#include <iostream>

using namespace std;

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

"0. PROVJERITE DA LI PREUZETI ZADACI PRIPADAJU VASOJ GRUPI (G1/G2)\n"

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

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

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

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

"OSIM U SLUCAJU DA POSTOJI ADEKVATAN RAZLOG ZA NJIHOVU MODIFIKACIJU. OSTALE "

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

"5. IZUZETAK BACITE SAMO U FUNKCIJAMA U KOJIMA JE TO NAZNACENO.\n"

"6. FUNKCIJE KOJE NE IMPLEMENTIRATE TREBAJU BITI OBRISANE (KAKO POZIV TAKO I DEFINICIJA)!\n"

"7. NA KRAJU ISPITA SVOJE RJESENJE KOPIRATE U .DOCX FAJL (IMENOVAN BROJEM INDEKSA)!\n"

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

"9. NEMOJTE POSTAVLJATI VISUAL STUDIO PROJEKTE, VEC SAMO .DOCX FAJL SA VASIM RJESENJEM!\n"

"10.SVE NEDOZVOLJENE RADNJE TOKOM ISPITA CE BITI SANKCIONISANE!\n"

"11. ZA POTREBE TESTIRANJA, U MAINU, BUDITE SLOBODNI DODATI TESTNIH PODATAKA (POZIVA FUNKCIJA) KOLIKO GOD SMATRATE DA JE POTREBNO!\n"

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

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

enum Kriteriji { CISTOCA, USLUGA, LOKACIJA, UDOBNOST };

template<class T1, class T2>

class Kolekcija {

T1\* \_elementi1;

T2\* \_elementi2;

int \* \_trenutno;

bool \_omoguciDupliranje;

public:

Kolekcija(bool omoguciDupliranje = true) {

\_elementi1 = nullptr;

\_elementi2 = nullptr;

\_trenutno = 0;

\_omoguciDupliranje = omoguciDupliranje;

}

~Kolekcija() {

delete[]\_elementi1; \_elementi1 = nullptr;

delete[]\_elementi2; \_elementi2 = nullptr;

delete \_trenutno; \_trenutno = 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 Datum {

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

public:

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

\_dan = new int(dan);

\_mjesec = new int(mjesec);

\_godina = new int(godina);

}

~Datum() {

delete \_dan; \_dan = nullptr;

delete \_mjesec; \_mjesec = nullptr;

delete \_godina; \_godina = nullptr;

}

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

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

return COUT;

}

};

class Komentar {

char\* \_sadrzajKomentara;

//uz komentar, gosti mogu ocijeniti svaki od kriterija zasebno, a int se odnosi na ocjenu u rasponu od 1 - 10.

//onemoguciti dupliranje kriterija tj. svaki kriterij se moze ocijeniti samo jednom...u suprotnom baciti objekat tipa exception

Kolekcija<Kriteriji, int>\* \_ocjeneKriterija;

public:

Komentar(const char\* sadrzajKomentara="") {

//kreirajte funkciju GetNizKaraktera ili na drugi adekvatan nacin izvrsite inicijalizaciju pokazivaca

\_sadrzajKomentara = GetNizKaraktera(sadrzajKomentara);

\_ocjeneKriterija = nullptr;

}

~Komentar() {

delete[] \_sadrzajKomentara; \_sadrzajKomentara = nullptr;

delete \_ocjeneKriterija; \_ocjeneKriterija = nullptr;

}

char\* GetSadrzajKomentara() { return \_sadrzajKomentara; }

Kolekcija<Kriteriji, int>\* GetOcjeneKriterija() { return \_ocjeneKriterija; }

};

class Gost {

unique\_ptr<char[]> \_imePrezime;

string \_emailAdresa;

string \_brojPasosa;

public:

Gost(const char\* imePrezime, string emailAdresa, string brojPasosa) {

//\_imePrezime = "INICIJALIZOVATI...";

\_emailAdresa = emailAdresa;

\_brojPasosa = "ISKORISTITI FUNKCIJU ValidirajBrojPasosa";

}

string GetEmail() { return \_emailAdresa; }

string GetBrojPasosa() { return \_brojPasosa; }

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

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

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

return COUT;

}

};

class Rezervacija {

Datum \_OD;

Datum \_DO;

//onemoguciti dodavanje gostiju sa istom email adresom ili brojem pasosa u okviru jedne rezervacije

vector<Gost> \_gosti;

//za svaku rezervaciju se moze ostaviti samo jedan komentar, bez obzira na broj gostiju

Komentar \_komentar;

public:

Rezervacija(Datum & OD, Datum & DO, Gost & gost):\_OD(OD), \_DO(DO) {

\_gosti.push\_back(gost);

}

vector<Gost> & GetGosti() { return \_gosti; }

Komentar GetKomentar() { return \_komentar; }

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

COUT << crt << "Rezervacija " << obj.\_OD << " - " << obj.\_DO <<" za goste: " << endl;

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

COUT <<"\t" << i + 1 << "." << obj.\_gosti[i];

cout <<crt << "Komentar rezervacije: " << endl << obj.\_komentar <<crt;

return COUT;

}

};

const char\* GetOdgovorNaPrvoPitanje() {

cout << "Pitanje -> Pojasnite osnovne preduslove koji moraju biti ispunjeni da bi se realizovao polimorfizam?\n";

return "Odgovor -> OVDJE UNESITE VAS ODGOVOR";

}

const char\* GetOdgovorNaDrugoPitanje() {

cout << "Pitanje -> Pojasnite razloge koristenja kljucnih rijeci abstract i ciste virtualne metode, te razlike izmedju njih?\n";

return "Odgovor -> OVDJE UNESITE VAS ODGOVOR";

}

void main() {

cout << PORUKA;

cin.get();

cout << GetOdgovorNaPrvoPitanje() << endl;

cin.get();

cout << GetOdgovorNaDrugoPitanje() << endl;

cin.get();

Datum

datum19062022(19, 6, 2022),

datum20062022(20, 6, 2022),

datum30062022(30, 6, 2022),

datum05072022(5, 7, 2022);

int kolekcijaTestSize = 9;

Kolekcija<int, int> kolekcija1(false);

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

kolekcija1.AddElement(i, i);//dodaje vrijednosti u kolekciju

try {

//ukoliko nije dozvoljeno dupliranje elemenata (provjeravaju se T1 i T2), metoda AddElement treba baciti objekat tipa exception

kolekcija1.AddElement(3, 3);

}

catch (exception& err) {

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

}

cout << kolekcija1 << crt;

/\* metoda InsertAt treba da doda vrijednosti drugog i treceg parametra na lokaciju koja je definisana prvim parametrom. Povratna vrijednost metode

\* je objekat (pozivaoc metode, u konkretnom slucaju objekat kolekcija1) u okviru koga su, na definisanu lokaciju, dodati zahtijevani parametri.

\* Nakon izvrsenja metode InsertAt, oba objekta, kolekcija1 i kolekcija2, bi trebali posjedovati sljedeci sadrzaj:

\* 10 10

\* 0 0

\* 1 1

\* 2 2

\* ....

\*/

Kolekcija<int, int> kolekcija2 = kolekcija1.InsertAt(0, 10, 10);

cout << kolekcija2 << crt;

/\*

\* za validaciju broja pasosa koristiti funkciju ValidirajBrojPasosa koja treba, koristeci regex, osigurati postivanje

sljedeceg formata:

- pocetak sadrzi jedan ili dva velika slova

- nakon toga slijede tri ili cetiri broja

- nakon toga moze, a ne mora, doci znak crtica (-) ili razmak ( )

- nakon toga slijede dva do cetiri broja

pomenutu funkciju iskoristiti u konstruktoru klase Gost, te ukoliko broj pasosa nije u validnom formatu onda njegova vrijednost treba biti postavljena na NOT VALID

\*/

if (ValidirajBrojPasosa("BH235-532"))

cout << "Broj pasosa validan" << endl;

if (ValidirajBrojPasosa("B123321"))

cout << "Broj pasosa validan" << endl;

if (ValidirajBrojPasosa("B1252 521"))

cout << "Broj pasosa validan" << endl;

if (!ValidirajBrojPasosa("H4521"))

cout << "Broj pasosa NIJE validan" << endl;

if (!ValidirajBrojPasosa("b1252 521"))

cout << "Broj pasosa NIJE validan" << endl;

Gost denis("Denis Music", "denis@fit.ba", "BH235-532");

Gost jasmin("Jasmin Azemovic", "jasmin@fit.ba", "B123321");

Gost adel("Adel Handzic", "adel@edu.fit.ba", "B1252 521");

Gost gostPasosNotValid("Ime Prezime", "korisnik@klix.ba", "H4521");//\_brojPasosa = NOT VALID

//rezervaciji se prilikom kreiranja mora dodijeliti gost koji je nosilac rezervacije

Rezervacija rezervacija(datum19062022, datum20062022, denis);

if(rezervacija.AddGost(jasmin))

cout << "Gost uspjesno dodan!";

Komentar komentarRezervacija("Nismo pretjerano zadovoljni uslugom, a ni lokacijom.");

komentarRezervacija.AddOcjenuKriterija(CISTOCA, 7);

komentarRezervacija.AddOcjenuKriterija(USLUGA, 4);

komentarRezervacija.AddOcjenuKriterija(LOKACIJA, 3);

komentarRezervacija.AddOcjenuKriterija(UDOBNOST,6);

try {

komentarRezervacija.AddOcjenuKriterija(UDOBNOST, 6);//kriterij udobnosti je vec ocijenjen!

}

catch (exception& err) {

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

}

//u okviru rezervacije postavlja vrijednost atributa \_komentar

rezervacija.SetKomentar(komentarRezervacija);

//ukoliko se rezervaciji doda komentar koji sadrzi najmanje dvije ocjene kriterija sa vrijednoscu manjom od 5, svim gostima,

//koji su dodati u rezervaciju, u zasebnom thread-u, se salje email sa sljedecim sadrzajem:

/\*

To: denis@fit.ba;jasmin@fit.ba;

Subject: Informacija

Postovani,

Zaprimili smo Vase ocjene, a njihova prosjecna vrijednost je X.X.

Zao nam je zbog toga, te ce Vas u najkracem periodu kontaktirati nasa Sluzba za odnose sa gostima.

Ugodan boravak Vam zelimo

Puno pozdrava

\*/

//ispisuje sve podatke o rezervaciji, gostima, komentar, ocjene pojedinih kriterija i prosjecnu ocjenu.

cout << rezervacija << endl;

/\* Primjer ispisa:

-------------------------------------------

Rezervacija 19.6.2022 - 20.6.2022 za goste:

1.Denis Music denis@fit.ba BH235-532

2.Jasmin Azemovic jasmin@fit.ba B123321

-------------------------------------------

Komentar rezervacije:

Nismo pretjerano zadovoljni uslugom, a ni lokacijom.

CISTOCA(7)

USLUGA(4)

LOKACIJA(3)

UDOBNOST(6)

-------------------------------------------

Prosjecna ocjena -> 5

\*/

Rezervacija rezervacijaSaAdelom = rezervacija;

if (rezervacijaSaAdelom.AddGost(adel))

cout << "Gost uspjesno dodan!";

if (!rezervacijaSaAdelom.AddGost(denis))

cout << "Gost je vec dodan na rezervaciju!";

cout << rezervacijaSaAdelom << endl;

/\*metoda GetBrojZnakova treba prebrojati koliko se unutar fajla, cije ime je proslijedjeno kao prvi parametar (relativna putanja), nalazi ukupno znakova, te koliko

se unutar pomenutog fajla nalazi trazenih znakova koji su proslijedjeni kao drugi parametar. Vrijednosti brojaca se vracaju kao par.\*/

pair<int, int> brojac = rezervacijaSaAdelom.GetBrojZnakova("rezervacije.txt", "\*\_+");

cout << "Ukupno znakova u fajlu: " << brojac.first << endl;

cout << "Ukupno trazenih znakova: " << brojac.second<< endl;

/\*

ako se u fajlu rezervacije.txt nalazi sadrzaj "denis+music\*\_" onda povratna vrijednost metode GetBrojZnakova treba biti par 13 i 3. za potrebe testiranja kreirajte

fajl sa zeljenim sadrzajem na istoj lokaciji kao i vas .exe fajl. vodite racuna o tome da se zeljeni naziv fajla definise vrijednoscu parametra.

\*/

cin.get();

system("pause>0");

}