```cpp
#include <iostream>
using namespace std;

/* ::NAPOMENA::
Radi jednostavnije izrade i lakseg testiranja zadataka, komentirajte testni dio k
oda, koji je obuhvacen funkcijama sa prefixom 'Zadatak'.
Kako budete implementirali odredjene funkcionalnosti, tada lagano pocnite sa 'otk
rivanjem' komentiranih dijelova koda.
*/

//Koristene skracenice u komentarima
// dflt. = default
// user-def. = user-defined (korisnicki-definirano)
// ctor = constructor (konstruktor)
// dtor = destructor (destruktor
// dflt. ctor = default constructor
// user-def. ctor = user-defined constructor

/******KRATAK PODSJETNIK[1] ******
Kreiranje objekata (varijabli) u statickoj memoriji [KORISTENJE KONSTRUKTORA]
    int(5); //Neimenovani objekat
    int broj(5);
    int broj = 5;

Kreiranje objekata (varijabli) u dinamickoj memoriji [KORISTENJE KONSTRUKTORA]
    new int(5);
    int* pok = new int(5);

Ekvivalentno tome imamo sljedece:--->
Staticka memorija:
    Student("IB200200"); //Neimenovani objekat, user-def ctor
    Student student("IB200200"); // user-def ctor
    //Niz:
    Student niz[100]; //poziva se se dflt. ctor za svaki element niza

Dinamicka memorija:
    new Student; //poziva se dflt. ctor
    new Student("IB200200"); // poziva se user-def. ctor
    Student * pok = new Student; //poziva se dflt. ctor
    Student * pok = new Student("IB200200"); //poziva se user-def. ctor
    //Niz:
    Student * niz = new Student[100]; //poziva se se dflt. ctor za svaki element
niza
*/
```

```
/******KRATAK PODSJETNIK[2] :: Pokazivaci i dinamicka memorija******
    1.Pokazivac na objekat:
        Student * student = new Student; // dflt. ctor se poziva jednom
    2.Pokazivac na niz:
        Student * studenti = new Student[10]; // dflt. ctor se poziva za svaki el
ement niza
    3.Staticki niz pokazivaca [gdje svaki pokazuje na tacno jedan objekat]
        Student * nizPok[100];
        for(int i=0;i<100;i++)
            nizPok[i] = new Student; // poziv dflt. ctor-a
    4.Staticki niz pokazivaca [gdje svaki pokazuje na niz objekata]
        Student * nizPok[100];
        for(int i=0;i<100;i++)
            nizPok[i] = new Student[vel]; // poziv dflt.ctor-
a onoliko puta, kolika je vrijednost 'vel'
    5.Pokazivac na pokazivac
        Student ** temp = new Student*;
        *temp = new Student; //poziv dflt. ctora
    6.Dinamicki niz pokazivaca [gdje svaki pokazuje na tacno jedan objekat]
        Student ** temp = new Student*[100];
        for(int i=0;i<100;i++)
            temp[i] = new Student;
    7.Dinamicki niz pokazivaca [gdje svaki pokazuje na niz objekata]
    Student ** temp = new Student*[100];
        for(int i=0;i<100;i++)
            temp[i] = new Student[vel];
*/

//Z1.0
char* AlocirajIKopiraj(const char* tekst);
class Datum
{
private:
    int _dan;
    int _mjesec;
    int _godina;
public:
    //Z1.1 Dflt. ctor [Postaviti na dflt. vrijednosti (1.1.2021)]
    Datum();
    //Z1.2 User-def. ctor
    Datum(int d, int m, int g);
    //Z1.3
    int GetDan() const;
    int GetMjesec() const;
    int GetGodina() const;
```

```
    void SetDan(int dan);
    void SetMjesec(int mjesec);
    void SetGodina(int godina);
    //Z1.4
    void Ispis();
    //Z1.5
    ~Datum();
};

//Vraca random string odredjene duzine, kojeg sacinjavaju velika i mala slova
char* GetRandomString(int duzina) {
    int velicina = duzina + 1;
    char* randomString = new char[velicina];
    for (size_t i = 0; i < velicina; i++) {
        int random_ascii = rand() % 26 + 65;
        char slovo = (char)random_ascii;
        bool pretvoriUMalo = (bool)(rand() % 2);
        if (pretvoriUMalo)
            slovo = tolower(slovo);
        randomString[i] = slovo;
    }
    randomString[velicina - 1] = '\0';
    return randomString;
}

class Sjediste {
private:
    char* _sjedisteId; // Koristiti GetRandomString funkciju prilikom inicijaliza
cije [neka je duzina stringa 10]
    char _red; //A-Z
    int _kolona; //1,2,3,..., 100
public:
    //Z2.1
    //Dflt. ctor
    //Postaviti red na ' ' a kolonu na 0
    Sjediste();
    //Z2.2
    //User-def. ctor
    Sjediste(char red, int kolona);
    //Z2.3
    char GetRed() const;
    int GetKolona() const;
    //Z2.4
    void SetRed(char red);
    void SetKolona(int kolona);
```

```
    //Z2.5
    //Ispisati sjediste u formatu: "[A-13]"
    void Ispis();
    //Z2.6
    ~Sjediste();
};

class Dvorana
{
private:
    char* _naziv;
    int _brojRedova; //max.26 [A-Z]
    int _brojKolona;
    Sjediste** _sjedista; //2D matrica objekata tipa Sjediste. Dimenzije matrice
su: (_brojRedova x _brojKolona)
public:
    //Z2.7
    Dvorana();
    //Z2.8
    //Uraditi dinamicku alokaciju 2D matrice sjedista pomocu pokazivaca na niz po
kazivaca (_sjedista)
    //Podesiti vrijednosti atributa objekata matrice na sljedeci nacin -->
    //Polje [0][0] => _red : A, _kolona : 1
    //Polje [0][1] => _red : A, _kolona : 2
    //Polje [0][2] => _red : A, _kolona : 3
    //...
    //Polje [1][0] => _red : B, _kolona : 1
    //itd.
    Dvorana(const char* naziv, int brojRedova, int brojKolona);

    //Z2.9
    void SetNaziv(const char* naziv);
    const char* GetNaziv() const;
    int GetBrojRedova() const;
    int GetBrojKolona() const;

    //Z2.10
    //Uraditi dealokaciju alocirane matrice sjedista, te zatim uraditi ponovnu al
okaciju na osnovu novih dimenzija
    void SetSjedista(int brojRedova, int brojKolona);

    //Z2.11
    //Vratiti adresu objekta sjediste, koje odgovara proslijedjenim vrijednostima
    //Ukoliko nije pronadjeno, vratiti nullptr
    Sjediste* GetSjediste(char red, int kolona);
```

```
    //Z2.12
    //Uraditi ispis na sljedeci nacin --->
    //Dvorana: Dvorana 7 Extreme
    //Redova: 11
    //Kolona: 20
    // ----------------------------
    // [A-1] [A-2] [A-3] .... [A-20]
    // [B-1] [B-2] [B-3] .... [B-20]
    // .
    // .
    // .
    // [K-1] [K-2] [K-3] .... [K-20]
    // ----------------------------
    void Ispis();

    //Z2.13
    ~Dvorana();
};
class Uposlenik {
private:
    char* _ime;
    char* _prezime;
    Datum* _datumRodjenja;
    bool* _spol;
    char _radnoMjesto[100];
    char* _grad;
    char _email[100];
public:
    //Z3.1
    Uposlenik();
    //Z3.2
    Uposlenik(const char* ime, const char* prezime, Datum datum, bool spol, const
 char* radnoM, const char* grad, const char* email);

    //Z3.3
    char* GetIme() const;
    char* GetPrezime() const;
    Datum GetDatumRodjenja() const;
    bool GetSpol() const;
    const char* GetRadnoMjesto() const;
    char* GetGrad() const;
    const char* GetEmail() const;

    //Z3.4
```

```
    void SetIme(const char* ime);
    void SetPrezime(const char* prezime);
    void SetDatumRodjenja(Datum datum);
    void SetSpol(bool spol);
    void SetRadnoMjesto(const char* radnoMjesto);
    void SetGrad(const char* grad);
    void SetEmail(const char* email);
    //Z3.5
    void Ispis();
    //Z3.6
    ~Uposlenik();
};

class Kino {
private:
    char* _naziv;
    char _adresa[100];

    int* _maxBrojDvorana; // pokazivac na varijablu (obezbijediti da sadrzi vrije
dnost velicine niza)
    int _trenutnoDvorana; //brojac objekata u dinamickom nizu (pocinje od 0)
    Dvorana* _dvorane; // pokazivac na niz objekata

    int _trenutnoUposlenika; //brojac
    Uposlenik* _uposlenici[50] = { nullptr }; // niz pokazivaca na objekte tipa U
poslenik

public:
    //Z4.1
    Kino(const char* naziv, const char* adresa, int maxBrojDvorana);
    //Z4.3
    void SetNaziv(const char* naziv);
    //Z4.4
    char* GetNaziv() const;
    const char* GetAdresa() const;
    int GetMaxBrojDvorana() const;
    int GetTrenutnoDvorana() const;

    //Z4.5
    bool DodajDvoranu(Dvorana& dvorana);
    //Z4.6
    bool DodajUposlenika(Uposlenik& uposlenik);
    //Z4.7
     //Ispisati naziv kina, adresu, nazive dvorana, te imena i prezimena uposleni
ka
```

```
    void Ispis();
    //Z4.8
    ~Kino();
};
void Zadatak1() {
    cout << "Testiranje klase 'Datum'" << endl;
    Datum euro2021; // Testiranje dflt. ctora
    euro2021.SetDan(11);
    euro2021.SetMjesec(6);
    euro2021.SetGodina(2021);
    Datum olimpijskeIgre(24, 7, 2021); // Testiranje user-def. ctora
    Datum paraOlimprijskeIgre(olimpijskeIgre.GetDan(), olimpijskeIgre.GetMjesec()
, olimpijskeIgre.GetGodina());
    paraOlimprijskeIgre.SetMjesec(8);
    cout << "EURO 2021:" << endl;
    euro2021.Ispis();
    cout << endl;
    cout << "Olimpijske igre 2021:" << endl;
    olimpijskeIgre.Ispis();
    cout << endl;
    cout << "Paraolimpijske igre 2021:" << endl;
    paraOlimprijskeIgre.Ispis();
    cout << endl;
    cout << "Dealokacija ..." << endl;
}

void Zadatak2() {
    cout << "Testiranje klase 'Sjediste'" << endl;
    Sjediste s1, s2('C', 4);
    Sjediste s3(s2.GetRed(), s2.GetKolona());
    s3.Ispis();
    cout << endl;
    s3.SetRed('M');
    s3.SetKolona(16);
    s3.Ispis();
    cout << endl;
    cout << "Done." << "\n\n";

    cout << "Testiranje klase 'Dvorana'" << endl;
    Dvorana extreme3("Dvorana Extreme 3", 7, 7);
    extreme3.Ispis();
    cout << endl;
    Dvorana realExtreme3(extreme3.GetNaziv(), extreme3.GetBrojRedova(), extreme3.
GetBrojKolona());
    realExtreme3.SetNaziv("Dvorana Real Extreme 3");
```

```
    realExtreme3.SetSjedista(10, 10);
    realExtreme3.Ispis();
    cout << endl;
    Dvorana real3D;
    real3D.SetNaziv("Dvorana Real 3D");
    real3D.SetSjedista(15, 8);
    real3D.Ispis();
    cout << endl;

    Sjediste* pok = real3D.GetSjediste('B', 2);
    cout << "Trazimo sjediste B-2 U dvorani Real 3D." << endl;
    if (pok != nullptr) {
        pok->Ispis();
        cout << endl << "Nadjeno." << endl;
    }
    else {
        cout << "Nije nadjeno." << endl;
    }
    pok = real3D.GetSjediste('X', 3);
    cout << "Trazimo sjediste X-3 u dvorani Real 3D." << endl;
    if (pok != nullptr) {
        pok->Ispis();
        cout << endl << "Nadjeno." << endl;
    }
    else {
        cout << "Nije nadjeno." << "\n\n";
    }
    cout << "Dealokacija dvorana..." << endl;
}
void Zadatak3() {
    cout << "Testiranje klase 'Uposlenik'" << endl;

    Datum datumRodjenja(3, 2, 1967);
    Uposlenik connanOBrien("Connan", "O'Brien", datumRodjenja, 1, "blagajnik", "B
ihac", "teamcoco@gmail.com");
    connanOBrien.SetEmail("orangecoconut@gmail.com");
    connanOBrien.Ispis();
    cout << endl;

    Uposlenik jimmyKimmel;
    jimmyKimmel.SetIme("Jimmy");
    jimmyKimmel.SetPrezime("Kimmel");
    jimmyKimmel.SetDatumRodjenja(Datum(4, 2, 1966));
    jimmyKimmel.SetSpol(true);
    jimmyKimmel.SetEmail("the_pranklord@hotmail.com");
```

```
    jimmyKimmel.SetRadnoMjesto("prodavac kokica");
    jimmyKimmel.SetGrad("Portland");
    jimmyKimmel.Ispis();
    cout << endl;

    Uposlenik chuckNorris;
    chuckNorris.SetIme("Chuck");
    chuckNorris.SetPrezime("Norris");
    chuckNorris.SetDatumRodjenja(Datum(30, 2, 1940));
    chuckNorris.SetSpol(true);
    chuckNorris.SetEmail("gmail@chucknorris.com");
    chuckNorris.SetRadnoMjesto("redar, domar, financijski menadzer, projekcijski
menadzer, direktor, ...");
    chuckNorris.SetGrad("Planeta Zemlja");
    chuckNorris.Ispis();
    cout << endl;
    cout << "Dealokacija uposlenika..." << endl;
}

void Zadatak4() {
    Kino dejaView("Deja View Cinema", "Bihac Ul. 76", 5);
    Dvorana d1("Dvorana 1", 10, 5), d2("Dvorana 2", 10, 6);
    cout << (dejaView.DodajDvoranu(d1) ?
        "Uspjesno dodavanje dvorane d1." :
        "Bezuspjesno dodavanje dvorane d1.") << endl;
    cout << (dejaView.DodajDvoranu(d2) ?
        "Uspjesno dodavanje dvorane d2." :
        "Bezuspjesno dodavanje dvorane d2.") << endl;;

    Datum datum1(2, 3, 1966), datum2(2, 4, 1969), datum3(1, 5, 1955);
    Uposlenik u1("Jonah", "Hillside", datum1, 1, "prodavac ulaznica", "Banja Luka
", "jonahillside@gmail.com");
    Uposlenik u2("Dwayne", "'The Mountain' Johnson", datum2, 1, "domar", "Sarajev
o", "johnrocksbigtime@gmail.com");
    Uposlenik u3("Tom", "Cruiser", datum3, 1, "vozac", "Brcko", "thomas_cruiser@g
mail.com");

    if (dejaView.DodajUposlenika(u1))
        cout << "Uspjesno dodan " << u1.GetIme() << " " << u1.GetPrezime() << end
l;
    if (dejaView.DodajUposlenika(u2))
        cout << "Uspjesno dodan " << u2.GetIme() << " " << u2.GetPrezime() << end
l;
    if (dejaView.DodajUposlenika(u3))
```

```
        cout << "Uspjesno dodan " << u3.GetIme() << " " << u3.GetPrezime() << end
l;
    dejaView.Ispis();
    cout << 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 << "Unesite odgovarajuci broj zadatka za testiranje: -->: ";
            cin >> izbor;
            cout << endl;
        } while (izbor < 1 || izbor > 4);
        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;
        default:break;
        }
        do {
            cout << "DA LI ZELITE NASTAVITI DALJE? (1/0): ";
            cin >> nastaviDalje;
        } while (nastaviDalje != 0 && nastaviDalje != 1);
    }
}
int main() {
    Menu();
    return 0;
}

#include <iostream>
using namespace std;

/* ::NAPOMENA::
```

```
Radi jednostavnije izrade i lakseg testiranja zadataka, komentirajte testni dio k
oda, koji je obuhvacen funkcijama sa prefixom 'Zadatak'.
Kako budete implementirali odredjene funkcionalnosti, tada lagano pocnite sa 'otk
rivanjem' komentiranih dijelova koda.
*/

//Koristene skracenice u komentarima
// dflt. = default
// user-def. = user-defined (korisnicki-definirano)
// ctor = constructor (konstruktor)
// dtor = destructor (destruktor
// dflt. ctor = default constructor
// user-def. ctor = user-defined constructor

/******KRATAK PODSJETNIK[1] ******
Kreiranje objekata (varijabli) u statickoj memoriji [KORISTENJE KONSTRUKTORA]
    int(5); //Neimenovani objekat
    int broj(5);
    int broj = 5;

Kreiranje objekata (varijabli) u dinamickoj memoriji [KORISTENJE KONSTRUKTORA]
    new int(5);
    int* pok = new int(5);

Ekvivalentno tome imamo sljedece:--->
Staticka memorija:
    Student("IB200200"); //Neimenovani objekat, user-def ctor
    Student student("IB200200"); // user-def ctor
    //Niz:
    Student niz[100]; //poziva se se dflt. ctor za svaki element niza

Dinamicka memorija:
    new Student; //poziva se dflt. ctor
    new Student("IB200200"); // poziva se user-def. ctor
    Student * pok = new Student; //poziva se dflt. ctor
    Student * pok = new Student("IB200200"); //poziva se user-def. ctor
    //Niz:
    Student * niz = new Student[100]; //poziva se se dflt. ctor za svaki element
niza
*/

/******KRATAK PODSJETNIK[2] :: Pokazivaci i dinamicka memorija******
    1.Pokazivac na objekat:
        Student * student = new Student; // dflt. ctor se poziva jednom
    2.Pokazivac na niz:
```

```
        Student * studenti = new Student[10]; // dflt. ctor se poziva za svaki el
ement niza
    3.Staticki niz pokazivaca [gdje svaki pokazuje na tacno jedan objekat]
        Student * nizPok[100];
        for(int i=0;i<100;i++)
            nizPok[i] = new Student; // poziv dflt. ctor-a
    4.Staticki niz pokazivaca [gdje svaki pokazuje na niz objekata]
        Student * nizPok[100];
        for(int i=0;i<100;i++)
            nizPok[i] = new Student[vel]; // poziv dflt.ctor-
a onoliko puta, kolika je vrijednost 'vel'
    5.Pokazivac na pokazivac
        Student ** temp = new Student*;
        *temp = new Student; //poziv dflt. ctora
    6.Dinamicki niz pokazivaca [gdje svaki pokazuje na tacno jedan objekat]
        Student ** temp = new Student*[100];
        for(int i=0;i<100;i++)
            temp[i] = new Student;
    7.Dinamicki niz pokazivaca [gdje svaki pokazuje na niz objekata]
    Student ** temp = new Student*[100];
        for(int i=0;i<100;i++)
            temp[i] = new Student[vel];
*/

//Z1.0
char* AlocirajIKopiraj(const char* tekst);
class Datum
{
private:
    int _dan;
    int _mjesec;
    int _godina;
public:
    //Z1.1 Dflt. ctor [Postaviti na dflt. vrijednosti (1.1.2021)]
    Datum();
    //Z1.2 User-def. ctor
    Datum(int d, int m, int g);
    //Z1.3
    int GetDan() const;
    int GetMjesec() const;
    int GetGodina() const;
    void SetDan(int dan);
    void SetMjesec(int mjesec);
    void SetGodina(int godina);
    //Z1.4
```

```
    void Ispis();
    //Z1.5
    ~Datum();
};

//Vraca random string odredjene duzine, kojeg sacinjavaju velika i mala slova
char* GetRandomString(int duzina) {
    int velicina = duzina + 1;
    char* randomString = new char[velicina];
    for (size_t i = 0; i < velicina; i++) {
        int random_ascii = rand() % 26 + 65;
        char slovo = (char)random_ascii;
        bool pretvoriUMalo = (bool)(rand() % 2);
        if (pretvoriUMalo)
            slovo = tolower(slovo);
        randomString[i] = slovo;
    }
    randomString[velicina - 1] = '\0';
    return randomString;
}

class Sjediste {
private:
    char* _sjedisteId; // Koristiti GetRandomString funkciju prilikom inicijaliza
cije [neka je duzina stringa 10]
    char _red; //A-Z
    int _kolona; //1,2,3,..., 100
public:
    //Z2.1
    //Dflt. ctor
    //Postaviti red na ' ' a kolonu na 0
    Sjediste();
    //Z2.2
    //User-def. ctor
    Sjediste(char red, int kolona);
    //Z2.3
    char GetRed() const;
    int GetKolona() const;
    //Z2.4
    void SetRed(char red);
    void SetKolona(int kolona);
    //Z2.5
    //Ispisati sjediste u formatu: "[A-13]"
    void Ispis();
    //Z2.6
```

```
    ~Sjediste();
};

class Dvorana
{
private:
    char* _naziv;
    int _brojRedova; //max.26 [A-Z]
    int _brojKolona;
    Sjediste** _sjedista; //2D matrica objekata tipa Sjediste. Dimenzije matrice
su: (_brojRedova x _brojKolona)
public:
    //Z2.7
    Dvorana();
    //Z2.8
    //Uraditi dinamicku alokaciju 2D matrice sjedista pomocu pokazivaca na niz po
kazivaca (_sjedista)
    //Podesiti vrijednosti atributa objekata matrice na sljedeci nacin -->
    //Polje [0][0] => _red : A, _kolona : 1
    //Polje [0][1] => _red : A, _kolona : 2
    //Polje [0][2] => _red : A, _kolona : 3
    //...
    //Polje [1][0] => _red : B, _kolona : 1
    //itd.
    Dvorana(const char* naziv, int brojRedova, int brojKolona);

    //Z2.9
    void SetNaziv(const char* naziv);
    const char* GetNaziv() const;
    int GetBrojRedova() const;
    int GetBrojKolona() const;

    //Z2.10
    //Uraditi dealokaciju alocirane matrice sjedista, te zatim uraditi ponovnu al
okaciju na osnovu novih dimenzija
    void SetSjedista(int brojRedova, int brojKolona);

    //Z2.11
    //Vratiti adresu objekta sjediste, koje odgovara proslijedjenim vrijednostima
    //Ukoliko nije pronadjeno, vratiti nullptr
    Sjediste* GetSjediste(char red, int kolona);

    //Z2.12
    //Uraditi ispis na sljedeci nacin --->
    //Dvorana: Dvorana 7 Extreme
```

```
    //Redova: 11
    //Kolona: 20
    // ----------------------------
    // [A-1] [A-2] [A-3] .... [A-20]
    // [B-1] [B-2] [B-3] .... [B-20]
    // .
    // .
    // .
    // [K-1] [K-2] [K-3] .... [K-20]
    // ----------------------------
    void Ispis();

    //Z2.13
    ~Dvorana();
};
class Uposlenik {
private:
    char* _ime;
    char* _prezime;
    Datum* _datumRodjenja;
    bool* _spol;
    char _radnoMjesto[100];
    char* _grad;
    char _email[100];
public:
    //Z3.1
    Uposlenik();
    //Z3.2
    Uposlenik(const char* ime, const char* prezime, Datum datum, bool spol, const
 char* radnoM, const char* grad, const char* email);

    //Z3.3
    char* GetIme() const;
    char* GetPrezime() const;
    Datum GetDatumRodjenja() const;
    bool GetSpol() const;
    const char* GetRadnoMjesto() const;
    char* GetGrad() const;
    const char* GetEmail() const;

    //Z3.4
    void SetIme(const char* ime);
    void SetPrezime(const char* prezime);
    void SetDatumRodjenja(Datum datum);
    void SetSpol(bool spol);
```

```
    void SetRadnoMjesto(const char* radnoMjesto);
    void SetGrad(const char* grad);
    void SetEmail(const char* email);
    //Z3.5
    void Ispis();
    //Z3.6
    ~Uposlenik();
};

class Kino {
private:
    char* _naziv;
    char _adresa[100];

    int* _maxBrojDvorana; // pokazivac na varijablu (obezbijediti da sadrzi vrije
dnost velicine niza)
    int _trenutnoDvorana; //brojac objekata u dinamickom nizu (pocinje od 0)
    Dvorana* _dvorane; // pokazivac na niz objekata

    int _trenutnoUposlenika; //brojac
    Uposlenik* _uposlenici[50] = { nullptr }; // niz pokazivaca na objekte tipa U
poslenik

public:
    //Z4.1
    Kino(const char* naziv, const char* adresa, int maxBrojDvorana);
    //Z4.3
    void SetNaziv(const char* naziv);
    //Z4.4
    char* GetNaziv() const;
    const char* GetAdresa() const;
    int GetMaxBrojDvorana() const;
    int GetTrenutnoDvorana() const;

    //Z4.5
    bool DodajDvoranu(Dvorana& dvorana);
    //Z4.6
    bool DodajUposlenika(Uposlenik& uposlenik);
    //Z4.7
     //Ispisati naziv kina, adresu, nazive dvorana, te imena i prezimena uposleni
ka
    void Ispis();
    //Z4.8
    ~Kino();
};
```

```
void Zadatak1() {
    cout << "Testiranje klase 'Datum'" << endl;
    Datum euro2021; // Testiranje dflt. ctora
    euro2021.SetDan(11);
    euro2021.SetMjesec(6);
    euro2021.SetGodina(2021);
    Datum olimpijskeIgre(24, 7, 2021); // Testiranje user-def. ctora
    Datum paraOlimprijskeIgre(olimpijskeIgre.GetDan(), olimpijskeIgre.GetMjesec()
, olimpijskeIgre.GetGodina());
    paraOlimprijskeIgre.SetMjesec(8);
    cout << "EURO 2021:" << endl;
    euro2021.Ispis();
    cout << endl;
    cout << "Olimpijske igre 2021:" << endl;
    olimpijskeIgre.Ispis();
    cout << endl;
    cout << "Paraolimpijske igre 2021:" << endl;
    paraOlimprijskeIgre.Ispis();
    cout << endl;
    cout << "Dealokacija ..." << endl;
}

void Zadatak2() {
    cout << "Testiranje klase 'Sjediste'" << endl;
    Sjediste s1, s2('C', 4);
    Sjediste s3(s2.GetRed(), s2.GetKolona());
    s3.Ispis();
    cout << endl;
    s3.SetRed('M');
    s3.SetKolona(16);
    s3.Ispis();
    cout << endl;
    cout << "Done." << "\n\n";

    cout << "Testiranje klase 'Dvorana'" << endl;
    Dvorana extreme3("Dvorana Extreme 3", 7, 7);
    extreme3.Ispis();
    cout << endl;
    Dvorana realExtreme3(extreme3.GetNaziv(), extreme3.GetBrojRedova(), extreme3.
GetBrojKolona());
    realExtreme3.SetNaziv("Dvorana Real Extreme 3");
    realExtreme3.SetSjedista(10, 10);
    realExtreme3.Ispis();
    cout << endl;
    Dvorana real3D;
```

```
    real3D.SetNaziv("Dvorana Real 3D");
    real3D.SetSjedista(15, 8);
    real3D.Ispis();
    cout << endl;

    Sjediste* pok = real3D.GetSjediste('B', 2);
    cout << "Trazimo sjediste B-2 U dvorani Real 3D." << endl;
    if (pok != nullptr) {
        pok->Ispis();
        cout << endl << "Nadjeno." << endl;
    }
    else {
        cout << "Nije nadjeno." << endl;
    }
    pok = real3D.GetSjediste('X', 3);
    cout << "Trazimo sjediste X-3 u dvorani Real 3D." << endl;
    if (pok != nullptr) {
        pok->Ispis();
        cout << endl << "Nadjeno." << endl;
    }
    else {
        cout << "Nije nadjeno." << "\n\n";
    }
    cout << "Dealokacija dvorana..." << endl;
}
void Zadatak3() {
    cout << "Testiranje klase 'Uposlenik'" << endl;

    Datum datumRodjenja(3, 2, 1967);
    Uposlenik connanOBrien("Connan", "O'Brien", datumRodjenja, 1, "blagajnik", "B
ihac", "teamcoco@gmail.com");
    connanOBrien.SetEmail("orangecoconut@gmail.com");
    connanOBrien.Ispis();
    cout << endl;

    Uposlenik jimmyKimmel;
    jimmyKimmel.SetIme("Jimmy");
    jimmyKimmel.SetPrezime("Kimmel");
    jimmyKimmel.SetDatumRodjenja(Datum(4, 2, 1966));
    jimmyKimmel.SetSpol(true);
    jimmyKimmel.SetEmail("the_pranklord@hotmail.com");
    jimmyKimmel.SetRadnoMjesto("prodavac kokica");
    jimmyKimmel.SetGrad("Portland");
    jimmyKimmel.Ispis();
    cout << endl;
```

```
    Uposlenik chuckNorris;
    chuckNorris.SetIme("Chuck");
    chuckNorris.SetPrezime("Norris");
    chuckNorris.SetDatumRodjenja(Datum(30, 2, 1940));
    chuckNorris.SetSpol(true);
    chuckNorris.SetEmail("gmail@chucknorris.com");
    chuckNorris.SetRadnoMjesto("redar, domar, financijski menadzer, projekcijski
menadzer, direktor, ...");
    chuckNorris.SetGrad("Planeta Zemlja");
    chuckNorris.Ispis();
    cout << endl;
    cout << "Dealokacija uposlenika..." << endl;
}

void Zadatak4() {
    Kino dejaView("Deja View Cinema", "Bihac Ul. 76", 5);
    Dvorana d1("Dvorana 1", 10, 5), d2("Dvorana 2", 10, 6);
    cout << (dejaView.DodajDvoranu(d1) ?
        "Uspjesno dodavanje dvorane d1." :
        "Bezuspjesno dodavanje dvorane d1.") << endl;
    cout << (dejaView.DodajDvoranu(d2) ?
        "Uspjesno dodavanje dvorane d2." :
        "Bezuspjesno dodavanje dvorane d2.") << endl;;

    Datum datum1(2, 3, 1966), datum2(2, 4, 1969), datum3(1, 5, 1955);
    Uposlenik u1("Jonah", "Hillside", datum1, 1, "prodavac ulaznica", "Banja Luka
", "jonahillside@gmail.com");
    Uposlenik u2("Dwayne", "'The Mountain' Johnson", datum2, 1, "domar", "Sarajev
o", "johnrocksbigtime@gmail.com");
    Uposlenik u3("Tom", "Cruiser", datum3, 1, "vozac", "Brcko", "thomas_cruiser@g
mail.com");

    if (dejaView.DodajUposlenika(u1))
        cout << "Uspjesno dodan " << u1.GetIme() << " " << u1.GetPrezime() << end
l;
    if (dejaView.DodajUposlenika(u2))
        cout << "Uspjesno dodan " << u2.GetIme() << " " << u2.GetPrezime() << end
l;
    if (dejaView.DodajUposlenika(u3))
        cout << "Uspjesno dodan " << u3.GetIme() << " " << u3.GetPrezime() << end
l;
    dejaView.Ispis();
    cout << 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 << "Unesite odgovarajuci broj zadatka za testiranje: -->: ";
            cin >> izbor;
            cout << endl;
        } while (izbor < 1 || izbor > 4);
        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;
        default:break;
        }
        do {
            cout << "DA LI ZELITE NASTAVITI DALJE? (1/0): ";
            cin >> nastaviDalje;
        } while (nastaviDalje != 0 && nastaviDalje != 1);
    }
}
int main() {
    Menu();
    return 0;
}
```