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

enum eRazred { PRVI = 1, DRUGI, TRECI, CETVRTI };

enum SortirajPo { T1, T2 };

char\* Alociraj(const char\* sadrzaj) {

if (sadrzaj == nullptr)return nullptr;

int vel = strlen(sadrzaj) + 1;

char\* temp = new char[vel];

strcpy\_s(temp, vel, sadrzaj);

return temp;

}

template<class T1, class T2>

class Kolekcija {

T1\* \_elementi1;

T2\* \_elementi2;

int \_trenutno;

bool \_omoguciDupliranje;

public:

Kolekcija(bool omoguciDupliranje = true) {

\_elementi1 = nullptr;

\_elementi2 = nullptr;

\_omoguciDupliranje = omoguciDupliranje;

}

~Kolekcija() {

delete[]\_elementi1; \_elementi1 = nullptr;

delete[]\_elementi2; \_elementi2 = nullptr;

}

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

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

int getTrenutno() { return \_trenutno; }

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 DatumVrijeme {

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

public:

DatumVrijeme(int dan = 1, int mjesec = 1, int godina = 2000, int sati = 0, int minuti = 0) {

\_dan = new int(dan);

\_mjesec = new int(mjesec);

\_godina = new int(godina);

\_sati = new int(sati);

\_minuti = new int(minuti);

}

~DatumVrijeme() {

delete \_dan; \_dan = nullptr;

delete \_mjesec; \_mjesec = nullptr;

delete \_godina; \_godina = nullptr;

delete \_sati; \_sati = nullptr;

delete \_minuti; \_minuti = nullptr;

}

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

COUT << \*obj.\_dan << "." << \*obj.\_mjesec << "." << \*obj.\_godina << " " << \*obj.\_sati << ":" << \*obj.\_minuti << endl;

return COUT;

}

DatumVrijeme(const DatumVrijeme& obj) {

\_dan = new int(\*obj.\_dan);

\_mjesec = new int(\*obj.\_mjesec);

\_godina = new int(\*obj.\_godina);

\_sati = new int(\*obj.\_sati);

\_minuti = new int(\*obj.\_minuti);

}

};

class Predmet {

char\* \_naziv;

int \_ocjena;

string \_napomena;

public:

Predmet(const char\* naziv = "", int ocjena = 0, string napomena = "") {

\_naziv = Alociraj(naziv);

\_ocjena = ocjena;

\_napomena = napomena;

}

~Predmet() {

delete[] \_naziv; \_naziv = nullptr;

}

friend ostream& operator<< (ostream& COUT, Predmet& obj) {

COUT << obj.\_naziv << " (" << obj.\_ocjena << ") " << obj.\_napomena << endl;

return COUT;

}

string GetNapomena() { return \_napomena; }

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

int GetOcjena() { return \_ocjena; }

void DodajNapomenu(string napomena) {

\_napomena += " " + napomena;

}

};

class Uspjeh {

eRazred\* \_razred;

//formalni argument DatumVrijeme se odnosi na vrijeme evidentiranja polozenog predmeta

Kolekcija<Predmet, DatumVrijeme> \_predmeti;

public:

Uspjeh(eRazred razred) {

\_razred = new eRazred(razred);

}

~Uspjeh() { delete \_razred; \_razred = nullptr; }

Kolekcija<Predmet, DatumVrijeme>\* GetPredmeti() { return &\_predmeti; }

eRazred\* GetERazred() { return \_razred; }

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

COUT << \*obj.\_razred << " " << obj.\_predmeti << endl;

return COUT;

}

};

class Kandidat {

char\* \_imePrezime;

string \_emailAdresa;

string \_brojTelefona;

vector<Uspjeh> \_uspjeh;

public:

Kandidat(const char\* imePrezime, string emailAdresa, string brojTelefona) {

\_imePrezime = Alociraj(imePrezime);

\_emailAdresa = emailAdresa;

\_brojTelefona = brojTelefona;

}

~Kandidat() {

delete[] \_imePrezime; \_imePrezime = nullptr;

}

friend ostream& operator<< (ostream& COUT, Kandidat& obj) {

COUT << obj.\_imePrezime << " " << obj.\_emailAdresa << " " << obj.\_brojTelefona << endl;

return COUT;

}

vector<Uspjeh>\* GetUspjeh() { return &\_uspjeh; }

string GetEmail() { return \_emailAdresa; }

string GetBrojTelefona() { return \_brojTelefona; }

char\* GetImePrezime() { return \_imePrezime; }

};

void main() {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. SVE KLASE TREBAJU POSJEDOVATI ADEKVATAN DESTRUKTOR

2. NAMJERNO IZOSTAVLJANJE KOMPLETNIH I/ILI POJEDINIH DIJELOVA DESTRUKTORA KOJI UZROKUJU RUNTIME ERROR AE BITI OZNACENO KAO "RE"

3. SPAŠAVAJTE PROJEKAT KAKO BI SE SPRIJEEILO GUBLJENJE URA?ENOG ZADATKA

4. PROGRAMSKI CODE SE TAKO?ER NALAZI U FAJLU CODE.TXT

5. NAZIVI FUNKCIJA, TE BROJ I TIP PARAMETARA MORAJU BITI IDENTIENI ONIMA KOJI SU KORIŠTENI U TESTNOM CODE-U, OSIM U SLUEAJU DA POSTOJI ADEKVATAN RAZLOG ZA NJIHOVU MODIFIKACIJU. OSTALE, POMOANE FUNKCIJE MOŽETE IMENOVATI I DODAVATI PO ŽELJI.

6. IZUZETAK BACITE U FUNKCIJAMA U KOJIMA JE TO NAZNAEENO.

7. IMATE POTPUNU SLOBODU DA U MAIN FUNKCIJI, ZA POTREBE TESTIRANJA, DODAJETE VISE POZIVA ILI SALJETE DRUGE VRIJEDNOSTI PARAMETARA

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

DatumVrijeme temp,

datum19062019\_1015(19, 6, 2019, 10, 15),

datum20062019\_1115(20, 6, 2019, 11, 15),

datum30062019\_1215(30, 6, 2019, 12, 15),

datum05072019\_1231(5, 7, 2019, 12, 31);

const int kolekcijaTestSize = 9;

Kolekcija<int, int> kolekcija1(false);

for (size\_t i = 0; i < kolekcijaTestSize; i++)

kolekcija1.AddElement(i + 1, 20 - i);

try {

//ukoliko nije dozvoljeno dupliranje elemenata, metoda AddElement baca izuzetak

kolekcija1.AddElement(6, 15);

}

catch (exception& err) {

cout << err.what() << endl;

}

cout << kolekcija1 << endl;

//na osnovu vrijednosti parametra sortira clanove kolekcije u rastucem redoslijedu

kolekcija1.SortirajRastuci(SortirajPo::T2);

cout << kolekcija1 << endl;

Kolekcija<int, int> kolekcija2 = kolekcija1;

cout << kolekcija2 << crt;

Kolekcija<int, int> kolekcija3;

kolekcija3 = kolekcija1;

cout << kolekcija3 << crt;

//napomena se moze dodati i prilikom kreiranja objekta

Predmet Matematika("Matematika", 5, "Ucesce na takmicenju"),

Fizika("Fizika", 5),

Hemija("Hemija", 2),

Engleski("Engleski", 5);

Fizika.DodajNapomenu("Pohvala za ostvareni uspjeh");

cout << Matematika << endl;

/\*

email adresa mora biti u formatu: text@outlook.com ili text@edu.fit.ba

u slucaju da email adresa nije validna, postaviti je na defaultnu: notSet@edu.fit.ba

za provjeru koristiti regex

\*/

Kandidat jasmin("Jasmin Azemovic", "jasmin@outlook.com", "033 281 172");

Kandidat adel("Adel Handzic", "adel@edu.fit.ba", "033 281 170");

Kandidat emailNotValid("Ime Prezime", "korisnik@klix.ba", "033 281 170");

/\*

uspjeh (tokom srednjoskolskog obrazovanja) se dodaje za svaki predmet na nivou razreda.

tom prilikom onemoguciti:

- dodavanje istoimenih predmeta na nivou jednog razreda,

- dodavanje vise predmeta u kratkom vremenskom periodu (na nivou jednog razreda, razmak izmedju dodavanja pojedinih predmeta mora biti najmanje 5 minuta).

razredi (predmeti ili uspjeh) ne moraju biti dodavani sortiranim redoslijedom (npr. prvo se moze dodati uspjeh za II razred, pa onda za I razred i sl.).

Funkcija vraca true ili false u zavisnosti od (ne)uspjesnost izvrsenja

\*/

if (jasmin.AddPredmet(DRUGI, Fizika, datum20062019\_1115))

cout << "Predmet uspjesno dodan!" << crt;

if (jasmin.AddPredmet(DRUGI, Hemija, datum30062019\_1215))

cout << "Predmet uspjesno dodan!" << crt;

if (jasmin.AddPredmet(PRVI, Engleski, datum19062019\_1015))

cout << "Predmet uspjesno dodan!" << crt;

if (jasmin.AddPredmet(PRVI, Matematika, datum20062019\_1115))

cout << "Predmet uspjesno dodan!" << crt;

//ne treba dodati Matematiku jer je vec dodana u prvom razredu

if (jasmin.AddPredmet(PRVI, Matematika, datum05072019\_1231))

cout << "Predmet uspjesno dodan!" << crt;

//ne treba dodati Fiziku jer nije proslo 5 minuta od dodavanja posljednjeg predmeta

if (jasmin.AddPredmet(PRVI, Fizika, datum20062019\_1115))

cout << "Predmet uspjesno dodan!" << crt;

/\*nakon evidentiranja uspjeha na bilo kojem predmetu kandidatu se salje email sa porukom:

FROM:info@edu.fit.ba

TO: emailKorisnika

Postovani ime i prezime, evidentirali ste uspjeh za X razred.

Pozdrav.

FIT Team.

ukoliko je prosjek na nivou tog razreda veci od 4.5 kandidatu se salje SMS sa porukom: "Svaka cast za uspjeh X.X u X razredu".

slanje poruka i emailova implemenitrati koristeci zasebne thread-ove.

\*/

cout << "USPJEH ISPISATI KORISTECI OSTREAM\_ITERATOR" << endl;

cout << jasmin << endl;

//vraca broj ponavljanja odredjene rijeci u napomenama, koristiti sregex\_iterator

cout << "Rijec takmicenje se pojavljuje " << jasmin.BrojPonavljanjaRijeci("takmicenju") << " puta." << endl;

//vraca niz predmeta koji su evidentiranih u periodu izmedju vrijednosti proslijedjenih parametara

vector<Predmet> jasminUspjeh = jasmin(DatumVrijeme(18, 06, 2019, 10, 15), DatumVrijeme(21, 06, 2019, 10, 10));

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

cout << jasminUspjeh[i] << endl;

Uspjeh\* uspjeh\_Irazred = jasmin[PRVI];//vraca uspjeh kandidata ostvaren u prvom razredu

if (uspjeh\_Irazred != nullptr)

cout << \*uspjeh\_Irazred << endl;

cin.get();

system("pause>0");

}