // REQUIRES
    //     reference to Shape object
    // PROMISES
    //     returns the distance between this Shape and S on the cartesian plane

    static double distance(Shape &S, Shape &T);
    // REQUIRES
    //     two references to Shape objects as arguments
    // PROMISES
    //     returns the distance between S and T on the cartesian plane

    virtual double area() const = 0;
    // PROMISES
    //     returns the area of the Shape

    virtual double perimeter() const = 0;
    // PROMISES
    //     returns the perimeter of the Shape

    void move (double dx, double dy);
    // PROMISES
    //     updates the origin of Shape by moving co-ordinates by (dx, dy)
};

#endif
```

Exercise B – Source file shape.cpp

```cpp
/*
 * File Name:               shape.cpp
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise B
 * Lab section:             B01
 * Completed by:            Aastha Patel, Bhavyai Gupta
 * Submission Date:         October 26, 2021
 */

#include "shape.h"
#include "point.h"
#include <stdio.h>
#include <string.h>
using namespace std;

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

Shape::~Shape()
{
    delete[] this->shapeName;
    this->shapeName = nullptr;
}

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

Shape &Shape::operator=(const Shape &rhs)
{
    if (this != &rhs)
    {
        delete[] this->shapeName;

        this->origin = Point(rhs.getOrigin().getx(), rhs.getOrigin().gety());
        this->shapeName = new char[strlen(rhs.getName()) + 1];
```

```cpp
        strcpy(this->shapeName, rhs.getName());
    }

    return *this;
}

void Shape::display() const
{
    printf("Shape Name  : %s\n", this->getName());
    this->getOrigin().display();
}

const Point &Shape::getOrigin() const
{
    return this->origin;
}

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

double Shape::distance(Shape &S) const
{
    double dist = this->getOrigin().distance(S.getOrigin());
    return dist;
}

double Shape::distance(Shape &S, Shape &T)
{
    double dist = S.getOrigin().distance(S.getOrigin(), T.getOrigin());
    return dist;
}

void Shape::move(double dx, double dy)
{
    double old_x = this->getOrigin().getx();
    double old_y = this->getOrigin().gety();

    origin.setx(old_x + dx);
    origin.sety(old_y + dy);
}
```

Exercise B – Source file square.h

```c
/*
 * File Name:                square.h
 * Course:                   ENSF 614 - Fall 2021
 * Lab # and Assignment #:   Lab 5 Exercise B
 * Lab section:              B01
 * Completed by:             Aastha Patel, Bhavyai Gupta
 * Submission Date:          October 26, 2021
 */

#include "shape.h"
#include "point.h"

#ifndef SQUARE_H
#define SQUARE_H

class Square : virtual public Shape
{
protected:
    double side_a;

public:
    Square(double x, double y, double side, const char *shapeName);
    // REQUIRES
    //    three arguments of type double and a pointer to built-in string
    // PROMISES
    //    creates Square object with the supplied arguments

    double area() const;
    // PROMISES
    //    returns area of the square

    double perimeter() const;
    // PROMISES
    //    returns perimeter of the square

    double get_side_a() const;
    // PROMISES
    //    returns side_a of the Square

    void set_side_a(double side);
    // REQUIRES
    //    an argument of type double
```

```
    // PROMISES
    //    sets the side_a of Square as side

    void display() const;
    // PROMISES
    //    prints the Square object to stdout
};

#endif
```

Exercise B – Source file square.cpp

```cpp
/*
 * File Name:               square.cpp
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise B
 * Lab section:             B01
 * Completed by:            Aastha Patel, Bhavyai Gupta
 * Submission Date:         October 26, 2021
 */

#include "square.h"
#include "shape.h"
#include "point.h"
#include <stdio.h>

using namespace std;

Square::Square(double x, double y, double side, const char *shapeName): Shape(x,
y, shapeName)
{
    this->set_side_a(side);
}

double Square::area() const
{
    return this->get_side_a() * this->get_side_a();
}

double Square::perimeter() const
{
    return this->get_side_a() * 4;
}

double Square::get_side_a() const
{
    return this->side_a;
}

void Square::set_side_a(double side)
{
    this->side_a = side;
}
```

```cpp
void Square::display() const
{
    printf("Square Name: %s\n", this->getName());
    this->getOrigin().display();
    printf("Side a: %.2f\n", this->get_side_a());
    printf("Area: %.2f\n", this->area());
    printf("Perimeter: %.2f\n", this->perimeter());
}
```

Exercise B – Source file rectangle.h

```cpp
/*
 * File Name:               rectangle.h
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise B
 * Lab section:             B01
 * Completed by:            Aastha Patel, Bhavyai Gupta
 * Submission Date:         October 26, 2021
 */

#include "square.h"

#ifndef RECTANGLE_H
#define RECTANGLE_H

class Rectangle : public Square
{
protected:
    double side_b;

public:
    Rectangle(double x, double y, double a, double b, const char *shapeName);
    // REQUIRES
    //     four arguments of type double and a pointer to built-in string
    // PROMISES
    //     creates Rectangle object with the supplied arguments

    double area() const;
    // PROMISES
    //     returns area of the rectangle

    double perimeter() const;
    // PROMISES
    //     returns perimeter of the rectangle

    double get_side_b() const;
    // PROMISES
    //     returns side_b of the Rectangle

    void set_side_b(double side);
    // REQUIRES
    //     an argument of type double
    // PROMISES
```

```
    //      sets the side_b of Rectangle as side

    void display() const;
    // PROMISES
    //      prints the Rectangle object to stdout
};

#endif
```

Exercise B – Source file rectangle.cpp

```cpp
/*
 * File Name:              rectangle.cpp
 * Course:                 ENSF 614 - Fall 2021
 * Lab # and Assignment #: Lab 5 Exercise B
 * Lab section:            B01
 * Completed by:           Aastha Patel, Bhavyai Gupta
 * Submission Date:        October 26, 2021
 */

#include "rectangle.h"
#include "square.h"
#include "shape.h"
#include "point.h"
#include <stdio.h>

using namespace std;

Rectangle::Rectangle(double x, double y, double a, double b, const char
*shapeName): Shape(x, y, shapeName), Square(x, y, a, shapeName)
{
    this->set_side_b(b);
}

double Rectangle::area() const
{
    return (this->get_side_a() * this->get_side_b());
}

double Rectangle::perimeter() const
{
    return (2*(this->get_side_a() + this->get_side_b()));
}

double Rectangle::get_side_b() const
{
    return this->side_b;
}

void Rectangle::set_side_b(double side)
{
    this->side_b = side;
}
```

```cpp
void Rectangle::display() const
{
    printf("Rectangle Name: %s\n", this->getName());
    this->getOrigin().display();
    printf("Side a: %.2f\n", this->get_side_a());
    printf("Side b: %.2f\n", this->get_side_b());
    printf("Area: %.2f\n", this->area());
    printf("Perimeter: %.2f\n", this->perimeter());
}
```

Exercise B – Source file graphicsWorld.h

```cpp
/*
 * File Name:              graphicsWorld.h
 * Course:                 ENSF 614 - Fall 2021
 * Lab # and Assignment #: Lab 5 Exercise B
 * Lab section:            B01
 * Completed by:           Aastha Patel, Bhavyai Gupta
 * Submission Date:        October 26, 2021
 */

#ifndef GRAPHICSWORLD_H
#define GRAPHICSWORLD_H

class GraphicsWorld
{
public:
    static void run();
    // PROMISES
    //    tests various functionalities implemented and print results on stdout
};

#endif
```

Exercise B – Source file graphicsWorld.cpp

```cpp
/*
 * File Name:              graphicsWorld.cpp
 * Course:                 ENSF 614 - Fall 2021
 * Lab # and Assignment #: Lab 5 Exercise B
 * Lab section:            B01
 * Completed by:           Aastha Patel, Bhavyai Gupta
 * Submission Date:        October 26, 2021
 */

#include "graphicsWorld.h"
#include "rectangle.h"
#include "square.h"
#include "shape.h"
#include "point.h"
#include <iostream>

using namespace std;

void GraphicsWorld::run()
{
    cout << "Authors: Aastha Patel, Bhavyai Gupta" << endl;

    // #if 0  // Change 0 to 1 to test Point
    Point m(6, 8);
    Point n(6, 8);
    n.setx(9);
    cout << "\nExpected to display 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 0  // 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 0  // Change 0 to 1 to test Rectangle
    cout << "\nTesting Functions in class Rectangle:" << endl;
    Rectangle a(5, 7, 12, 15, "RECTANGLE A");
    a.display();
```

```cpp
    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 assignment operator.\n"
        << endl;
    rec3.display();
    // #endif // end of block to test Rectangle
```

```cpp
    // #if 0  // 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
}
```

Exercise B – Source file lab5ExB.cpp

```cpp
/*
 * File Name:               lab5ExB.cpp
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise B
 * Lab section:             B01
 * Completed by:            Aastha Patel, Bhavyai Gupta
 * Submission Date:         October 26, 2021
 */

#include "graphicsWorld.h"

using namespace std;

int main(int argc, char const *argv[])
{
    GraphicsWorld::run();
    return 0;
}
```
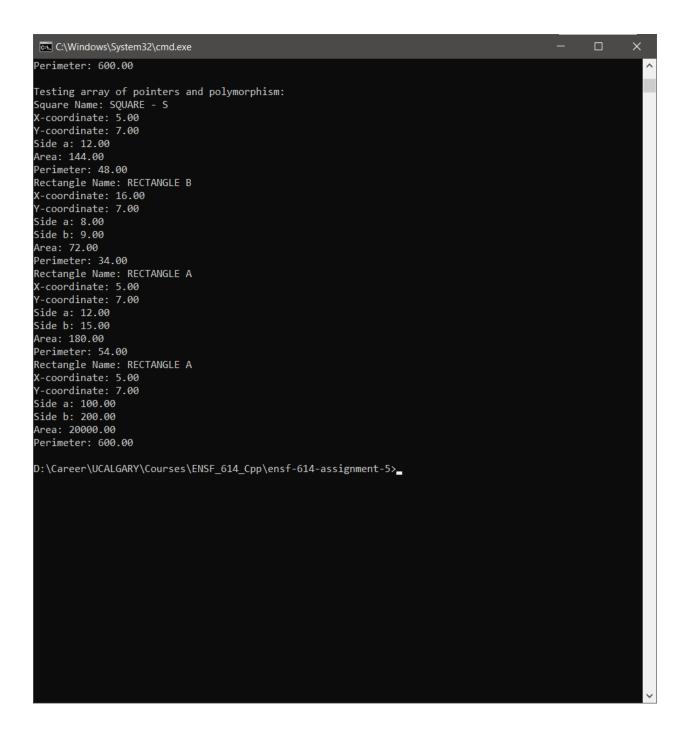
Exercise B – Program Output

```
C:\Windows\System32\cmd.exe                                                    —    □    ✕

D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-5>g++ -Wall lab5ExB.cpp point.cpp shape.cpp
square.cpp rectangle.cpp graphicsWorld.cpp -o lab5ExB.exe

D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-5>.\lab5ExB.exe
Authors: Aastha Patel, Bhavyai Gupta

Expected to display the distance between m and n is: 3
The distance between m and n is: 3
Expected second version of the distance function also print: 3
The distance between m and n is again: 3

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
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
```

```
C:\Windows\System32\cmd.exe                                    —    □    ✕

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 assignment operator.

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

```
C:\Windows\System32\cmd.exe                                    —    □    ✕

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

D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-5>_
```

Exercise C – Source file circle.h

```cpp
/*
 * File Name:                circle.h
 * Course:                   ENSF 614 - Fall 2021
 * Lab # and Assignment #:   Lab 5 Exercise C
 * Lab section:              B01
 * Completed by:             Aastha Patel, Bhavyai Gupta
 * Submission Date:          October 26, 2021
 */

#include "rectangle.h"
#include "shape.h"

#ifndef CIRCLE_H
#define CIRCLE_H

class Circle : virtual public Shape
{
protected:
    double radius;

public:
    Circle(double x, double y, double r, const char *shapeName);
    // REQUIRES
    //     three args x,y for origin r for radius and a char pointer for name
    // PROMISES
    //     create Circle object from the given args

    double area() const;
    // PROMISES
    //     calculate & return the area of circle

    double perimeter() const;
    // PROMISES
    //     calculate and returns the perimeter of circle

    double get_radius() const;
    // PROMISES
    //     return the radius of circle

    void set_radius(double radius);
    // REQUIRES
    //     double arg for radius
```

```cpp
    // PROMISES
    //    set the radius of circle

    void display() const;
    // PROMISES
    //    display Circle's origin, area, perimeter and name
};

#endif
```

Exercise C – Source file circle.cpp

```cpp
/*
 * File Name:              circle.cpp
 * Course:                 ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise C
 * Lab section:            B01
 * Completed by:           Aastha Patel, Bhavyai Gupta
 * Submission Date:        October 26, 2021
 */

#include "circle.h"
#include "shape.h"
#include "point.h"

#include <stdio.h>
#include <math.h>
using namespace std;

Circle::Circle(double x, double y, double r, const char *shapeName) : Shape(x, y,
shapeName)
{
    this->set_radius(r);
}

double Circle::area() const
{
    return (M_PI * pow(this->get_radius(), 2));
}

double Circle::perimeter() const
{
    return (2 * M_PI * this->get_radius());
}

double Circle::get_radius() const
{
    return this->radius;
}

void Circle::set_radius(double r)
{
    this->radius = r;
}
```

```cpp
void Circle::display() const
{
    printf("Circle Name: %s\n", this->getName());
    this->getOrigin().display();
    printf("Radius: %.2f\n", this->get_radius());
    printf("Area: %.2f\n", this->area());
    printf("Perimeter: %.2f\n", this->perimeter());
}
```

Exercise C – Source file curvecut.h

```cpp
/*
 * File Name:              curvecut.h
 * Course:                 ENSF 614 - Fall 2021
 * Lab # and Assignment #: Lab 5 Exercise C
 * Lab section:            B01
 * Completed by:           Aastha Patel, Bhavyai Gupta
 * Submission Date:        October 26, 2021
 */

#include "circle.h"
#include "rectangle.h"

#ifndef CURVECUT_H
#define CURVECUT_H

class CurveCut : public Circle, public Rectangle
{
protected:
    double width;

public:
    CurveCut(double x, double y, double a, double w, double r, const char
*shapeName);
    // REQUIRES
    //    five args x, y for origin a, w for rectangle r for circle and a char
pointer for name
    // PROMISES
    //    create CurveCut object from the given args

    double area() const;
    // PROMISES
    //    calculate & return the area of curvecut

    double perimeter() const;
    // PROMISES
    //    calculate & return the perimeter of curvecut

    void display() const;
    // PROMISES
    //    displays CurveCut's origin, length, width, radius and name
};
```

```
#endif
```

Exercise C – Source file curvecut.cpp

```cpp
/*
 * File Name:              curvecut.cpp
 * Course:                 ENSF 614 - Fall 2021
 * Lab # and Assignment #: Lab 5 Exercise C
 * Lab section:            B01
 * Completed by:           Aastha Patel, Bhavyai Gupta
 * Submission Date:        October 26, 2021
 */

#include "curvecut.h"
#include "circle.h"
#include "shape.h"
#include "point.h"

#include <stdio.h>
#include <math.h>
using namespace std;

CurveCut::CurveCut(double x, double y, double w, double l, double r, const char
*shapeName) : Shape(x, y, shapeName), Circle(x, y, r, shapeName), Rectangle(x, y,
w, l, shapeName)
{
    double minLength = w < l ? w : l;

    if (r > minLength)
    {
        fprintf(stderr, "\n[FAIL] The radius of the circle must be always less
than or equal the smaller of the width and length. Exit!\n");
        exit(1);
    }
}

double CurveCut::area() const
{
    return (Rectangle::area() - (Circle::area() / 4));
}

double CurveCut::perimeter() const
{
    return Rectangle::perimeter() - (2 * this->get_radius()) +
(Circle::perimeter() / 4);
}
```

```cpp
void CurveCut::display() const
{
    printf("CurveCut Name: %s\n", this->getName());
    this->getOrigin().display();
    printf("Width: %.2f\n", this->get_side_a());
    printf("Length: %.2f\n", this->get_side_b());
    printf("Radius of the cut: %.2f\n", this->get_radius());
}
```

Exercise C – Updated source file graphicsWorld.cpp

```cpp
/*
 * File Name:              graphicsWorld.cpp
 * Course:                 ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise B and C
 * Lab section:            B01
 * Completed by:           Aastha Patel, Bhavyai Gupta
 * Submission Date:        October 26, 2021
 */

#include "graphicsWorld.h"
#include "curvecut.h"
#include "circle.h"
#include "rectangle.h"
#include "square.h"
#include "shape.h"
#include "point.h"
#include <iostream>

using namespace std;

void GraphicsWorld::run()
{
    cout << "Authors: Aastha Patel, Bhavyai Gupta" << endl;

    cout << "\n\n+----------------+\n";
    cout << "|   EXERCISE B   |\n";
    cout << "+----------------+\n\n";

    // #if 0  // Change 0 to 1 to test Point
    Point m(6, 8);
    Point n(6, 8);
    n.setx(9);
    cout << "\nExpected to display 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 0  // Change 0 to 1 to test Square
    cout << "\n\nTesting Functions in class Square:" << endl;
    Square s(5, 7, 12, "SQUARE - S");
```

```cpp
    s.display();
    // #endif // end of block to test Square

    // #if 0  // 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"
```

```cpp
              << "Side b: 200\n"
              << "Area: 20000\n"
              << "Perimeter: 600\n";
        cout << "\nIf it doesn't there is a problem with your assignment operator.\n"
              << endl;
        rec3.display();
        // #endif // end of block to test Rectangle

        // #if 0  // 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


        cout << "\n\n+----------------+\n";
        cout << "|   EXERCISE C   |\n";
        cout << "+----------------+\n\n";



        // #if 0
        cout << "\nTesting Functions in class Circle:" << endl;
        Circle c(3, 5, 9, "CIRCLE C");
        c.display();
        cout << "the area of " << c.getName() << " is: " << c.area() << endl;
        cout << "the perimeter of " << c.getName() << " is: " << c.perimeter() <<
endl;
        d = a.distance(c);
        cout << "\nThe distance between rectangle a and circle c is: " << d << endl;

        CurveCut rc(6, 5, 10, 12, 9, "CurveCut rc");
        rc.display();
        cout << "the area of " << rc.getName() << " is: " << rc.area() << endl;
        cout << "the perimeter of " << rc.getName() << " is: " << rc.perimeter();
        d = rc.distance(c);
        cout << "\nThe distance between rc and c is: " << d << endl;
        // Using array of Shape pointers:
```

```cpp
    // Shape *sh[4];
    sh[0] = &s;
    sh[1] = &a;
    sh[2] = &c;
    sh[3] = &rc;
    sh[0]->display();
    cout << "The area of " << sh[0]->getName() << " is: " << sh[0]->area();
    cout << "\nthe perimeter of " << sh[0]->getName() << " is: " << sh[0]-
>perimeter() << endl << endl;
    sh[1]->display();
    cout << "\nThe area of " << sh[1]->getName() << " is: " << sh[1]->area();
    cout << "\nthe perimeter of " << sh[0]->getName() << " is: " << sh[1]-
>perimeter() << endl << endl;
    sh[2]->display();
    cout << "\nThe area of " << sh[2]->getName() << " is: " << sh[2]->area();
    cout << "\nthe circumference of " << sh[2]->getName() << " is: " << sh[2]-
>perimeter() << endl << endl;
    sh[3]->display();
    cout << "\nThe area of " << sh[3]->getName() << " is: " << sh[3]->area();
    cout << "\nthe perimeter of " << sh[3]->getName() << " is: " << sh[3]-
>perimeter() << endl << endl;
    cout << "\nTesting copy constructor in class CurveCut:" << endl;
    CurveCut cc = rc;
    cc.display();
    cout << "\nTesting assignment operator in class CurveCut:" << endl;
    CurveCut cc2(2, 5, 100, 12, 9, "CurveCut cc2");
    cc2.display();
    cc2 = cc;
    cc2.display();
    // #endif
}
```

Exercise C – Source file lab5ExC.cpp

```cpp
/*
 * File Name:               lab5ExC.cpp
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 5 Exercise C
 * Lab section:             B01
 * Completed by:            Aastha Patel, Bhavyai Gupta
 * Submission Date:         October 26, 2021
 */

#include "graphicsWorld.h"

using namespace std;

int main(int argc, char const *argv[])
{
    GraphicsWorld::run();
    return 0;
}
```
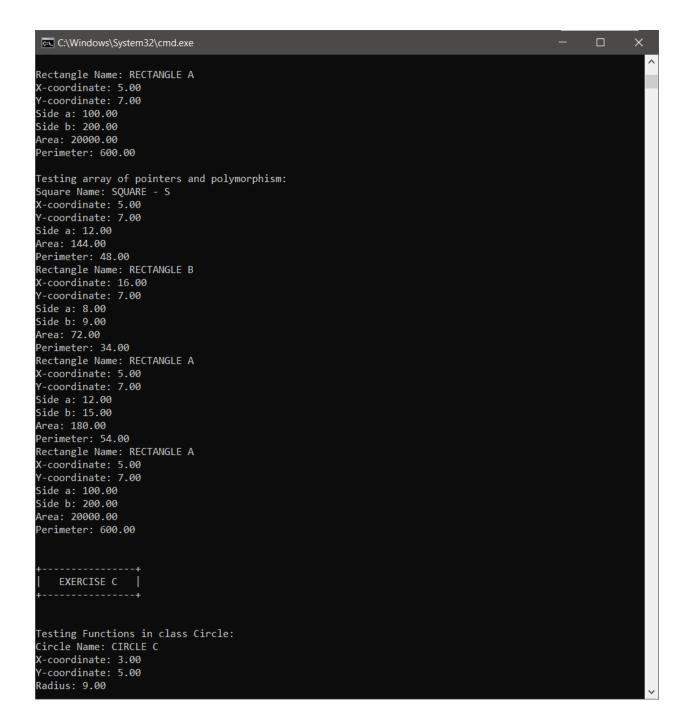
Exercise C – Program Output

```
D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-5>g++ -Wall lab5ExC.cpp point.cpp shape.cpp
square.cpp rectangle.cpp circle.cpp curvecut.cpp graphicsWorld.cpp -o lab5ExC.exe

D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-5>.\lab5ExC.exe
Authors: Aastha Patel, Bhavyai Gupta


+----------------+
|    EXERCISE B  |
+----------------+


Expected to display the distance between m and n is: 3
The distance between m and n is: 3
Expected second version of the distance function also print: 3
The distance between m and n is again: 3

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
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 assignment operator.
```

```
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


+----------------+
|    EXERCISE C  |
+----------------+


Testing Functions in class Circle:
Circle Name: CIRCLE C
X-coordinate: 3.00
Y-coordinate: 5.00
Radius: 9.00
```

```
Area: 254.47
Perimeter: 56.55
the area of CIRCLE C is: 254.469
the perimeter of CIRCLE C is: 56.5487

The distance between rectangle a and circle c is: 2.82843
CurveCut Name: CurveCut rc
X-coordinate: 6.00
Y-coordinate: 5.00
Width: 10.00
Length: 12.00
Radius of the cut: 9.00
the area of CurveCut rc is: 56.3827
the perimeter of CurveCut rc is: 40.1372
The distance between rc and c is: 3
Square Name: SQUARE - S
X-coordinate: 5.00
Y-coordinate: 7.00
Side a: 12.00
Area: 144.00
Perimeter: 48.00
The area of SQUARE - S is: 144
the perimeter of SQUARE - S is: 48

Rectangle Name: RECTANGLE A
X-coordinate: 5.00
Y-coordinate: 7.00
Side a: 400.00
Side b: 300.00
Area: 120000.00
Perimeter: 1400.00

The area of RECTANGLE A is: 120000
the perimeter of SQUARE - S is: 1400

Circle Name: CIRCLE C
X-coordinate: 3.00
Y-coordinate: 5.00
Radius: 9.00
Area: 254.47
Perimeter: 56.55

The area of CIRCLE C is: 254.469
the circumference of CIRCLE C is: 56.5487

CurveCut Name: CurveCut rc
X-coordinate: 6.00
Y-coordinate: 5.00
Width: 10.00
```

```
Length: 12.00
Radius of the cut: 9.00

The area of CurveCut rc is: 56.3827
the perimeter of CurveCut rc is: 40.1372


Testing copy constructor in class CurveCut:
CurveCut Name: CurveCut rc
X-coordinate: 6.00
Y-coordinate: 5.00
Width: 10.00
Length: 12.00
Radius of the cut: 9.00

Testing assignment operator in class CurveCut:
CurveCut Name: CurveCut cc2
X-coordinate: 2.00
Y-coordinate: 5.00
Width: 100.00
Length: 12.00
Radius of the cut: 9.00
CurveCut Name: CurveCut rc
X-coordinate: 6.00
Y-coordinate: 5.00
Width: 10.00
Length: 12.00
Radius of the cut: 9.00

D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-5>
```