#include <iostream>

#include <memory>

#include <functional>

using namespace std;

//Z0.1

char\* AlocirajIKopiraj(const char\* izvor);

//Z0.2 :: Funkcija vraca max. od dva elementa

template<typename T>

T Max(T el1, T el2);

//Z0.3 :: Funkcija vraca min. od dva elementa

template<typename T>

T Min(T el1, T el2);

//Z0.4 :: Ispis niza (tipa T) gdje se delimiter ispisuje izmedju dva susjedna elementa tog niza

template<class T>

void IspisNiza(T niz[], int vel, const char\* delimiter = ", ");

/\*Z0.5 :: Napisati genericku funkciju 'BubbleSort' za sortiranje niza elemenata tipa T

       :: Iskoristiti vec postojeci kod za sortiranje niza integera ('BubbleSort.txt') i prepraviti ga

       :: reverse' polje oznacava da li se radi o uzlaznom ili silaznom sortiranju

\*/

template<class T>

void BubbleSort(T niz[], int vel, bool reverse = false);

class Datum

{

private:

    unique\_ptr<int> \_dan;

    unique\_ptr<int> \_mjesec;

    unique\_ptr<int> \_godina;

public:

    //Z1.1

    Datum();

    //Z1.2

    Datum(int dan, int mjesec, int godina);

    //Z1.3

    Datum(const Datum& obj);

    //Z1.4

    Datum(Datum&& obj);

    //Z1.5

    Datum& operator =(const Datum& obj);

    //Z1.6

    void SetDan(int dan);

    void SetMjesec(int mjesec);

    void SetGodina(int godina);

    //Z1.7

    int GetDan() const;

    int GetMjesec() const;

    int GetGodina() const;

    //Z1.8

    bool CheckIfValid();

    //Z1.9 :: Staticka metoda za generisanje datuma (sa ugradjenom provjerom validnosti tog datuma)

    static Datum GetRandomDatum();

    //Z1.10

    ~Datum();

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

    friend bool operator == (const Datum& d1, const Datum& d2);

    friend bool operator > (const Datum&, const Datum&);

};

//Z1.11

bool operator == (const Datum& d1, const Datum& d2);

bool operator != (const Datum& d1, const Datum& d2);

//Z1.12

bool operator >(const Datum& d1, const Datum& d2);

bool operator >= (const Datum& d1, const Datum& d2);

bool operator <= (const Datum& d1, const Datum& d2);

bool operator < (const Datum& d1, const Datum& d2);

//Z1.13

ostream& operator << (ostream& COUT, const Datum& obj);

//Vektor :: Kolekcija elemenata (tipa T) sa mogucnoscu dinamickog prosirivanja

template<class T>

class Vektor {

private:

    int \_trenutno;  //oznacava velicinu niza

    int \_maxElemenata;

    T\* \_elementi;

public:

    //Z2.1

    Vektor(int maxElemenata = 10);

    //Z2.2

    Vektor(const Vektor<T>& kolekcija);

    //Z2.3

    Vektor(Vektor<T>&& kolekcija);

    //Z2.4

    Vektor<T>& operator = (const Vektor<T>& kolekcija);

    //Z2.5 :: Osigurati da je index unutar opsega [0,\_trenutno-1]. Koristiti genericke funkcije 'Min' i 'Max'

    T& operator [] (int index) const;

    T\* GetElementi() const;

    //Z2.6

    void ProsiriVektor(int prosiriZa);

    //Z2.7 :: Dodavanje novog elementa (ukoliko je \_trenutno < \_maxElemenata, uraditi prosirivanje niza za 10)

    void operator += (T noviElement);

    //Z2.8

    bool UkloniZadnjiElement();

    //Z2.9

    bool CheckIfSadrzi(T element);

    //Z2.10 :: Sortiranje elemenata na osnovu specificiranog algoritma

    void Sortiraj(void(\*algoritamSortiranja)(T[], int, bool), bool reverse = false);

    //Z2.11

    int GetTrenutno() const;

    int GetMaxElemenata() const;

    //Z.12

    ~Vektor();

    friend ostream& operator << <> (ostream& COUT, const Vektor<T>& kolekcija);

};

//Z2.13 :: Ispisuje elemente kolekcije. Za pristup elementima koristiti preklopljeni operator []

template<class T>

ostream& operator << <>(ostream& COUT, const Vektor<T>& kolekcija);

class Fudbaler {

private:

    char\* \_ime;

    char\* \_prezime;

    unique\_ptr<Datum> \_datumRodjenja;

    float\* \_visina; // npr. 1.75 m

    char \_pozicija[50]; //pozicija na terenu

    int \_trenutnoNagrada; //brojac :: krece od 0

    char\* \_nagrade[100] = { nullptr };

public:

    //Z3.1

    Fudbaler();

    //Z3.2

    Fudbaler(const char\* ime, const char\* prezime, Datum datum, float visina, const char\* pozicija);

    //Z3.3

    Fudbaler(const Fudbaler& obj);

    //Z3.4

    Fudbaler(Fudbaler&& obj);

    //Z3.5

    Fudbaler& operator = (const Fudbaler& obj);

    //Z3.6 :: Dodavanje nove nagrade u niz pokazivaca '\_nagrade'. Uraditi alokaciju prilikom dodavanja

    bool operator()(const char\* nagrada);

    //Z3.7

    const char\* GetIme() const;

    const char\* GetPrezime() const;

    Datum GetDatumRodjenja() const;

    float GetVisina() const;

    const char\* GetPozicija() const;

    int GetTrenutnoNagrada() const;

    //Z3.8 :: Dohvacanje jednog elementa iz niza '\_nagrade'

    const char\* operator[] (int index) const;

    //Z3.9

    void SetIme(const char\* ime);

    void SetPrezime(const char\* prezime);

    void SetDatumRodjenja(Datum datumRodjenja);

    void SetVisina(float visina);

    void SetPozicija(const char\* pozicija);

    //Z3.10

    ~Fudbaler();

};

//Z3.11

ostream& operator <<(ostream& COUT, const Fudbaler& fudbaler);

//Z3.12 :: Poredjenje vrijedosti svih atrbitura jednog objekta sa vrijednostima atributa drugog objekta

bool operator == (const Fudbaler& f1, const Fudbaler& f2);

bool operator != (const Fudbaler& f1, const Fudbaler& f2);

auto GetRandomBrojIgraca = []() {

    return rand() % 100 + 1;

};

class FudbalskiTim {

private:

    const char\* \_naziv;

    Datum \_datumOsnivanja;

    char\* \_domaciStadion;

    Vektor<Fudbaler> \_fudbaleri;

    Vektor<int> \_brojevi; // brojevi fudbalera na majicama

    function<int()> \_rasporedjivac;

public:

    //Z4.1 :: Postaviti defaultne vrijednost

    FudbalskiTim(function<int()> rasporedjivac = GetRandomBrojIgraca);

    //Z4.2

    FudbalskiTim(const char\* naziv, Datum datumOsnivanja, const char\* domaciStadion, function<int()> rasporedjivac = GetRandomBrojIgraca);

    //Z4.3

    FudbalskiTim(const FudbalskiTim& obj);

    //Z4.4

    FudbalskiTim(FudbalskiTim&& obj);

    //Z4.5

    FudbalskiTim& operator = (const FudbalskiTim& obj);

    //Z4.5

    const char\* GetNaziv() const;

    Datum GetDatumOsnivanja() const;

    const char\* GetDomaciStadion() const;

    function<int()> GetRasporedjivac() const;

    Vektor<Fudbaler> GetFudbalere() const;

    Vektor<int> GetBrojeve() const;

    //Z4.6

    void SetNaziv(const char\* naziv);

    void SetDatumOsnivanja(Datum datumOsnivanja);

    void SetDomaciStadion(const char\* stadion);

    /\*Z4.7 :: Dodati novog fudbalera, i asocirati uz njega novi (neiskoristeni) broj koristenjem pokazivaca na funkciju '\_rasporedjivac'

           :: Dodavanje vrsiti u vektore '\_fudbaleri' i '\_brojevi'\*/

    void operator += (Fudbaler fudbaler);

    //4.8

    Fudbaler& operator[](int index) const;

    //4.9 :: Setter za polje '\_brojevi'. Promijeniti vrijednost elementa na lokaciji specificiranoj parametrom 'index'

    void SetBroj(int index, int noviBroj);

    //Z4.10

    ~FudbalskiTim();

    friend ostream& operator << (ostream& COUT, const FudbalskiTim& tim);

    friend bool operator == (const FudbalskiTim& tim1, const FudbalskiTim& tim2);

};

//Z4.11 :: Ispisati obiljezja '\_naziv', '\_datumOsnivanja', '\_domaciStadion', te imena, prezimena i brojeve fudbalera, kao i pozicije na kojima igraju

ostream& operator << (ostream& COUT, const FudbalskiTim& tim);

//Z4.12 :: Poredjenje objekata (poredjenje svih atributa)

bool operator == (const FudbalskiTim& tim1, const FudbalskiTim& tim2);

bool operator != (const FudbalskiTim& tim1, const FudbalskiTim& tim2);

class Utakmica {

private:

    pair<FudbalskiTim, FudbalskiTim> \_timovi;

    pair<int, int> \_postignutiGolovi;

public:

    //Z5.1 '\_postignutiGolovi' postaviti na vrijednosti <0,0>

    Utakmica();

    //Z5.2

    Utakmica(FudbalskiTim domacin, FudbalskiTim gosti);

    //Z5.3

    Utakmica(const Utakmica& obj);

    //Z5.4

    Utakmica(Utakmica&& obj);

    //Z5.5

    Utakmica& operator = (const Utakmica& obj);

    //Z5.6

    FudbalskiTim GetDomacin() const;

    FudbalskiTim GetGosti() const;

    //Z5.7

    void SetDomacin(FudbalskiTim tim);

    void SetGosti(FudbalskiTim tim);

    //Z5.8 :: int golZa = {1 if Domacin, 2 if Gosti}

    void DodajGol(int golZa);

    //Z5.9

    ~Utakmica() {

    }

    friend ostream& operator <<(ostream& COUT, const Utakmica& utakmica);

    friend bool operator == (const Utakmica& u1, const Utakmica& u2);

};

/\*Z5.10 :: Ispisati utakmicu u sljedecem formatu:

        Stadion: Camp Nou

        Barcelona - Arsenal [2:2]

\*/

ostream& operator <<(ostream& COUT, const Utakmica& utakmica);

//Z5.11

bool operator == (const Utakmica& u1, const Utakmica& u2);

void Zadatak1() {

    Datum aprilFoolsDay;

    Datum laborDay(1, 5, 2021);

    Datum newYearsEve(31, 12, 2021);

    aprilFoolsDay.SetDan(1);

    aprilFoolsDay.SetMjesec(4);

    aprilFoolsDay.SetGodina(2021);

    Datum d1(aprilFoolsDay);

    Datum d2(move(laborDay));

    Datum d3;

    d3 = newYearsEve;

    cout << d1 << endl;

    cout << d2 << endl;

    cout << d3 << endl;

    cout << "Dealokacija..." << endl;

}

void Zadatak2() {

    cout << ":::::::::: Vektor<int> :::::::::::\n";

    Vektor<int> ocjeneFilma;

    int brojOcjena = 30;

    for (int i = 0; i < brojOcjena; i++)

        ocjeneFilma += rand() % 10 + 1;

    IspisNiza(ocjeneFilma.GetElementi(), ocjeneFilma.GetTrenutno(), ", ");

    ocjeneFilma.Sortiraj(BubbleSort);

    cout << "Ocjene filma nakon sortiranja [Ascending]:" << endl;

    IspisNiza(ocjeneFilma.GetElementi(), ocjeneFilma.GetTrenutno(), ", ");

    ocjeneFilma.Sortiraj(BubbleSort, true);

    cout << "Ocjene filma nakon sortiranja [Descending]:" << endl;

    IspisNiza(ocjeneFilma.GetElementi(), ocjeneFilma.GetTrenutno(), ", ");

    Vektor<int> ocjeneFilma2(ocjeneFilma);

    Vektor<int> ocjeneFilma3(move(ocjeneFilma));

    Vektor<int> ocjeneFilma4;

    ocjeneFilma4 = ocjeneFilma2;

    auto checkIfIsti = [](Vektor<int>& v1, Vektor<int>& v2) {

        if (v1.GetTrenutno() != v2.GetTrenutno())

            return false;

        for (int i = 0; i < v1.GetTrenutno(); i++)

            if (v1[i] != v2[i])

                return false;

        return true;

    };

    cout << (checkIfIsti(ocjeneFilma, ocjeneFilma2) ? "Isti vektori" : "Nisu isti vektori") << endl;

    cout << (checkIfIsti(ocjeneFilma, ocjeneFilma3) ? "Isti vektori" : "Nisu isti vektori") << endl;

    cout << (checkIfIsti(ocjeneFilma2, ocjeneFilma3) ? "Isti vektori" : "Nisu isti vektori") << endl;

    cout << (checkIfIsti(ocjeneFilma4, ocjeneFilma2) ? "Isti vektori" : "Nisu isti vektori") << endl;

    cout << "\n\n";

    cout << ":::::::::: Vektor<Datum> :::::::::::\n";

    Vektor<Datum> datumi;

    int brojDatuma = 15;

    for (int i = 0; i < brojDatuma; i++)

        datumi += Datum::GetRandomDatum(); // dodavanje novog datuma u vektor

    cout << datumi << endl;

    datumi.Sortiraj(BubbleSort);

    cout << "Datumi nakon sortiranja [Ascending]:" << endl;

    cout << datumi << endl;

    datumi.Sortiraj(BubbleSort, true);

    cout << "Datumi nakon sortiranja [Descending]:" << endl;

    cout << datumi << endl;

    cout << "Dealokacija..." << endl;

}

void Zadatak3() {

    cout << "::::::::::klasa Fudbaler:::::::::::\n";

    Fudbaler gandhi;

    gandhi.SetIme("Mohandas");

    gandhi.SetPrezime("Gandhi");

    gandhi.SetVisina(1.70f);

    gandhi.SetDatumRodjenja(Datum(5, 5, 1990));

    gandhi.SetPozicija("Obrambeni vezni");

    //Dodjeljivanje nagrada igracu

    gandhi("Fair play award 2018"); // poziv operatora '()' za dodavanje novog elementa u niz '\_nagrade'

    gandhi("Fair play award 2019");

    gandhi("Fair play award 2020");

    gandhi("Fair play award 2021");

    gandhi("The best team leader award 2019");

    gandhi("The best team leader award 2020");

    gandhi("The greatest comeback award 2020");

    gandhi("The greatest comeback award 2021");

    gandhi("Sportsmanship award 2019");

    gandhi("Sportsmanship award 2020");

    gandhi("The most effective player vs. England national team 2020");

    gandhi("The most effective player vs. England national team 2021");

    gandhi("The 1st player to score 5 goals vs. England in a single match");

    gandhi("The best Indian football player of all time");

    cout << gandhi << "\n";

    cout << "Testiranje user-def. konstruktora: ";

    Fudbaler gandhi2(gandhi.GetIme(), gandhi.GetPrezime(), gandhi.GetDatumRodjenja(), gandhi.GetVisina(), gandhi.GetPozicija());

    for (int i = 0; i < gandhi.GetTrenutnoNagrada(); i++)

        gandhi2(gandhi[i]); // dodajemo nagrade drugom objektu

    cout << ((gandhi == gandhi2) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje konstruktora kopije: ";

    Fudbaler gandhiImpostor(gandhi);

    cout << ((gandhi == gandhiImpostor) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje move konstruktora: ";

    Fudbaler gandhiMover(move(gandhiImpostor));

    cout << ((gandhi == gandhiMover) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje operatora '=': ";

    Fudbaler martinLutherKing;

    martinLutherKing = gandhi;

    cout << ((gandhi == martinLutherKing) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Dealokacija..." << endl;

}

void Zadatak4() {

    cout << "::::::::::klasa FudbalskiTim:::::::::::\n";

    Fudbaler trump("Donald J.", "Trump", Datum(1, 3, 2150), 1.66f, "Centralni napadac");

    Fudbaler dalaiLama("Tenzin", "Gyatso", Datum(5, 3, 2151), 1.78f, "Obrambeni vezni");

    Fudbaler superSemir("Dr. Semir", "Osmanagic", Datum(5, 3, 2150), 1.85f, "Golman");

    Fudbaler ciroImpostor("Tarik", "Filipovic", Datum(5, 3, 2149), 1.85f, "Stoper");

    Fudbaler kreso("Kresimir", "Misak", Datum(5, 3, 2158), 1.66f, "Krilni vezni");

    Fudbaler theForbiddenOne("Chuck", "Norris", Datum(32, 12, 2154), 2.5f, "Na svim pozicijama istovremeno");

    FudbalskiTim osvajaciGalaksije;

    osvajaciGalaksije.SetNaziv("Osvajaci Galaksije (F.C)");

    osvajaciGalaksije.SetDomaciStadion("Milenijsko gnijezdo");

    osvajaciGalaksije.SetDatumOsnivanja(Datum(5, 5, 2100));

    osvajaciGalaksije += trump;

    osvajaciGalaksije += dalaiLama;

    osvajaciGalaksije += superSemir;

    osvajaciGalaksije += ciroImpostor;

    osvajaciGalaksije += kreso;

    osvajaciGalaksije += theForbiddenOne;

    cout << osvajaciGalaksije << endl;

    cout << "Testiranje user-def. konstruktora: ";

    FudbalskiTim galaktikosi(osvajaciGalaksije.GetNaziv(), osvajaciGalaksije.GetDatumOsnivanja(), osvajaciGalaksije.GetDomaciStadion(),

        osvajaciGalaksije.GetRasporedjivac());

    Vektor<Fudbaler> fudbaleri = osvajaciGalaksije.GetFudbalere();

    Vektor<int> brojevi = osvajaciGalaksije.GetBrojeve();

    for (int i = 0; i < fudbaleri.GetTrenutno(); i++)

        galaktikosi += fudbaleri[i];

    for (int i = 0; i < brojevi.GetTrenutno(); i++)

        galaktikosi.SetBroj(i, brojevi[i]);

    cout << ((osvajaciGalaksije == galaktikosi) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje konstruktora kopije: ";

    FudbalskiTim osvajaci2(osvajaciGalaksije);

    cout << ((osvajaciGalaksije == osvajaci2) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje move konstruktora: ";

    FudbalskiTim osvajaci3(move(osvajaci2));

    cout << ((osvajaciGalaksije == osvajaci3) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje operatora '=': ";

    FudbalskiTim osvajaci4;

    osvajaci4 = osvajaciGalaksije;

    cout << ((osvajaci4 == osvajaciGalaksije) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Dealokacija..." << endl;

}

void Zadatak5() {

    cout << "::::::::::klasa Utakmica:::::::::::\n";

    //Tim :: 1

    Fudbaler sleepyJoe("Joe", "Biden", Datum(3, 4, 1950), 1.77f, "Obrambeni vezni");

    Fudbaler stableTrump("Donald J.", "Trump", Datum(5, 4, 1951), 1.88f, "Centralni napadac");

    Fudbaler lyingTed("Ted", "Cruz", Datum(6, 6, 1952), 1.8f, "Centralni vezni");

    Fudbaler theLittleMan("Marco", "Rubio", Datum(7, 7, 1953), 1.6f, "Ofanzivni vezni");

    Fudbaler lowEnergyJeb("Jeb", "Bush", Datum(8, 8, 1954), 1.68f, "Golman");

    //Tim :: 2

    Fudbaler facebookLord("Mark", "Zuckerberg", Datum(3, 4, 1960), 1.74f, "Obrambeni vezni");

    Fudbaler elonTheMartian("Elon", "Musk", Datum(5, 5, 1961), 1.77f, "Leteci napadac");

    Fudbaler codeJeff("Jeff", "Bezos", Datum(5, 3, 1961), 1.8f, "Centralni vezni");

    Fudbaler theVaxGuy("Bill", "Gates", Datum(15, 3, 1955), 1.85f, "Golman");

    Fudbaler cryptoGuru("Satoshi", "Nakamoto", Datum(15, 3, 1955), 1.85f, "Nevidljiv na terenu");

    FudbalskiTim cowboys("Cowboys (F.C)", Datum(1, 3, 1960), "Wild West Park");

    cowboys += sleepyJoe;

    cowboys += stableTrump;

    cowboys += lyingTed;

    cowboys += theLittleMan;

    cowboys += lowEnergyJeb;

    FudbalskiTim aliens("Aliens (F.C)", Datum(5, 6, 1955), "Galaxy Centre");

    aliens += facebookLord;

    aliens += elonTheMartian;

    aliens += codeJeff;

    aliens += theVaxGuy;

    aliens += cryptoGuru;

    Utakmica spaceClassico;

    spaceClassico.SetDomacin(cowboys);

    spaceClassico.SetGosti(aliens);

    spaceClassico.DodajGol(1); //gol za domacina

    spaceClassico.DodajGol(1); //gol za domacina

    spaceClassico.DodajGol(2); //gol za goste

    spaceClassico.DodajGol(2); //gol za goste

    spaceClassico.DodajGol(1); //gol za domacina

    cout << spaceClassico << endl;

    cout << "Testiranje user-def. konstruktora: ";

    Utakmica derbi(cowboys, aliens);

    derbi.DodajGol(1);

    derbi.DodajGol(1);

    derbi.DodajGol(2);

    derbi.DodajGol(2);

    derbi.DodajGol(1);

    cout << ((spaceClassico == derbi) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje konstruktora kopije: ";

    Utakmica kopija(spaceClassico);

    cout << ((spaceClassico == kopija) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje move konstruktora: ";

    Utakmica clashOfTheTitans(move(kopija));

    cout << ((spaceClassico == clashOfTheTitans) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Testiranje operatora '=': ";

    Utakmica mecNaseGeneracije;

    mecNaseGeneracije = spaceClassico;

    cout << ((spaceClassico == mecNaseGeneracije) ? "Ispravno uradjen" : "Nije ispravno uradjen") << endl;

    cout << "Dealokacija..." << endl;

}

void Menu() {

    int nastaviDalje = 1;

    while (nastaviDalje == 1) {

        int izbor = 0;

        do {

            system("cls");

            cout << "\*\*\*Zadaci\*\*\*" << endl;

            cout << "(1) Zadatak 1" << endl;

            cout << "(2) Zadatak 2" << endl;

            cout << "(3) Zadatak 3" << endl;

            cout << "(4) Zadatak 4" << endl;

            cout << "(5) Zadatak 5" << endl;

            cout << "Unesite odgovarajuci broj zadatka za testiranje: -->: ";

            cin >> izbor;

            cout << endl;

        } while (izbor < 1 || izbor > 5);

        switch (izbor) {

        case 1: Zadatak1(); cout << "Zadatak 1. Done." << endl; break;

        case 2: Zadatak2(); cout << "Zadatak 2. Done." << endl; break;

        case 3: Zadatak3(); cout << "Zadatak 3. Done." << endl; break;

        case 4: Zadatak4(); cout << "Zadatak 4. Done." << endl; break;

        case 5: Zadatak5(); cout << "Zadatak 5. Done." << endl; break;

        default:break;

        }

        do {

            cout << "DA LI ZELITE NASTAVITI DALJE? (1/0): ";

            cin >> nastaviDalje;

        } while (nastaviDalje != 0 && nastaviDalje != 1);

    }

}

int main() {

    Menu();

    return 0;

}