```cpp
// W8_1_2.cpp : Defines the entry point for the console application.
//              Przeciazenie operatora + (binarnego) na podstawie operator-funkcji zaprzyjaznionej
//              Przeciazenie operatora ++ (unarnego) na podstawie operator-funkcji zaprzyjaznionej

#include "stdafx.h"
#include <iostream>
using namespace std;

class coord
{
    double x;
    double y;
public:
    coord(double xx, double yy) { x = xx; y = yy; }
    coord() { x = 0.0;  y = 0.0; }
    void set(double xx, double yy) { x = xx; y = yy; }
    coord get() { return *this; }
    void disp(char *tit) { cout << tit << ": " << x << "  " << y << endl; }
    friend coord operator + (double lew, coord &praw);
    friend coord operator + (coord &lew, double praw);

    friend coord operator ++ (coord &ob);   //prefiksowa forma
    friend coord operator ++ (coord &ob, int notused);   //postfiksowa forma

};

coord operator + (double lew, coord &praw)
{
    coord res;
    res.x = lew+praw.x;
    res.y = lew+praw.y;
    return res;
}

coord operator + (coord &lew, double praw)
{
    coord res;
    res.x = lew.x+praw;
    res.y = lew.y+praw;
    return res;
}

coord operator ++ (coord &ob)
/* prefiksowa forma: ++ob*/
{
    ++ob.x;
    ++ob.y;
    return ob;
}

coord operator ++ (coord &ob, int notused)
/* postfiksowa forma: ob++*/
{
    coord prev = ob;
    ++ob.x;
    ++ob.y;
    return prev;
}

int _tmain(int argc, _TCHAR* argv[])
{
    coord ob1(1, 2), res;
    ob1.disp("ob1");

    res = 10.0+ob1;
    res.disp("res = 10.0+ob1");

    res.set(0, 0);
    res = ob1+10.0;
    res.disp("res = ob1+10.0");

    ob1.set(1, 2);
```

```
    res.set(0, 0);
    ob1.disp("ob1");
    res = ++ob1;
    res.disp("res = ++ob1");
    ob1.disp("++ob1");

    ob1.set(1, 2);
    res.set(0, 0);
    ob1.disp("ob1");
    res = ob1++;
    res.disp("res = ob1++");
    ob1.disp("ob1++");

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