#include<iostream>

#include<vector>

#include<functional>

using namespace std;

const char\* PORUKA = "\n-------------------------------------------------------------------------------\n"

"1. SVE KLASE TREBAJU POSJEDOVATI ADEKVATAN DESTRUKTOR\n"

"2. NAMJERNO IZOSTAVLJANJE KOMPLETNIH I/ILI POJEDINIH DIJELOVA DESTRUKTORA CE BITI OZNACENO KAO RE\n"

"3. SPASAVAJTE PROJEKAT KAKO BI SE SPRIJECILO GUBLJENJE URADJENOG ZADATKA\n"

"4. NAZIVI FUNKCIJA, TE BROJ I TIP PARAMETARA MORAJU BITI IDENTICNI ONIMA KOJI SU KORISTENI U TESTNOM CODE-U,\n"

"\tOSIM U SLUCAJU DA POSTOJI ADEKVATAN RAZLOG ZA NJIHOVU MODIFIKACIJU. OSTALE\n"

"\tPOMOCNE FUNKCIJE MOZETE IMENOVATI I DODAVATI PO ZELJI.\n"

"5. RJESENJA ZADATKA POSTAVITE NA FTP SERVER U ODGOVARAJUCI FOLDER!\n"

"-------------------------------------------------------------------------------\n";

const char\* crt = "\n-------------------------------------------\n";

class Datum {

    int\* \_dan, \* \_mjesec, \* \_godina;

public:

    Datum(int dan = 1, int mjesec = 1, int godina = 2000);

    ~Datum() {

    }

    friend ostream& operator<< (ostream& COUT, const Datum& obj) {

        COUT << \*obj.\_dan << "." << \*obj.\_mjesec << "." << \*obj.\_godina;

        return COUT;

    }

};

char\* GetNizKaraktera(const char\* sadrzaj, bool dealociraj = false);

template<class T1, class T2>

class Kolekcija {

    T1\* \_elementi1;

    T2\* \_elementi2;

    int \_trenutno;

    bool \_omoguciDupliranje;

public:

    Kolekcija(bool omoguciDupliranje = true) {

        \_elementi1 = nullptr;

        \_elementi2 = nullptr;

    }

    T1\* GetElementi1() const { return \_elementi1; }

    T2\* GetElementi2() const { return \_elementi2; }

    ~Kolekcija() {

    }

    T1& getElement1(int lokacija)const { return \_elementi1[lokacija]; }

    T2& getElement2(int lokacija)const { return \_elementi2[lokacija]; }

    int getTrenutno() { return \_trenutno; }

    bool DupliranjeOmoguceno() { return \_omoguciDupliranje; }

    friend ostream& operator<< (ostream& COUT, const Kolekcija& obj) {

        for (size\_t i = 0; i < obj.\_trenutno; i++)

            COUT << obj.getElement1(i) << " " << obj.getElement2(i) << endl;

        return COUT;

    }

};

class Predmet {

    const char\* \_naziv;

    char\* \_detalji;

    char \_akademskaGodina[100];

    unique\_ptr<int> \_brojECTS;

    Kolekcija<int, Datum>\* \_ocjene;

public:

    Predmet(const char\* naziv = "", const char\* detalji = "", const char\* akademskaGodina = "", int brojECTS = 5);

    ~Predmet() {

        delete[] \_naziv;

        \_naziv = nullptr;

    }

    void AddOcjena(int ocjena, Datum datum) {

        \_ocjene->AddElement(ocjena, datum);

    }

    const char\* GetNaziv() { return \_naziv; }

    Kolekcija<int, Datum>& GetOcjene() { return \*\_ocjene; }

    friend ostream& operator <<(ostream& COUT, const Predmet& predmet) {

        COUT << "Naziv: " << predmet.\_naziv << endl;

        COUT << "Detalji: " << predmet.\_detalji << endl;

        COUT << "Akademska godina: " << predmet.\_akademskaGodina << endl;

        COUT << "Broj ECTS:" << \*predmet.\_brojECTS << endl;

        COUT << "Ocjene: " << endl;

        COUT << \*predmet.\_ocjene << endl;

        return COUT;

    }

};

//formula: (x1 + x2 + x3 + ... + xn)/n;

function<float(int[], int)> aritmetickaSredina;

//formula:  (x1 \* x2 \* x3 \* ... \* xn)^(1/n);

function<float(int[], int)> geometrijskaSredina;

//formula:  n/((1/x1) + (1/x2) + ... + (1/xn))

function<float(int[], int)> harmonijskaSredina;

int GetRandomBroj(int a, int b) {

    return rand() % b + a;

}

void main() {

    cout << PORUKA;

    Datum datum1(1, 1, 2021), datum2(2, 2, 2021), datum3(3, 3, 2021), datum4(4, 4, 2021);

    Datum temp(datum1);

    cout << ((datum1 == temp) ? "Objekti sa istim vrijednostima" : "Objekti sa razlicitim vrijednostima") << endl;

    Datum temp2(move(temp));

    cout << ((datum1 == temp2) ? "Objekti sa istim vrijednostima" : "Objekti sa razlicitim vrijednostima") << endl;

    Datum temp3;

    temp3 = datum1;

    cout << ((datum1 == temp3) ? "Objekti sa istim vrijednostima" : "Objekti sa razlicitim vrijednostima") << endl;

    cout << "\n";

    Kolekcija<Datum, int> ocjene(false);

    ocjene.AddElement(datum1, 10);

    ocjene.AddElement(datum1, 10);

    ocjene.AddElement(datum1, 8);

    ocjene.AddElement(datum2, 6);

    ocjene.AddElement(datum3, 6);

    ocjene.AddElement(datum4, 7);

    Kolekcija<Datum, int> ocjene2(ocjene);

    cout << ((ocjene == ocjene2) ? "Kolekcije sa istim vrijednostima" : "Kolekcije sa razlicitim vrijednostima") << endl;

    Kolekcija<Datum, int> ocjene3(move(ocjene2));

    cout << ((ocjene == ocjene3) ? "Kolekcije sa istim vrijednostima" : "Kolekcije sa razlicitim vrijednostima") << endl;

    Kolekcija<Datum, int> ocjene4;

    ocjene4 = ocjene;

    cout << ((ocjene == ocjene4) ? "Kolekcije sa istim vrijednostima" : "Kolekcije sa razlicitim vrijednostima") << endl;

    cout << ocjene4 << endl;

    int kolekcijaTestSize = 9;

    Kolekcija<int, int> brojevi(false);

    for (int i = 0; i <= kolekcijaTestSize; i++) {

        int broj1 = GetRandomBroj(1, 3);

        int broj2 = GetRandomBroj(1, 3);

        brojevi.AddElement(broj1, broj2);

    }

    cout << brojevi << endl;

    ///\*objekat 'brojevi2' ce biti inicijalizovan elementima koji se u objektu 'brojevi' nalaze na lokacijama 1 - 4

    //ukljucujuci i te lokacije.

    Kolekcija<int, int> brojevi2 = brojevi(1, 4);

    cout << brojevi2 << crt;

    Predmet operativniSistemi("Operativni sistemi", "Neki detalji...", "2020/2021", 7),

        diskretnaMatematika("Diskretna matematika", "Neki detalji...", "2020/2021", 6),

        engleskiJezik1("Engleski Jezik I", "Neki detalji...", "2020/2021", 6),

        programiranje1("Programiranje I", "Neki detalji...", "2020/2021", 7);

    programiranje1.AddOcjena(7, Datum(5, 4, 2021));

    programiranje1.AddOcjena(8, Datum(11, 4, 2021));

    programiranje1.AddOcjena(10, Datum(26, 4, 2021));

    Predmet pr1(programiranje1);

    cout << ((pr1 == programiranje1) ? "Objekti sa istim vrijednostima" : "Objekti sa razlicitim vrijednostima") << endl;

    Predmet pr\_1(move(pr1));

    cout << ((pr\_1 == programiranje1) ? "Objekti sa istim vrijednostima" : "Objekti sa razlicitim vrijednostima") << endl;

    Predmet PR1;

    PR1 = programiranje1;

    cout << ((PR1 == programiranje1) ? "Objekti sa istim vrijednostima" : "Objekti sa razlicitim vrijednostima") << endl;

    cout << PR1 << endl;

    cout << aritmetickaSredina(PR1.GetOcjene().GetElementi1(), PR1.GetOcjene().getTrenutno()) << endl;

    cout << geometrijskaSredina(PR1.GetOcjene().GetElementi1(), PR1.GetOcjene().getTrenutno()) << endl;

    cout << harmonijskaSredina(PR1.GetOcjene().GetElementi1(), PR1.GetOcjene().getTrenutno()) << endl;

    system("pause>0");

}