#include <iostream>

using namespace std;

class complex

{

    int real;

    int img;

public:

    void *setreal*(int *r*)

    {

        this->real = r;

    }

    void *setimg*(int *i*)

    {

        this->img = i;

    }

    int *getreal*()

    {

        return this->real;

    }

    int *getimg*()

    {

        return this->img;

    }

*complex*()

    {

        this->real = 0;

        this->img = 0;

    }

*complex*(int *a*, int *b*)

    {

        this->real = a;

        this->img = b;

    }

    complex operator+(int *a*)

    {

        complex temp;

        temp.real = this->real + a;

        temp.img = this->img + a;

        return temp;

    }

    complex operator+(complex *c2*)

    {

        complex temp;

        temp.real = this->real + c2.real;

        temp.img = this->img + c2.img;

        return temp;

    }

    complex operator-(int *a*)

    {

        complex temp;

        temp.real = this->real - a;

        temp.img = this->img - a;

        return temp;

    }

    complex operator-(complex *c2*)

    {

        complex temp;

        temp.real = this->real - c2.real;

        temp.img = this->img - c2.img;

        return temp;

    }

    complex operator\*(int *a*)

    {

        complex temp;

        temp.real = this->real \* a;

        temp.img = this->img \* a;

        return temp;

    }

    complex operator\*(complex *c2*)

    {

        complex temp;

        temp.real = this->real \* c2.real;

        temp.img = this->img \* c2.img;

        return temp;

    }

    complex operator/(int *a*)

    {

        complex temp;

        temp.real = this->real / a;

        temp.img = this->img / a;

        return temp;

    }

    complex operator/(complex *c2*)

    {

        complex temp;

        temp.real = this->real / c2.real;

        temp.img = this->img / c2.img;

        return temp;

    }

    void *display*()

    {

        cout << "\n " << real << " / " << img << "i";

    }

    void *displayS*()

    {

        cout << "\n " << real << " - " << img << "i";

    }

    void *displayA*()

    {

        cout << "\n " << real << " + " << img << "i";

    }

    void *displayM*()

    {

        cout << "\n " << real << " \* " << img << "i";

    }

};

complex operator+(int, complex);

complex operator-(int, complex);

complex operator\*(int, complex);

complex operator/(int, complex);

int *main*()

{

    complex *c1*(102, 220);

    complex *c2*(54, 7);

    complex c3;

    complex c4, c5;

    cout << "\n \n Add";

    c3 = c1 + c3;

    c4 = c1 + 10;

    c5 = 10 + c1;

    c1.*displayA*();

    c2.*displayA*();

    c3.*displayA*();

    c4.*displayA*();

    c5.*displayA*();

    cout << "\n \n sub";

    c3 = c1 - c3;

    c4 = c1 - 10;

    c5 = 10 - c1;

    c1.*displayS*();

    c2.*displayS*();

    c3.*displayS*();

    c4.*displayS*();

    c5.*displayS*();

    cout << "\n \n multi";

    c3 = c1 \* c3;

    c4 = c1 \* 10;

    c5 = 10 \* c1;

    c1.*displayM*();

    c2.*displayM*();

    c3.*displayM*();

    c4.*displayM*();

    c5.*displayM*();

    cout << "\n \n Divi";

    c3 = c1 / c2;

    c4 = c1 / 10;

    c5 = 250 / c1;

    c1.*display*();

    c2.*display*();

    c3.*display*();

    c4.*display*();

    c5.*display*();

}

complex operator+(int *a*, complex *c1*)

{

    complex temp;

    int i, r;

    r = a + c1.*getreal*();

    i = a + c1.*getimg*();

    temp.*setreal*(r);

    temp.*setimg*(i);

    return temp;

}

complex operator-(int *a*, complex *c1*)

{

    complex temp;

    int i, r;

    r = a - c1.*getreal*();

    i = a - c1.*getimg*();

    temp.*setreal*(r);

    temp.*setimg*(i);

    return temp;

}

complex operator\*(int *a*, complex *c1*)

{

    complex temp;

    int i, r;

    r = a \* c1.*getreal*();

    i = a \* c1.*getimg*();

    temp.*setreal*(r);

    temp.*setimg*(i);

    return temp;

}

complex operator/(int *a*, complex *c1*)

{

    complex temp;

    int i, r;

    r = a / c1.*getreal*();

    i = a / c1.*getimg*();

    temp.*setreal*(r);

    temp.*setimg*(i);

    return temp;

}

Output:

