```
Script started on 2021-11-24 16:28:32-06:00 [TERM="xterm" TTY="/dev/pts/0" COLUMNS=
k_gandhi6@ares:~$ cat Point.info
Name: Kush Gandhi
Class: CSC122-001
Titlle: Operate on this! (lab)
Levels: 2
Description: Uses overloaded operators to update the point class
and wrote a test main for testing midpoint, distance, and equal or not.
k_gandhi6@ares:~$ cat Point.h
#ifndef POINT_CLASS_HEADER_INCLUDED
#define POINT_CLASS_HEADER_INCLUDED
#include <iostream>

// A 2D point class
class Point
{
    double x, // x cordinate of point
           y; //y cordinate 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; //output this point
    friend std::ostream& operator << (std::ostream & os, const Point & a)
        { a.Output(os); return os; };

    void Input(std::istream & is = std::cin); //input this point
    friend std::istream & operator>>(std::istream & is, Point & a)
        { a.Input(is); return is; }

    double distance(const Point & other) const; // distance between this point and
    double operator-(const Point& a) const
        { Point b(*this); return b.distance(a); }

    double get_x(void) const { return x; }
    double get_y(void) const { return y; }

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

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

    Point midPoint(const Point& p)//find the midpoint
    {
        Point mp((get_x() + p.get_x()) / 2, (get_y() + p.get_y()) / 2);
        return mp;
    }

    Point operator/(const Point & b)
    {
        Point a(*this);
        return a.midPoint(b);
    }

    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& b)//to check ==
    {
        Point a(*this);
        return ((a.get_x() == b.get_x())
            && (a.get_y() == b.get_y()));

    }
    bool operator!=(const Point& b)//check !=
    {
        Point a(*this);
        return !(a == b);
    }
};

#endif
k_gandhi6@ares:~$ cat Point.cpp
#include <iostream>
#include <cmath>
#include "Point.h"
using namespace std;

//input function for (x,y) format
void Point::Input(std::istream& is)
{
    char temp;
    is >> temp >> x >> temp >> y >> temp;
    return;
}
//output for (x,y) format
void Point::Output(std::ostream& os) const
{
    os << '(' << x << ", " << y << ')';
    return;
}
//calculate the distance between 2 points
double Point::distance(const Point& other)const
{
    return sqrt(pow(other.x - x, 2.0) +
           pow(other.y - y, 2.0));
}
//using setters functions
void Point::set_x(double new_x)
{
    x = new_x;
    return;
}
```

```
void Point::set_y(double new_y)
{
    y = new_y;
    return;
}

//function to flip points
Point Point::flip_x(void) const
    { return Point(x, -y); }
Point Point::flip_y(void) const
    { return Point(-x, y); }

//function to move points for x and y
Point Point::shift_x(double move_by) const
    { return Point(x + move_by, y); }
Point Point::shift_y(double move_by) const
    { return Point(x, y + move_by); }
k_gandhi6@ares:~$ cat PointTest.cpp
#include <iostream>
#include "Point.h"
using namespace std;

int main()
{
    Point a, b;

    cout << "\t\tTest For Point Class" << endl
         << "================================================="
         << "\nEnter your 1st cordinate point: ";
    cin >> a;
    cout << "\nEnter your 2nd point: ";
    cin >> b;

    Point check(a / b);

    cout << "\nThe distance between " << a << " and "
         << b << " is: " << (a - b) << "\nThe midpoint is: " << check;

    cout << "\n\nTesting the points if equal or not equal..." << endl;
    //0 = false, 1 = true
    cout << "\nAre the points equal? " << (a == b);
    cout << "\nAre the points not equal? " << (a != b);
    cout << "\n=================================================" << endl;
    cout << endl;
}
k_gandhi6@ares:~$ CPP Point PointTest
Point.cpp...
PointTest.cpp***
In file included from Point.cpp:3:
Point.h: In member function ‘bool
Point::operator==(const Point&)’:
Point.h:57:28: warning: comparing
floating-point with ‘==’ or ‘!=’
is unsafe [-Wfloat-equal]
   57 |         return ((a.get_x() == b.get_x())
      |                  ~~~~~~~~~~^~~~~~~~~~~~
Point.h:58:27: warning: comparing
floating-point with ‘==’ or ‘!=’
is unsafe [-Wfloat-equal]
   58 |             && (a.get_y() == b.get_y()));
      |                 ~~~~~~~~~~^~~~~~~~~~~~
In file included from PointTest.cpp:2:
Point.h: In member function ‘bool
Point::operator==(const Point&)’:
Point.h:57:28: warning: comparing
floating-point with ‘==’ or ‘!=’
is unsafe [-Wfloat-equal]
   57 |         return ((a.get_x() == b.get_x())
      |                  ~~~~~~~~~~^~~~~~~~~~~~
Point.h:58:27: warning: comparing
floating-point with ‘==’ or ‘!=’
is unsafe [-Wfloat-equal]
   58 |             && (a.get_y() == b.get_y()));
      |                 ~~~~~~~~~~^~~~~~~~~~~~


k_gandhi6@ares:~$ ./PointTest.out
                Test For Point Class
=================================================
Enter your 1st cordinate point: (1,2)

Enter your 2nd point: (1,2)

The distance between (1, 2) and (1, 2) is: 0
The midpoint is: (1, 2)

Testing the points if equal or not equal...

Are the points equal? 1
Are the points not equal? 0
=================================================

k_gandhi6@ares:~$ ./PointTest.out
                Test For Point Class
=================================================
Enter your 1st cordinate point: (2,5)

Enter your 2nd point: (12,3)

The distance between (2, 5) and (12, 3) is: 10.198
The midpoint is: (7, 4)

Testing the points if equal or not equal...

Are the points equal? 0
Are the points not equal? 1
=================================================
```

```
k_gandhi6@ares:~$ ./PoinyointTest.out
                Test For Point Class
==================================================
Enter your 1st cordinate point: (1.4,2.4)

Enter your 2nd point: (1.6,5.3)

The distance between (1.4, 2.4) and (1.6, 5.3) is: 2.90689
The midpoint is: (1.5, 3.85)

Testing the points if equal or not equal...

Are the points equal? 0
Are the points not equal? 1
==================================================

k_gandhi6@ares:~$ cat Point.tpq
1. Which operators are members and which are non-members?
   Do any have to be members?

    - operator==, operator !=, operator/, operator- are all members in the
      class point. The non-members are operator<< and operator>>. Whenever
      we use a object we need to use members since it's neccesary.

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?

    - All of the overloaded operator should be const becasue we can't have data
      being changed since it would produce the wrong results.

3. What type do equality and and inequality return? Input? Output? Assignment?

    - The inequality and equality return a bool, Input returns a istream
      and Output returns ostream, and Assignment returns the refernce of the
      points.

4. Do you agree with your friend's decision to use operator/ for midpoint?
   Why/Why not?

    - No, this is not the best design to choose becasue we never clssify
      where the point shifted, moved, or what axis. The operator/ is to
      calculate the midpoint by dividing the points.

5. Why didn't you overload operators for less than, greater than, etc.?

    - You can't overload these operators because in some scenarios
      we could x greater than x or y less than y. Simply they can't
      be > or <.

6. Your friend wanted to overload operators for the flip and shift methods,
   too (~ and += respectively). Why did you talk them out of it? Why wasn't
   this a good idea?

    - This would defintily be bad idea becasue the '+=' is meant to be
      an assignment operator and the ~ is bitwise inverse operator

7. Just because you've added operators, should you necessarily remove
   the old methods that did these jobs?

    - No, because all you would have to do is update the methods with
      the version of the Point class.
k_gandhi6@ares:~$ exit
exit

Script done on 2021-11-24 16:32:13-06:00 [COMMAND_EXIT_CODE="0"]
```