lab1

Bhupinder Singh

February 8, 2019

Contents

1 lab1.h **1**

2 main.cpp **4**

1 lab1.h

Lab Goal

```
The following program is supposed to give the students
a review on the basic concepts of C++ and get them warmed up before
starting to work more on data structure related topics.  In order to do
this, the lab requires the students to use structures and arrays which
are the fundamentals of studying data structures.
```

Following are the libraries and namespace used by the program.

```
#include <iostream>
#include <cmath>

using namespace std;
```

Here, we intialize the required structures in order to
get the desired results.
Point --> has two double values: x and y;
Triangle1 --> has 3 points: A, B and C
Triangle2 --> has two arrays with three elements each: x and y.

```
double distance();
double perimeter();
double perimeter2();

struct Point{
        public:
                double x;
                double y;
};

struct Triangle1{
        public:
                Point a;
                Point b;
                Point c;
};
```

Triangle2 is a speciality in this program, the logic works the same way
as traingle1, but the difference being triangle2 uses arrays instead of
Points.
In the next file: main.cpp, we will see the specific function
that was made for this structure.

```
struct Triangle2{
        public:
                double x[3];
                double y[3];
                Triangle2(double x1,double y1,double x2,double y2,double x3,
                    double y3);
};
```

2 main.cpp

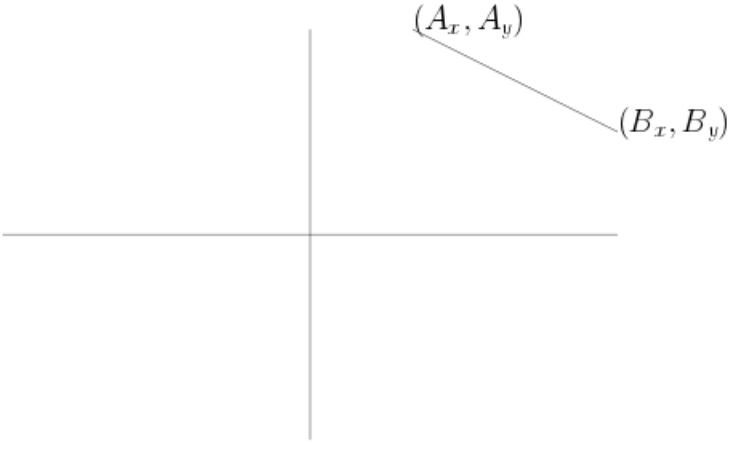
```
#include "lab1.h"


This is the constructor that intializes all the array
elements to the respective coordinate values.

Triangle2::Triangle2(double x1,double y1,double x2,double y2,double x3,
    double y3)
{
        x[0] = x1;
        y[0] = y1;
        x[1] = x2;
        y[1] = y2;
        x[2] = x3;
        y[2] = y3;
}
```

Function that calculates the distance between two points: A and B
The logic used is the distance formula which is available online.

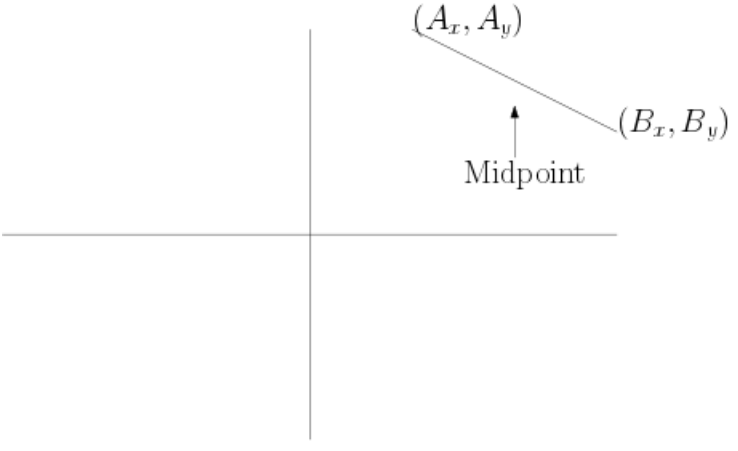
$$Distance = \sqrt{(A_x - B_x)^2 + (A_y - B_y)^2}$$




```
double distance (Point a, Point b)
{
        double xpart, ypart, res;
        xpart = a.x - b.x;
        ypart = a.y - b.y;
        xpart = pow(xpart, 2);
        ypart = pow(ypart, 2);
        res = xpart + ypart;
        return(sqrt(res));
}
```

Function that calculates the midpoint given two points: A and B
The midpoint formula was also found through online resources.
It adds both the respective coordinates and divides them by 2 to get
the midpoint between the points.

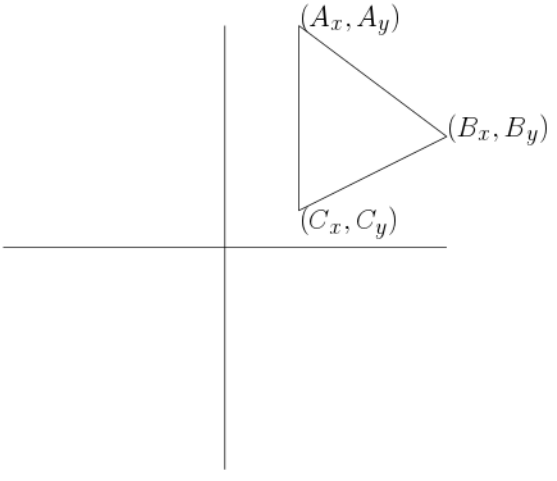
$$Distance = \sqrt{(A_x - B_x)^2 + (A_y - B_y)^2}$$




```
Point midpoint(Point a, Point b)
{
        Point temp;
        temp.x = (a.x + b.x)/2;
        temp.y = (a.y + b.y)/2;
        return(temp);
}
```

Function that calculates the perimeter of a triangle:
given 3 points: A, B and C.
The function calls the previously coded distance function and adds the
values from that function to output the resulting perimeter.

Perimeter = (Distance from A to B) + (Distance from B to C) + (Distance from C to A)




```
double perimeter(Point a, Point b, Point c)
{
        double dist1, dist2, dist3;
        dist1 = distance(a,b);
        dist2 = distance(b,c);
        dist3 = distance(c,a);
        return(dist1 + dist2 + dist3);
}
```

The Following function is specifically made for Exercise 4.This function takes the array values from main and assigns them to points.
The reason of doing this is easing the workload of the programmer, by doing this, I did not have to make another function that needs to calculate the same result that a previous function already did.
After they are assigned to points, the previous perimeter function gets called that calculates the perimeter again and outputs the result.

```
double perimeter2 (Triangle2 tri)
{
        Point a;
        a.x = tri.x[0]; a.y = tri.y[0];
        Point b;
        b.x = tri.x[1]; b.y = tri.y[1];
        Point c;
        c.x = tri.x[2]; c.y = tri.y[2];
        return(perimeter(a,b,c));
}
```

```
Function main, takes in all the values from the user and outputs the
desired results.
In this program, the only purpose of main is to get the necessary values
from the user, call the respective function and output the result.

main()
{
        double result_d, result_p;
        Point a, b, c, result_m;

        cout << "Hello, please enter the x-cordinate of first point." <<
            endl;
        cin >> a.x;
        cout << "Please enter the y-cordinate of first point." << endl;
        cin >> a.y;
        cout << "Please enter the x-cordinate of second point." << endl;
        cin >> b.x;
        cout << "Please enter the y-cordinate of second point." << endl;
        cin >> b.y;
        result_d = distance(a,b);
        cout << "The distance between the two points is " << result_d <<
            endl;
        result_m = midpoint(a,b);
        cout << "The midpoint is ("<< result_m.x << ", "<< result_m.y << ")"
             <<endl;
        cout << "Please enter the x-component of third cordinate." << endl;
        cin >> c.x;
        cout << "Please enter the y-component of third cordinate." << endl;
        cin >> c.y;
        result_p = perimeter(a,b,c);
```

```
        cout << "The perimeter of the triangle is " << result_p << endl;
        Triangle2 tri(a.x, a.y, b.x, b.y, c.x, c.y);
        cout << "The perimeter of the trainlge is " << perimeter2(tri);

        return 0;
}
```

Here is the sample output of the program!