```
Script started on Wed 26 Apr 2017 11:55:33 AM CDT
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ pwd
/home/students/b_pepa/CSC122/point
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ cat point.info
Brandon Pepa
CSC122-001
Operate on this!
Lab

Levels:
(Level 2)

**(Level 2)**

Description:
        This program overloads some useful function for the point
class. This makes the class easier to use and more intitive (for most)
and requires less typing(more efficient coding).
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ cat point.h
#ifndef POINT_CLASS_HEADER_INCLUDED
#define POINT_CLASS_HEADER_INCLUDED

#include <iostream>

const double SIGF = 0.0001; //Represents the number of significant figures in comparis
on.


// A 2D point class
class Point
{
   double x, // x coordinate of point
          y; // y coordinate of point

public:
   Point(void) : x(0), y(0) {}
   Point(double new_x, double new_y) : x(new_x), y(new_y) {}
   Point(const Point & other) : x(other.x) , y(other.y) {}

   void Output(std::ostream & os = std::cout) const;
   friend std::ostream & operator<<(std::ostream & os, const Point & p)
      { p.Output(os); return os; }

   void Input(std::istream & is = std::cin);
   friend std::istream & operator>>(std::istream & is, Point & p)
      { p.Input(is); return is; }

   double distance(const Point & other) const;
   double operator-(const Point & p) const
      { Point q(*this); return q.distance(p); }

   double get_x(void) const { return x; }
   double get_y(void) const { return y; }
   double operator[](char c)
      { return c == 'x' ? get_x() : (c == 'y' ? get_y() : 0); }

   Point midpoint( const Point & p)
      { Point t( (get_x() + p.get_x())/2 , (get_y() + p.get_y())/2 );
        return t; }
   Point operator/(const Point & q)
      { Point p(*this); return p.midpoint(q); }

   void set_x(double new_x);
   void set_y(double new_y);

   Point flip_x(void) const;
   Point flip_y(void) const;

   Point shift_x(double move_by) const;
   Point shift_y(double move_by) const;

   bool operator==(const Point & q)
      {  Point p(*this);
         return ((p.get_x() - q.get_x()) < SIGF
             &&  (p.get_y() - q.get_y()) < SIGF); }
   bool operator!=(const Point & q)
      {  Point p(*this);
         return !(p == q); }


};

#endif
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ cat point.cpp
#include "point.h"

#include <iostream>
#include <cmath>

void Point::Input(std::istream & is)
{
   char dummy;
   is >> dummy >> x >> dummy >> y >> dummy;
   return;
}

// output standard 2D point notation (x,y)
void Point::Output(std::ostream & os) const
{
   os << '(' << x << ", " << y << ')';
   return;
}

// calculate distance between two 2D points --
// the one that called us and the argument
double Point::distance(const Point & other) const
{
   return sqrt(pow(other.x-x, 2.0) +
               pow(other.y-y, 2.0));
}

// set coordinates to programmer-specified values
void Point::set_x(double new_x)
{
   x = new_x;        // no error checking since anything is legal
   return;
}

// set coordinates to programmer-specified values
void Point::set_y(double new_y)
{
   y = new_y;        // no error checking since anything is legal
   return;
}

// creates a point flipped about the x axis from us
Point Point::flip_x(void) const
{
```

```
   return Point(x,-y);
}

// creates a point flipped about the y axis from us
Point Point::flip_y(void) const
{
   return Point(-x,y);
}

// creates a point shifted along the x axis from us
Point Point::shift_x(double move_by) const
{
   return Point(x+move_by,y);
}

// creates a point shifted along the y axis from us
Point Point::shift_y(double move_by) const
{
   return Point(x,y+move_by);
}
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ cat test.cpp
#include "point.h"
#include <iostream>
using namespace std;

int main(void)
{
   Point p,q;

   cout << "\tWelcome to the overloaded point testing program"
        << "\n" << string(60, '*')
        << "\nPlease enter the coordinates for your first point\n";
   cin >> p;
   cout << "Please enter the coordinates for your second point\n";
   cin >> q;

   Point r(p/q);

   cout << "\nThe distance between " << p
        << " and " << q << " is:\n" << (q-p)
        << "\nThe midpoint is " << r;

   cout << "\n\nThese points are equal: "
        << (p==q)? string("true") : string("false");
   cout << "\nThese points are not equal: "
        << (p!=q)? string("true") : string("false");
   cout << endl;


return 0;
}
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ \033[KCPP test point
point.cpp...
test.cpp***


\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ ./\033[Ktest.out
        Welcome to the overloaded point testing program
**********************************************************
Please enter the coordinates for your first point
(1,0)
Please enter the coordinates for your second point
(1,1)

The distance between (0, 0) and (1, 1) is:
1.41421
The midpoint is (0.5, 0.5)

These points are equal: 1
These points are not equal: 0
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ ./test.out
        Welcome to the overloaded point testing program
**********************************************************
Please enter the coordinates for your first point
(0,01,01.2)
Please enter the coordinates for your second point
(0.01,1.2)

The distance between (0.01, 1.2) and (0.01, 1.2) is:
0
The midpoint is (0.01, 1.2)

These points are equal: 1
These points are not equal: 0
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ ./test.out
        Welcome to the overloaded point testing program
**********************************************************
Please enter the coordinates for your first point
(-4.5,4.5)
Please enter the coordinates for your second point
(4.5,-4.5)

The distance between (-4.5, 4.5) and (4.5, -4.5) is:
12.7279
The midpoint is (0, 0)

These points are equal: 0
These points are not equal: 1
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ ./test.out
        Welcome to the overloaded point testing program
**********************************************************
Please enter the coordinates for your first point
(7,8)
Please enter the coordinates for your second point
(4,3)

The distance between (7, 8) and (4, 4) is:
5
The midpoint is (5.5, 6)

These points are equal: 0
These points are not equal: 1
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ cat test\033[K033[K033[K033[K
point.tpq
Thought Provoking Questions
1.Which opeators are members and which are non-members? Do any have to be members?
    operator-, operator==, operator!=, and operator/ are all members becuase
    the calling object is of the same class. operator>> and operator<< are
    non-members. operators that are called from a point object have to be members

2.Which operators should be const? What other methods might well be made const?
  In general, what is the rule which determines if a method should be made const?
    operator-, operator<<, operator==, operator!=, and operator/ shoule all be made
    const. a method should be const if it does not need to change the data in the clas
s.

3.What type do equality and inequality return? Input? Output? Assignment?
    equality and inequality return a bool, input returns an istream, output
```

```
    returns an ostream, assignments returns a reference of point

4.Do you agree with your friend’s decision to use operator/ for midpoint? Why/Why not?
    operator overloading is supposed to match as close to the actual operator as possi
ble
    using / for midpoint suggests that one point divided by another point yields a mid
point
    which is not really a good parallel to division... oh also there’s no way to speci
fy
    which axis you want to shift, or flip. do you want both to be flipped? or shifted?

5.Why didn’t you overload operators for less than, greater than, etc.?
    These operators weren’t overloaded because what does it reall mean if a point
    is greater than another point? what if the x is greater than the other x, but the
    y is less than the other y? they can’t be greater than or less than in that situat
ion

6.Your friend wanted to overload operators frot eh flip and shift methods, too
  (~ and += respectively). Why did you talk them out of it? Why wasn’t this a good ide
a?
    This wasn’t a good idea, becuase it doesn’t represent at all what the actual opera
tor~
    and += do for normal numbers. ~is a bitwise inverse operator, and += is supposed t
o be
    assignment.
\033]0;b_pepa@mars:~/CSC122/point\007[b_pepa@mars point]$ exit
exit

Script done on Wed 26 Apr 2017 11:58:38 AM CDT
```