1.

#include<iostream>

using namespace std;

class Datum {

public:int dan;

public:int mjesec;

public:int godina;

Datum() {

cout << "Unesite dan: ";

cin >> dan;

cout << "Unesite mjesec: ";

cin >> mjesec;

cout << "Unesite godinu: ";

cin >> godina;

}

Datum(int d, int m, int g) {

dan = d;

mjesec = m;

godina = g;

}

void Ispis() {

cout << dan << "." << mjesec << "." << godina << endl;

}

};

void GetVeciDatum(Datum prvi, Datum drugi) {

//funkcija ispisuje koji od proslijeđena dva datuma je veći

//odnosno noviji

//npr datum 27.01.2019 je 'veći' od 20.01.2019.

cout << "Veći datum je: ";

if(prvi.godina > drugi.godina)

prvi.Ispis();

else if (prvi.godina < drugi.godina)

drugi.Ispis();

else {//godine su jednake

if (prvi.mjesec > drugi.mjesec)

prvi.Ispis();

else if (prvi.mjesec < drugi.mjesec)

drugi.Ispis();

else {//i mjeseci su jednaki

if (prvi.dan > drugi.dan)

prvi.Ispis();

else if (prvi.dan < drugi.dan)

drugi.Ispis();

else //ako su i dani jednaki

cout << "Datumi su jednaki!\n";

}

}

}

void main() {

Datum d1;

int d, m, g;

cout << "Unesite dan: ";

cin >> d;

cout << "Unesite mjesec: ";

cin >> m;

cout << "Unesite godinu: ";

cin >> g;

Datum d2(d,m,g);

GetVeciDatum(d1,d2);

system("pause>0");

}

2.

#include <iostream>

using namespace std;

class PolozeniPredmet {

public:char \* \_naziv;

public: int \_ocjena;

//Napraviti

//1. Defaultni konstruktor

//2. Konstruktor koji prima parametre:naziv i ocjena

//3. Konstruktor kopije

//4. Destruktor

//5. Funkcije definisane ispod

PolozeniPredmet()

{

char pom[] = "----";

int temp = strlen(pom) + 1;

\_naziv = new char[temp];

strcpy\_s(\_naziv, temp, pom);

\_ocjena = 0;

}

PolozeniPredmet(char \* naziv, int ocjena)

{

int temp = strlen(naziv) + 1;

\_naziv = new char[temp];

strcpy\_s(\_naziv, temp, naziv);

\_ocjena = ocjena;

}

PolozeniPredmet(const PolozeniPredmet & original) {

int temp = strlen(original.\_naziv) + 1;

\_naziv = new char[temp];

strcpy\_s(\_naziv, temp, original.\_naziv);

\_ocjena = original.\_ocjena;

}

void Kopiraj(const PolozeniPredmet & original) {

//Kopira vrijednosti iz proslijeđenog objekta u trenutni

delete[]\_naziv; // zbog onoga sto se prije nalazilo u ovom nizu

int temp = strlen(original.\_naziv) + 1;

\_naziv = new char[temp];

strcpy\_s(\_naziv, temp, original.\_naziv);

\_ocjena = original.\_ocjena;

}

~PolozeniPredmet() { delete[] \_naziv; \_naziv = nullptr; }

};

class KolekcijaPolozenih {

PolozeniPredmet \* \_polozeni;

int \_trenutno;

int \_max;

public:

//konstruktor,

//konstruktor kopije,

//destruktor

//potrebne funkcije

KolekcijaPolozenih(int max = 5) {

\_polozeni = new PolozeniPredmet[max];

\_trenutno = 0;

\_max = max;

}

void AddPolozeniPredmet (PolozeniPredmet & original) {

//Dodaje novi polozeni predmet u niz

//Voditi računa da se ne moze dodavati isti predmet (istog naziva)

cout << "POKUSAJ DODAVANJA PREDMETA " << original.\_naziv << endl;

if (\_trenutno == \_max) {

cout << "Nemoguće dodati više od " << \_max << " predmeta.\n";

}

else {

for (int i = 0; i < \_trenutno; i++)

{

if (!strcmp(original.\_naziv, \_polozeni[i].\_naziv)) {

cout << "predmet nemoguce dodati jer vec postoji isti " << endl;

return;

}

}

\_polozeni[\_trenutno].Kopiraj(original);

\_trenutno++;

cout << "Predmet uspjesno dodat" << endl;

}

}

void ispis() {

for (int i = 0; i <\_trenutno; i++)

{

cout << \_polozeni[i].\_naziv << \_polozeni[i].\_ocjena << endl;

}

}

~KolekcijaPolozenih() {

delete[] \_polozeni;

\_polozeni = nullptr;

}

};

void main() {

KolekcijaPolozenih K(1);

char naziv[100];

cout << "Unesite naziv ";

cin.getline(naziv, 100);

PolozeniPredmet A(naziv, 5);

cout << "Unesite naziv ";

cin.getline(naziv, 100);

PolozeniPredmet B(naziv, 5);

cout << "Unesite naziv ";

cin.getline(naziv, 100);

PolozeniPredmet C(naziv, 5);

K.ispis();

cout << "\n==============================" << endl;

K.AddPolozeniPredmet(A);

cout << "\n==============================" << endl;

K.AddPolozeniPredmet(B);

cout << "\n==============================" << endl;

K.AddPolozeniPredmet(C);

cout << "\n==============================" << endl;

K.ispis();

cout << "\n==============================" << endl;

system("pause>0");

}

3.

#include<iostream>

using namespace std;

class Vozilo

{

private:

char\* markaVozila;

int godProizvodnje;

public:

Vozilo(char\* marka = "---", int god = 0) {

markaVozila = new char[strlen(marka) + 1];

strcpy\_s(markaVozila, strlen(marka) + 1, marka);

godProizvodnje = god;

}

~Vozilo() {

delete[] markaVozila; markaVozila = nullptr;

}

Vozilo(Vozilo &original) {

markaVozila = new char[strlen(original.markaVozila) + 1];

strcpy\_s(markaVozila, strlen(original.markaVozila) + 1, original.markaVozila);

godProizvodnje = original.godProizvodnje;

}

void Kopiraj(Vozilo original) {

markaVozila = new char[strlen(original.markaVozila) + 1];

strcpy\_s(markaVozila, strlen(original.markaVozila) + 1, original.markaVozila);

godProizvodnje = original.godProizvodnje;

}

void Ispis() {

cout << "Marka vozila: " << markaVozila << endl;

cout << "Godina proizvodnje: " << godProizvodnje << endl;

}

};

class Vozac

{

char ime[20];

char\* prezime;

Vozilo\* NjegovoVozilo;

int BrVozacke;

public:

Vozac(char\* \_ime = "---", char\* \_pre = "---", int brvoz = 0) :NjegovoVozilo(nullptr) {

strcpy\_s(ime, 20, \_ime);

prezime = new char[strlen(\_pre) + 1];

strcpy\_s(prezime, strlen(\_pre) + 1, \_pre);

BrVozacke = brvoz;

NjegovoVozilo = new Vozilo();

}

~Vozac() {

delete[] prezime; prezime = nullptr;

}

Vozac(Vozac &original) {

strcpy\_s(ime, 20, original.ime);

prezime = new char[strlen(original.prezime) + 1];

strcpy\_s(prezime, strlen(original.prezime) + 1, original.prezime);

BrVozacke = original.BrVozacke;

NjegovoVozilo = new Vozilo();

NjegovoVozilo->Kopiraj(\*original.NjegovoVozilo);

}

void Ispis() {

cout << "Ime: " << ime << endl;

cout << "Prezime: " << prezime << endl;

cout << "Broj vozacke: " << BrVozacke << endl;

cout << "Vozilo: ";

NjegovoVozilo->Ispis();

cout << endl;

}

void DodajVozilo(Vozilo v) {

delete NjegovoVozilo;

NjegovoVozilo = new Vozilo;

NjegovoVozilo->Kopiraj(v);

}

};

void main() {

Vozac prvi("Test", "Test", 1234);

Vozilo prvo("BMW", 2016);

prvi.DodajVozilo(prvo);

prvi.Ispis();

system("pause>0");

}