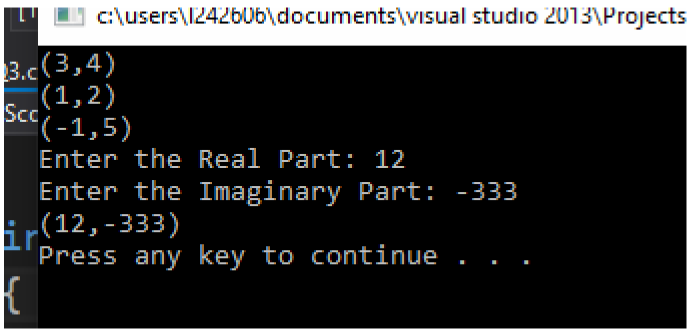
Q1:

```
#include <iostream>
using namespace std;

class ComplexNumber{
	int real;
	int imag;
public:
	ComplexNumber(int real = 0, int imag = 0)
	{
		this->real = real;
		this->imag = imag;
	}
	ComplexNumber(ComplexNumber& obj)
	{
		this->real = obj.real;
		this->imag = obj.imag;
	}

	ComplexNumber& operator +(ComplexNumber& obj)
	{
		ComplexNumber c1(this->real + obj.real, this->imag + obj.imag);
		return c1;
	}

	ComplexNumber& operator -(ComplexNumber& obj)
	{
		ComplexNumber c1(this->real - obj.real, this->imag - obj.imag);
		return c1;
	}

	ComplexNumber& operator *(ComplexNumber& obj)
	{
		ComplexNumber c1(this->real * obj.real + this->imag*obj.imag * -1, this->real * obj.imag
+ this->imag*obj.real);
		return c1;
	}

	friend ostream& operator <<(ostream& out, ComplexNumber& obj);
	friend istream& operator >>(istream& in, ComplexNumber& obj);
};

ostream& operator <<(ostream& out, ComplexNumber& obj)
{
	out << "(" << obj.real << "," << obj.imag << ")";
	return out;
}
```

```
istream& operator >>(istream& in, ComplexNumber& obj)
{
	cout << "Enter the Real Part: ";
	in >> obj.real;
	cout << "Enter the Imaginary Part: ";
	in >> obj.imag;

	return in;
}


int main()
{
	ComplexNumber c1(2, 3);
	ComplexNumber c2(1, 1);
	ComplexNumber c3;

	c3 = c1 + c2;
	cout << c3 << endl;

	c3 = c1 - c2;
	cout << c3 << endl;

	c3 = c1 * c2;
	cout << c3 << endl;

	ComplexNumber c4;
	cin >> c4;
	cout << c4 << endl;

	system("pause");
	return 0;
}
```



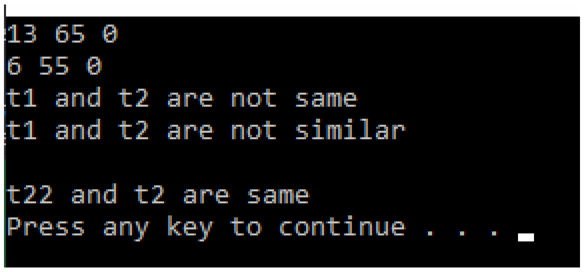
c:\users\l242606\documents\visual studio 2013\Projects

```
(3,4)
(1,2)
(-1,5)
Enter the Real Part: 12
Enter the Imaginary Part: -333
(12,-333)
Press any key to continue . . .
```

Q2:

```
#include <iostream>
using namespace std;


class Time{
        int hours;
        int mins;
        int sec;
public:
        Time(int hours = 0, int mins = 0, int sec = 0)
        {
                this->hours = hours;
                this->mins = mins;
                this->sec = sec;
        }

        Time operator +(Time& obj)
        {
                int s=0, m=0, h=0;
                s = this->sec + obj.sec;
                while (s >= 60)
                {
                        m++;
                        s = s - 60;
                }
                m = m + this->mins + obj.mins;
                while (mins >= 60)
                {
                        h++;
                        mins = mins - 60;
                }
                h = h + this->hours + obj.hours;

                Time t1(h, m, s);
                return t1;
        }

        Time operator -(Time& obj)
        {
                int s = 0, m = 0, h = 0;

                if (this->sec - obj.sec < 0)
                {
                        m--;
                        s = 60 + this->sec - obj.sec;
                }
```

```
		else
			s = s + this->sec - obj.sec;

		if (m + this->mins - obj.mins < 0)
		{
			h--;
			m = 60 + this->mins - obj.mins;
		}
		else
			m = m + this->mins - obj.mins;

		if (h + this->hours - obj.hours < 0)
		{
			cout << "Subtraction not possible (Time can not be negative)\n";
			return 0;
		}
		else
			h = h + this->hours - obj.hours;

		Time t1(h, m, s);
		return t1;
	}

	void operator ++()
	{
		this->sec++;
		if (this->sec >= 60)
		{
			this->sec - 60;
			this->mins++;
		}
		if (this->mins >= 60)
		{
			this->mins - 60;
			this->hours++;
		}
	}
	void operator --()
	{
		if (this->hours == 0 && this->mins == 0 && this->sec == 0)
		{
			cout << "can not be decremented further";
		}
		else{
			this->sec--;
			if (this->sec < 0)
			{
				this->mins--;
				this->sec + 60;
				}
				if (this->mins < 0)
				{
					this->hours--;
					this->mins + 60;
				}
			}
		}

		bool operator ==(Time& obj)
		{
			if (obj.hours == hours && obj.mins == mins && obj.sec==sec)
				return true;
			return false;
		}
		bool operator !=(Time& obj)
		{
			if (obj.hours == hours && obj.mins == mins && obj.sec&&sec)
				return false;
			return true;
		}

		void print()
		{
			cout << hours << " " << mins << " " << sec << endl;
		}
};


int main()
{
	Time t1(10, 30, 00);
	Time t2(3, 35, 00);
	Time t22(3, 35, 00);
	Time t3;
	t3 = t1 + t2;
	t3.print();
	t3 = t1 - t2;
	t3.print();
	if (t1 == t2)
		cout << "t1 and t2 are same\n";
	else
		cout << "t1 and t2 are not same\n";
	if (t1!=t2)
		cout << "t1 and t2 are not similar\n";
	else
		cout << "t1 and t2 are similar\n";

	cout << endl;
	if (t22 == t2)
		cout << "t22 and t2 are same\n";
	else
		cout << "t22 and t2 are not same\n";

	system("pause");
	return 0;
}
```

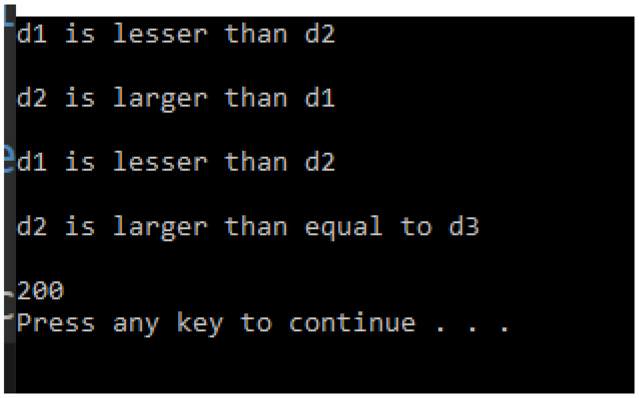


```
13 65 0
6 55 0
t1 and t2 are not same
t1 and t2 are not similar

t22 and t2 are same
Press any key to continue . . .
```

Q3:

```
#include <iostream>
using namespace std;

class Distance{
	int dist;
public:
	Distance(int dist = 0)
	{
		this->dist = dist;
	}

	bool operator <(Distance& obj)
	{
		if (this->dist < obj.dist)
			return true;
		return false;
	}

	bool operator >(Distance& obj)
	{
		if (this->dist > obj.dist)
			return true;
		return false;
	}

	bool operator <=(Distance& obj)
	{
		if (this->dist <= obj.dist)
			return true;
		return false;
```

```
        }

        bool operator >=(Distance& obj)
        {
                if (this->dist >= obj.dist)
                        return true;
                return false;
        }

        void operator +=(int m)
        {
                this->dist = this->dist + m;
        }

        void operator -=(int m)
        {
                this->dist = this->dist - m;
        }

        void print()
        {
                cout << this->dist << endl;
        }
};

int main()
{
        Distance d1(100);
        Distance d2(200);
        Distance d3(100);

        if (d1 > d2)
                cout << "d1 is greater than d2\n";
        else
                cout << "d1 is lesser than d2\n";
        cout << endl;

        if (d1 < d2)
                cout << "d2 is larger than d1\n";
        else
                cout << "d2 is smaller than d1\n";
        cout << endl;

        if (d1 >= d2)
                cout << "d1 is greater than equal to d2\n";
        else
                cout << "d1 is lesser than d2\n";
        cout << endl;
        if (d3 <= d2)
                cout << "d2 is larger than equal to d3\n";
        else
                cout << "d2 is smaller than d3\n";
        cout << endl;

        d3 += 100;
        d3.print();

        system("pause");
        return 0;
}
```

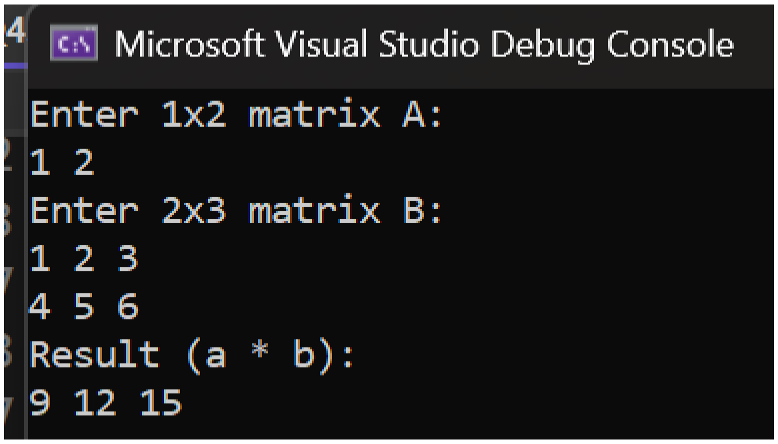


```
d1 is lesser than d2

d2 is larger than d1

d1 is lesser than d2

d2 is larger than equal to d3

200
Press any key to continue . . .
```

Q4:

```
#include <iostream>
using namespace std;

class Matrix {
    int rows;
    int cols;
    int** data;

public:
    Matrix(int r = 0, int c = 0) : rows(r), cols(c)
    {
        data = new int* [rows];
        for (int i = 0; i < rows; i++)
            data[i] = new int[cols]();
    }

    Matrix(const Matrix& other) : rows(other.rows), cols(other.cols)
    {
        data = new int* [rows];
        for (int i = 0; i < rows; i++)
        {
            data[i] = new int[cols];
            for (int j = 0; j < cols; j++)
                data[i][j] = other.data[i][j];
        }
    }

    Matrix& operator=(const Matrix& other)
    {
        if (this == &other)
            return *this;

        for (int i = 0; i < rows; i++)
            delete[] data[i];
        delete[] data;

        rows = other.rows;
        cols = other.cols;
        data = new int* [rows];
        for (int i = 0; i < rows; i++)
        {
            data[i] = new int[cols];
            for (int j = 0; j < cols; j++)
                data[i][j] = other.data[i][j];
        }
        return *this;
    }

    ~Matrix()
    {
        for (int i = 0; i < rows; i++)
            delete[] data[i];
        delete[] data;
    }

    void InputMatrix()
    {
        for (int i = 0; i < rows; i++)
            for (int j = 0; j < cols; j++)
                cin >> data[i][j];
    }

    Matrix operator*(Matrix& obj)
    {
        if (cols != obj.rows)
        {
            cout << "Multiplication not possible\n";
            return Matrix(0, 0);
        }
        Matrix result(rows, obj.cols);
        for (int i = 0; i < rows; i++)
            for (int j = 0; j < obj.cols; j++)
```

```
            for (int k = 0; k < cols; k++)
                result.data[i][j] += data[i][k] * obj.data[k][j];
        return result;
    }

    void DisplayMatrix()
    {
        for (int i = 0; i < rows; i++)
        {
            for (int j = 0; j < cols; j++)
                cout << data[i][j] << " ";
            cout << endl;
        }
    }
};

int main() {
    Matrix a(1, 2);
    cout << "Enter 1x2 matrix A:\n";
    a.InputMatrix();

    Matrix b(2, 3);
    cout << "Enter 2x3 matrix B:\n";
    b.InputMatrix();

    Matrix c = a * b;
    cout << "Result (a * b):\n";
    c.DisplayMatrix();

    return 0;
}
```


Microsoft Visual Studio Debug Console

```
Enter 1x2 matrix A:
1 2
Enter 2x3 matrix B:
1 2 3
4 5 6
Result (a * b):
9 12 15
```