```
#include <iostream>
#include <iomanip>
#include <assert.h>
using namespace std;

const int U1_UPB = 100; //Upper bound of U1
const int U1_LOB = -100;//Lower bound of U1
const int prt_UPB = 2;  //Upper bound of prt
const int prt_LOB = -1; //Lower bound of prt

class Qnum_t {
    public:
        Qnum_t(int _U1): U1(_U1), prt(0){
            assert(U1 < U1_UPB && U1 > U1_LOB);
            //cout<<"Constructing Qnum " << this << endl;
        }
        Qnum_t(int _U1, int _prt): U1(_U1), prt(_prt){
            assert(U1 < U1_UPB && U1 > U1_LOB && prt < prt_UPB && prt > prt_LOB);
            //cout<<"Constructing Qnum " << this << endl;
        }

        Qnum_t(const Qnum_t& _q):U1(_q.U1), prt(_q.prt){
            //cout<<"Copying Qnum " << this << " from " << &_q << endl;
        }
        ~Qnum_t(){
            //cout<<"Destructing Qnum " << this<< endl;
        };

        void set(int _U1 = 0, int _prt = 0);
        friend bool operator< (const Qnum_t& q1, const Qnum_t& q2);
        friend bool operator<= (const Qnum_t& q1, const Qnum_t& q2);
        friend bool operator== (const Qnum_t& q1, const Qnum_t& q2);
        friend Qnum_t operator- (const Qnum_t& q1);
        friend Qnum_t operator* (const Qnum_t& q1, const Qnum_t& q2);
        friend ostream& operator<< (ostream& os, const Qnum_t& q);
    private:
        int U1;
        int prt;

};
```