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

/* ::NAPOMENA::
Radi jednostavnije izrade i lakseg testiranja zadataka komentirajte testni dio ko
da, 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)
// copy ctor = copy constructor (konstruktor kopije)
// move ctor = move constructor (konstruktor premjestanja)

//Z0.1
char* AlocirajIKopiraj(const char* tekst);

int Min(int broj1, int broj2) { return (broj1 <= broj2) ? broj1 : broj2; }
int Max(int broj1, int broj2) { return (broj1 >= broj2) ? broj1 : broj2; }

//Z0.2 :: Vratiti broj znamenki za dati broj
int IzracunajBrojZnamenki(int broj);

//Z0.3 :: Pretvoriti (int) u (char*). Obezbijediti da je 'broj' u opsegu [INT_MIN
, INT_MAX]
char* IntToStr(int broj);

bool PrijestupnaGodina(int godina) {
    return (((godina % 4 == 0) && (godina % 100 != 0)) ||
        (godina % 400 == 0));
}

//Z0.4 :: Vratiti broj dana za dati mjesec (Voditi racuna o prijestupnim godinama
)
int GetBrojDanaUMjesecu(int mjesec, int godina);

class Datum
{
private:
    int* _dan;
    int* _mjesec;
    int* _godina;
```

```
public:
    //Z1.1 :: Dflt. ctor
    Datum();

    //Z1.2 :: User-def. ctor
    Datum(int d, int m, int g);

    //Z1.3 :: Copy ctor
    Datum(const Datum& obj);

    //Z1.4 :: Move ctor
    Datum(Datum&& obj);

    //Z1.5 :: Operator =
    Datum& operator =(const Datum& obj);

    //Z1.6 :: Getteri
    int GetDan() const;
    int GetMjesec() const;
    int GetGodina() const;

    //Z1.7 :: Setteri
    void SetDan(int dan);
    void SetMjesec(int mjesec);
    void SetGodina(int godina);
    //Z1.8 :: dtor
    ~Datum();
};

//Z1.9 :: Ispisati datum
ostream& operator << (ostream& COUT, const Datum obj);

//Z1.10 :: Porediti dva datuma po vrijednostima atributa
bool operator == (const Datum& d1, const Datum& d2);

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

//Z1.11 :: Kreirati novi datum kao rezultat dodavanja varijable 'brojDana' na obj
ekat 'obj'
Datum operator + (Datum& obj, int brojDana);

//Z1.12 :: Provjeriti da li je 'd1' veci (noviji datum) od '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);

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

//Z1.13 Izracunati razliku (u danima) izmedju objekata 'd1' i 'd2'
int operator -(Datum& d1, Datum& d2);

class Clan {
    const int _clanId;
    char _korisnickoIme[30];
    char _lozinka[20];
    Datum* _datumRegistracije;
    bool* _spol;
    static int _clanIdCounter;
public:
    //Z2.0 :: Vratiti vrijednost statickog atributa _clanIdCounter
    static int GetCounter();

    //Z2.1 :: Dflt. ctor  [Postaviti _clanId na vrijednost statickog brojaca. Zat
im, uvecati brojac]
    Clan();

    //Z2.2 :: User-
def. ctor [Postaviti _clanId na vrijednost statickog brojaca. Zatim, uvecati broj
ac]
    Clan(const char* korisnickoIme, const char* lozinka, Datum datumReg, bool spo
l);

    //Z2.3 :: Copy ctor [kopirati obj._clanId u _clanId]
    Clan(const Clan& obj);

    //Z2.4 :: Move ctor [kopirati obj._clanId u _clanId]
    Clan(Clan&& obj);

    //Z2.5 :: operator dodjele
    Clan& operator = (const Clan& obj);

    //Z2.6 :: Getteri
    const char* GetKorisnickoIme() const;
    const char* GetLozinka() const;
    Datum GetDatumPrijave() const;
    bool GetSpol() const;

    //Z2.7 :: Setteri
```

```
    void SetKorisnickoIme(const char* korisnickoIme);
    void SetLozinka(const char* lozinka);
    void SetDatumRegistracije(Datum datumRegistracije);
    void SetSpol(bool spol);

    //Z2.8 :: dtor
    ~Clan();
};
int Clan::_clanIdCounter = 1; // Inicijalizacija statickog atributa

//Z2.9 :: Ispisati podatke o clanu
ostream& operator <<(ostream& COUT, const Clan& clan);

//Z2.10 :: operator == [Porediti clanove 'c1' i 'c2' po korisnickom imenu]
bool operator ==(const Clan& c1, const Clan& c2);

class Post {
    char* _postId;
    char* _korisnickoIme; //_korisnickoIme clana foruma koji je objavio post
    Datum _datumObjavljivanja;
    char* _sadrzaj;
    static int _postIdCounter;
public:
    //Z3.0 :: Vratiti staticki brojac _postIdCounter
    static int GetCounter();

    //Iskoristiti funkciju IntToStr za pretvaranje trenutne vrijednosti statickog
 atributa '_postIdCounter' u dinamicki niz karaktera
    //Povecati vrijednost '_postIdCounter'
    static char* GetNextPostId();

    //Z3.1 :: Postaviti sve atribute na dflt. vrijednosti
    Post();

    //Z3.2 :: Za inicijalizaciju _postId iskoristiti staticku funkciju GetNextPos
tId
    Post(const char* korisnickoIme, Datum datumO, const char* sadrzaj);

    //Z3.3 :: Inicijalizirati '_postId' na osnovu 'obj._postId'
    Post(const Post& obj);

    //Z3.4 :: Move ctor
    Post(Post&& obj);

    //Z3.5 :: operator dodjele
```

```
    Post& operator = (const Post& obj);

    //Z3.6 :: Getteri
    char* GetKorisnickoIme() const;
    Datum GetDatumObjavljivanja() const;
    char* GetSadrzaj() const;

    //Z3.7 :: Setteri
    //Settovati '_postId' pomocu staticke funkcije
    void SetNewPostId();

    void SetKorisnickoIme(const char* korisnickoIme);
    void SetDatumObjavljivanja(Datum d);
    void SetSadrzaj(const char* sadrzaj);

    //Z3.8 :: dtor
    ~Post();
};
int Post::_postIdCounter = 1000; // Inicijalizacija statickog atributa

//Z3.9 :: Ispisati podatke o postu
ostream& operator <<(ostream& COUT, const Post& p);
const int maxBrojPostova = 100;
class Sekcija {
    char* _naziv;
    char* _kratakOpis;
    int _trenutnoPostova;
    Post* _postovi[maxBrojPostova] = { nullptr };
public:
    //Z4.1 :: Dflt. ctor
    Sekcija();

    //Z4.2 :: User-def. ctor
    Sekcija(const char* naziv, const char* kratakOpis);

    //Z4.3 :: Copy ctor
    Sekcija(const Sekcija& obj);

    //Z4.4 :: Move ctor
    Sekcija(Sekcija&& obj);

    //Z4.5 :: operator dodjele
    Sekcija& operator = (const Sekcija& obj);

    //Z4.6 :: Getteri
```

```
    char* GetNaziv() const;
    char* GetKratakOpis() const;
    Post GetPostAtI(int index) const;

    //Z4.7 :: Setteri
    void SetNaziv(const char* naziv);
    void SetKratakOpis(const char* kratakOpis);

    //Z4.8 :: operator +=
    //Dodati novi post u niz pokazivaca
    //Onemoguciti dodavanje u slucaju da je popunjen niz pokazivaca
    bool operator +=(Post& p);

    //Z4.9 :: dtor
    ~Sekcija();
};
//Z4.10 :: Ispisati podatke o sekciji [ukljucujuci i postove]
ostream& operator << (ostream& COUT, const Sekcija& obj);

const int maxBrojSekcija = 20;
class Forum {
    char* _naziv;
    int _trenutnoSekcija;
    Sekcija _sekcije[maxBrojSekcija];
    int _maxClanova;
    Clan* _clanovi;
    int _trenutnoClanova;
public:
    //Z5.1 :: Dflt. ctor
    Forum();

    //Z5.2 :: User-def. ctor
    Forum(const char* naziv, int maxClanova);

    //Z5.3 :: Copy ctor
    Forum(const Forum& obj);

    //Z5.4 :: Move ctor
    Forum(Forum&& obj);

    //Z5.5 :: Getteri
    int GetTrenutnoSekcija() const;
    Sekcija GetSekcijaAtI(int index) const;
    int GetBrojClanova() const;
    int MaxBrojClanova() const;
```

```
    Clan GetClanAtI(int index) const;


    //Z5.6 :: Setteri
    void SetNaziv(const char* naziv);

    //Z5.7 :: Setter za _maxClanova
    /*
        Osigurati da je (noviMaxBrojClanova> _maxBrojClanova)
        Kreirati novi niz velicine (noviMaxBrojClanova)
        Kopirati sadrzaj iz starog niza u novi niz
    */
    void SetMaxClanova(int noviMaxBrojClanova);

    //Z5.8 Operator += (dodavanje nove sekcije)
    bool operator += (const Sekcija sekcija);

    //Z5.9 :: operator += (dodavanje novog clana)
    //Ukoliko brojac dosegne vrijednost '_maxClanova', uraditi prosirivanje niza
za 10 koristenjem metode 'SetMaxClanova'
    void operator += (const Clan clan);

    //Z5.10 :: dtor
    ~Forum();
};
//Z5.11 :: Ispisati podatke o forumu, ispisati sekcije [zajedno sa postovima] te
korisnicka imena forumasa [clanova]
ostream& operator <<(ostream& COUT, const Forum& f);

void Zadatak1() {
    int broj = 56511;
    cout << "Pretvaranje broja (56511) u str] : " << endl;
    char* stringBroj = IntToStr(broj);
    cout << stringBroj << endl;
    delete[] stringBroj;
    stringBroj = nullptr;
    cout << "Sve prijestupne godine izmedju [1900-2021]: " << endl;
    for (size_t i = 1900; i <= 2021; i++)
        if (PrijestupnaGodina(i))
            cout << i << ", ";
    cout << endl;
    Datum starWarsDay; //dflt. ctor
    starWarsDay.SetDan(4);
    starWarsDay.SetMjesec(5);
    starWarsDay.SetGodina(2021);
```

```
    cout << "Star Wars day: " << starWarsDay << endl; // operator <<

    Datum worldUfoDay(starWarsDay.GetDan() - 3, starWarsDay.GetMjesec() + 2, star
WarsDay.GetGodina()); //user-def. ctor
    cout << "World Ufo day: " << worldUfoDay << endl;

    Datum laborDay(starWarsDay); //copy ctor
    laborDay.SetDan(1);
    cout << "Labor day (BiH): " << laborDay << endl;

    Datum victoryDay(move(laborDay)); //Move ctor
    victoryDay.SetDan(9);
    cout << "Victory day (BiH): " << victoryDay << endl;

    Datum juneSolstice(21, 6, 2021), juneSolstice_copy;
    juneSolstice_copy = juneSolstice;
    cout << "June Solstice (BiH): " << juneSolstice << endl;
    cout << "Dealokacija ..." << endl;

    Datum datumi[] = { Datum(1,2,2021), Datum(31,12, 2020), Datum(31, 12, 2021) }
;
    cout << "Razlika u danima: --->" << endl;
    cout << "Razlika izmedju: " << datumi[0] << " i " << datumi[1] << " je " << d
atumi[0] - datumi[1] << endl; // operator -
    cout << "Razlika izmedju: " << datumi[0] << " i " << datumi[2] << " je " << d
atumi[0] - datumi[2] << endl; // operator -
    cout << "Razlika izmedju: " << datumi[1] << " i " << datumi[2] << " je " << d
atumi[1] - datumi[2] << endl; // operator -

    //Testiranje operatora +
    Datum someDatum(5, 5, 2025);
    cout << "Test datum: " << someDatum << endl;
    cout << someDatum << " + 30 dana  = " << someDatum + 30 << endl; // operator
+
    cout << "Dealokacija..." << endl;
}

void Zadatak2() {

    Clan almightyBruce;
    almightyBruce.SetKorisnickoIme("almightyBruce");
    almightyBruce.SetDatumRegistracije(Datum(1, 1, 2021));
    almightyBruce.SetSpol(0);
    almightyBruce.SetSpol(1);
    almightyBruce.SetLozinka("its'Goooood");
```

```
    cout << almightyBruce << endl;

    Clan crazyMage("CrazyMage", "PA$$w0rd", Datum(3, 12, 2019), 1);
    Clan copyCrazyMage(crazyMage);
    cout << copyCrazyMage << endl;

    Clan azermyth("Azermyth", "azerpass", Datum(1, 4, 2020), 1);
    Clan noviAzer(move(azermyth));
    cout << noviAzer << endl;
    cout << "Testiranje operatora '==' " << endl;
    cout << (crazyMage == copyCrazyMage ? "Isti clan!" : "Razlici clanovi!") << e
ndl;

    Clan aceVentura;
    aceVentura = noviAzer;
    aceVentura.SetKorisnickoIme("8Ventura");
    cout << aceVentura << endl;
    cout << "Dealokacija..." << endl;
}
void Zadatak3() {
    Post p1;
    p1.SetNewPostId();
    p1.SetKorisnickoIme("Neo");
    p1.SetDatumObjavljivanja(Datum(5, 5, 2021));
    p1.SetSadrzaj("Izasao sam iz matrice. Osjecaj je prelijep...");
    cout << p1 << endl;

    Post p2("Trinity", Datum(5, 5, 2021), " Kolega @Neo, you don't say.");
    Post copyp2(p2);
    cout << copyp2 << endl;

    Post p3("Ementaler", Datum(6, 5, 2021), "Pozdrav ljudi. Ovdje Igor sa Hcl-
a...");
    Post pr3new(move(p3));
    cout << pr3new << endl;

    Post p4;
    p4 = pr3new;
    p4.SetNewPostId();
    p4.SetKorisnickoIme("Agent Smith");
    p4.SetSadrzaj("Dragi kolega @Neo, pripremite se da vas dealociram.");
    cout << p4 << endl;
    cout << "Dealokacija..." << endl;
}
```

```
void Zadatak4() {
    Sekcija letNaMars("Let na mars, all about...", "Neki opis...");
    Post p1("bad_karma13", Datum(2, 3, 2020), "Ispucao je losu srecu na Cybertruc
ku.. Ovo uspijeva 100%");
    Post p2("monkey_see_monkey_do", Datum(3, 3, 2020), "Kad ono uzlijece Elon sa
svojima? xD");
    Post p3("cerealKillerHoho", Datum(3, 3, 2020), "Teraformiranje Marsa ce se po
kazati kao prevelik zalogaj za nasu generaciju...");
    Post p4("dr_Michio_Kaku", Datum(3, 3, 2020), "Ovo je prvi korak u kolonizacij
i Suncevog sistema...");
    letNaMars += p1;
    letNaMars += p2;
    letNaMars += p3;
    Sekcija mars2(letNaMars);
    mars2 += p4;

    Sekcija mars3(move(mars2));
    Post p5("superSonic", Datum(3, 3, 2020), "Zelimo novo gostovanje g.Muska kod
Joe Rogena!");
    mars3 += p5;
    Sekcija mars4;
    mars4 = mars3;
    cout << mars4 << endl;
    cout << "Dealokacija..." << endl;
}

void Zadatak5() {
    Forum nebula("Nebula:: forum o fizici i metafizici", 10);
    Clan arwen_dor("arwenix", "L0trI$L1fe", Datum(11, 1, 2021), 0);
    Clan thomasAnderson("neo", "one", Datum(12, 1, 2021), 1);
    Clan rickC_137("rickestRick", "wabalubadubdub", Datum(3, 3, 2021), 1);

    //Dodavanje clanova preko operatora +=
    nebula += arwen_dor;
    nebula += thomasAnderson;
    nebula += rickC_137;
    //
    Sekcija newAge("New Age", "Sta predstavlja New Age?");
    Post p1("arwenix", Datum(3, 3, 2020), "Postoji niz proturijecnih definicija o
 novom fenomenu ...");
    Post p2("neo", Datum(4, 3, 2020), "Nova religija? Ili ipak samo nova paradigm
a? ...");
    Post p3("rickestRick", Datum(5, 3, 2020), "Ovisi od konteksta u kojem se poja
vljuje");
    newAge += p1; // dodavanje posta
```

```
    newAge += p2; // dodavanje posta
    newAge += p3; // dodavanje posta
    //
    Sekcija telepatija("Telepatija i telekineza", "Parapsiholoski fenomeni");
    Post p4("arwenix", Datum(6, 3, 2020), "Na ovom podrucju najvise se proslavio
Uri Geller ...");
    Post p5("neo", Datum(7, 3, 2020), "Medju poznatije slucajeve ubraja se i Nina
 Kulagina...");
    telepatija += p4; // dodavanje posta
    telepatija += p5; // dodavanje posta
    //
    nebula += newAge; // dodavanje sekcije
    nebula += telepatija; // dodavanje sekcije
    //
    Forum copy_of_nebula(nebula);
    Forum nebula_prime(move(copy_of_nebula));
    cout << nebula_prime;
    cout << "Dealokacija..." << endl;
}

int 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);
    }
    return 0;
}

int main() {
    Menu();
    return 0;
}
```