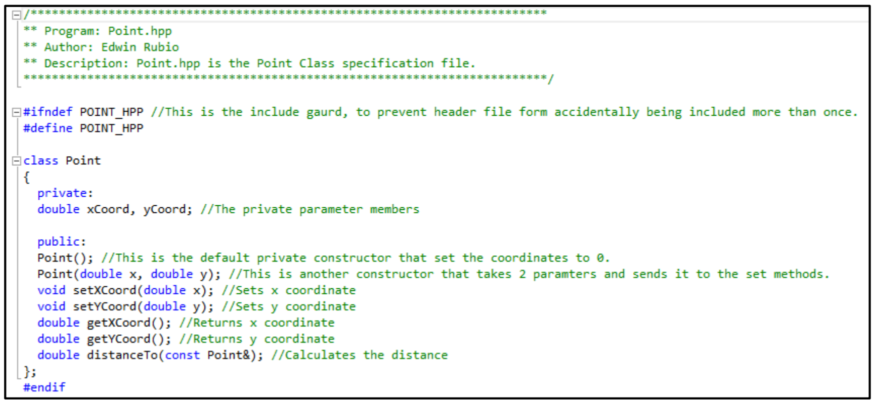
Assignment 6 (Comparisons)

Name: Edwin Rubio

For each pairwise comparison, where do you think your program is better? Why? Where do you think the other program is better? Why?

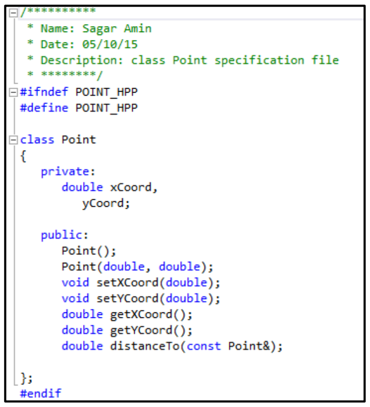
First Comparison

When comparing Sagar Amin source code against mine I quickly saw that my code had more descriptive comments, see figures 1 & 2. Having comments in the in the source code helps to understand the flow of the code. This is beneficial for other programmers reviewing my code, or if I review my own code after some time. Also this is a good vital programming skill for the real world.



```
/*************************************************************************
** Program: Point.hpp
** Author: Edwin Rubio
** Description: Point.hpp is the Point Class specification file.
*************************************************************************/

#ifndef POINT_HPP //This is the include gaurd, to prevent header file form accidentally being included more than once.
#define POINT_HPP

class Point
{
  private:
  double xCoord, yCoord; //The private parameter members

  public:
  Point(); //This is the default private constructor that set the coordinates to 0.
  Point(double x, double y); //This is another constructor that takes 2 paramters and sends it to the set methods.
  void setXCoord(double x); //Sets x coordinate
  void setYCoord(double y); //Sets y coordinate
  double getXCoord(); //Returns x coordinate
  double getYCoord(); //Returns y coordinate
  double distanceTo(const Point&); //Calculates the distance
};
#endif
```

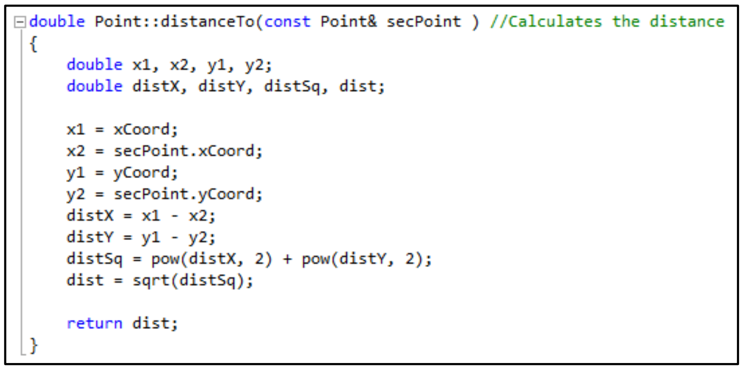
Figure 1 - My Point.hpp File



```
/*********
 * Name: Sagar Amin
 * Date: 05/10/15
 * Description: class Point specification file
 * ********/
#ifndef POINT_HPP
#define POINT_HPP

class Point
{
   private:
      double xCoord,
         yCoord;

   public:
      Point();
      Point(double, double);
      void setXCoord(double);
      void setYCoord(double);
      double getXCoord();
      double getYCoord();
      double distanceTo(const Point&);

};
#endif
```

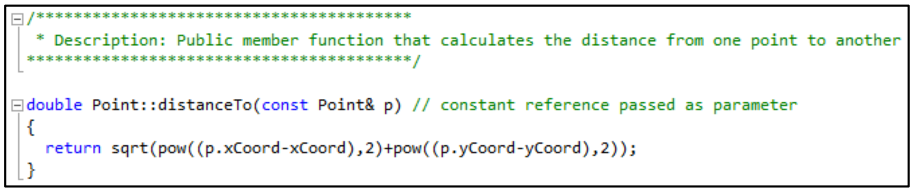
Figure 2 - Sagar Amin's Point.hpp File

On the other hand, Sagar's "distanceTo" Point class method is more efficient than mine, see figure 3 & 4. The reason is that I declared more double type variables in the method which takes up more memory space and more processing time. Sagar's "distanceTo" method was really straight forward since it passed the referenced object which does not require utilizing more memory space and processing time.



```
double Point::distanceTo(const Point& secPoint ) //Calculates the distance
{
    double x1, x2, y1, y2;
    double distX, distY, distSq, dist;

    x1 = xCoord;
    x2 = secPoint.xCoord;
    y1 = yCoord;
    y2 = secPoint.yCoord;
    distX = x1 - x2;
    distY = y1 - y2;
    distSq = pow(distX, 2) + pow(distY, 2);
    dist = sqrt(distSq);

    return dist;
}
```

Figure 3 - My "distanceTo" Method

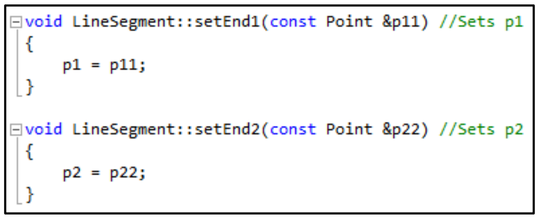


```
/*****************************************
 * Description: Public member function that calculates the distance from one point to another
*****************************************/

double Point::distanceTo(const Point& p) // constant reference passed as parameter
{
  return sqrt(pow((p.xCoord-xCoord),2)+pow((p.yCoord-yCoord),2));
}
```

Figure 4 - Sagar's "distanceTo" Method

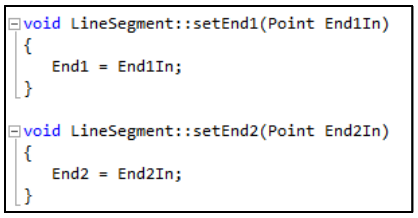
Second Comparison

When comparing Jason DiMedio source code against mine I noticed that my "setEnd1" and "setEnd2" mutators methods that belong to "LineSegment" class were more efficient. Since I passed the "LineSegment" variables via reference as compared to Jason's that passes it by value, see figures 5 & 6. The advantage here is that passing by reference there is no need to use more memory space or processing time.



```
void LineSegment::setEnd1(const Point &p11) //Sets p1
{
    p1 = p11;
}

void LineSegment::setEnd2(const Point &p22) //Sets p2
{
    p2 = p22;
}
```

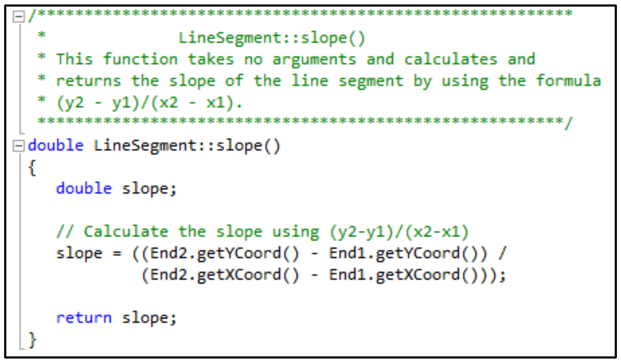
Figure 5 - My LineSegment Class Mutators



```
void LineSegment::setEnd1(Point End1In)
{
   End1 = End1In;
}

void LineSegment::setEnd2(Point End2In)
{
   End2 = End2In;
}
```

Figure 6 - Jason's LineSegment Class Mutators

On the other side, Jason's comments were exceptional since he described every method with great detail, see figure 7. You can see in his comments he specifies if any arguments are used, as well mentions what value is being returned back, and formula used.

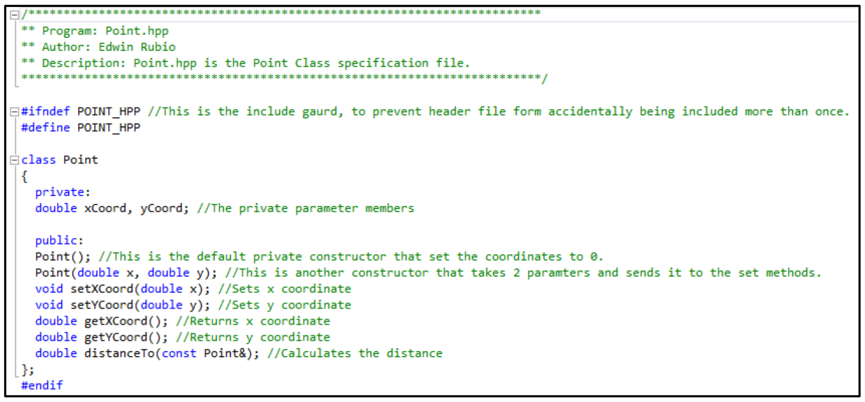


```
/**********************************************************
 *              LineSegment::slope()
 * This function takes no arguments and calculates and
 * returns the slope of the line segment by using the formula
 * (y2 - y1)/(x2 - x1).
 *********************************************************/
double LineSegment::slope()
{
   double slope;

   // Calculate the slope using (y2-y1)/(x2-x1)
   slope = ((End2.getYCoord() - End1.getYCoord()) /
            (End2.getXCoord() - End1.getXCoord()));

   return slope;
}
```

Figure 7 - Jason's Exceptional Comments

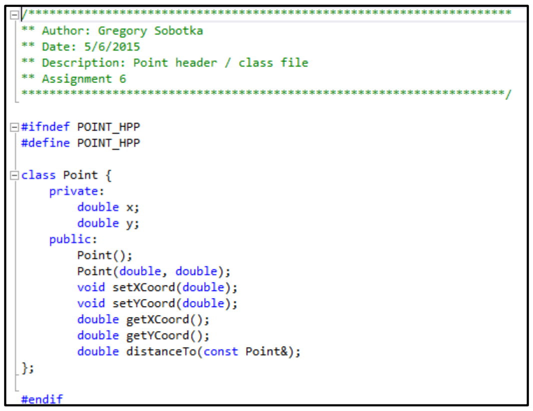
Third Comparison

When comparing Gregory Sobotka source code against mine I noticed that all of his class specification files did not have no documentation/comments explaining the class. This is like releasing a new product to the market without no documentation. In the real world, this is a must since it is most likely to be used by other programmers. Saying this, my source code for the class specifications are better documented, see images 8 & 9.



```
/**********************************************************************
** Program: Point.hpp
** Author: Edwin Rubio
** Description: Point.hpp is the Point Class specification file.
**********************************************************************/

#ifndef POINT_HPP //This is the include gaurd, to prevent header file form accidentally being included more than once.
#define POINT_HPP

class Point
{
  private:
  double xCoord, yCoord; //The private parameter members

  public:
  Point(); //This is the default private constructor that set the coordinates to 0.
  Point(double x, double y); //This is another constructor that takes 2 paramters and sends it to the set methods.
  void setXCoord(double x); //Sets x coordinate
  void setYCoord(double y); //Sets y coordinate
  double getXCoord(); //Returns x coordinate
  double getYCoord(); //Returns y coordinate
  double distanceTo(const Point&); //Calculates the distance
};
#endif
```

Figure 8 - My Point Class Specification



```
/*******************************************************************
** Author: Gregory Sobotka
** Date: 5/6/2015
** Description: Point header / class file
** Assignment 6
*******************************************************************/

#ifndef POINT_HPP
#define POINT_HPP

class Point {
    private:
        double x;
        double y;
    public:
        Point();
        Point(double, double);
        void setXCoord(double);
        void setYCoord(double);
        double getXCoord();
        double getYCoord();
        double distanceTo(const Point&);
};

#endif
```

Figure 9 - Gregory Sobotka Point Class Specification

On the other hand, his consistency on indentation (formatting) was perfect when compared to my indentation. His indentation allowed to visually see what belongs to what. For example, when comparing the Point Class specifications you can see that I did not use indentation, see figure 8, below the access specifiers. This can cause confusion when other programmers are reviewing my source code.

What have you learned from looking at other people's code, and how can you apply it in future assignments?

Viewing other programmer's code allows me to see things in a broader view. By seeing things in a broader view allows me to make hard to implement functions to an easier logic. But I have to keep in mind that it is not easy to follow other programmer's source code since they might not have the same logic view as I do. This brings up that adding comments to the source code help other programmers to see our logic in the code, and for future reference. In overall, looking at other programmer's code allows me to see many different possibilities of logic, and that adding comments into the source code helps to explain the logic.