1. ZADATAK

PRII – (G1) – NPP18/19

Izvršiti definiciju funkcija na način koji odgovara opisu (komentarima) datom neposredno uz pozive ili nazive funkcija. Možete dati komentar na bilo koju liniju code-a koju smatrate da bi trebalo unaprijediti ili da će eventualno uzrokovati grešku prilikom kompajliranja. Također, možete dodati dodatne funkcije koje će vam olakšati implementaciju programa.

```
#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. NAZIVI FUNKCIJA, TE BROJ I TIP PARAMETARA MORAJU BITI IDENTICNI ONIMA
KOJI SU KORISTENI U TESTNOM CODE-U,\n"
"\tOSIM U SLUCAJU DA POSTOJI ADEKVATAN RAZLOG ZA NJIHOVU MODIFIKACIJU.
OSTALE\n"
"\tPOMOCNE 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.ZA TESTIRANJE BUDITE SLOBODNI DODATI TESTNIH PODATAKA (POZIVA METODA)
KOLIKO GOD SMATRATE DA JE POTREBNO!\n"

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

const char* crt = "\n-------------------------------------------\n";
enum Predmet { UIT, PRI, PRII, PRIII, RSI, RSII };
const int brojRjesenja = 6;
const char* NIJE_VALIDNA = "<VRIJEDNOST_NIJE_VALIDNA>";

char* GetNizKaraktera(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, int max = 10>
class Kolekcija {
    T1 _elementi1[max];
    T2 _elementi2[max];
    int  * _trenutno;
public:
    Kolekcija() {
        _trenutno = nullptr;
    }
    ~Kolekcija(){
        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;
    }
};
/*na odredjeno pitanje kandidat je duzan postaviti vise rjesenja od kojih ce
svako biti ocijenjeno*/
class Pitanje {
    char* _sadrzaj;
    //int se odnosi na ocjenu u opsegu  1 - 5, a Datum na datum kada je
odgovor/rjesenje ocijenjeno
    Kolekcija<int, Datum, brojRjesenja>* _ocjeneRjesenja;
public:
    Pitanje(const char* sadrzaj) {
        _sadrzaj = GetNizKaraktera(sadrzaj);
        _ocjeneRjesenja = nullptr;
    }
    ~Pitanje() {
        delete[] _sadrzaj; _sadrzaj = nullptr;
        delete _ocjeneRjesenja; _ocjeneRjesenja = nullptr;
    }
    char* GetSadrzaj() { return _sadrzaj; }
    Kolekcija<int, Datum, brojRjesenja>& GetOcjene() { return
*_ocjeneRjesenja; }
};
class Ispit {
    Predmet _predmet;
    vector<Pitanje*> _pitanjaOdgovori;
public:
    Ispit(Predmet predmet = UIT) {
        _predmet = predmet;
    }
    ~Ispit() {
        for (size_t i = 0; i < _pitanjaOdgovori.size(); i++) {
            delete _pitanjaOdgovori[i];
            _pitanjaOdgovori[i] = nullptr;
        }
    }
    vector<Pitanje*>& GetPitanjaOdgovore() { return _pitanjaOdgovori; }
    Predmet GetPredmet() { return _predmet; }
    friend ostream& operator<< (ostream& COUT, const Ispit& obj) {
```

```
        COUT << obj._predmet << endl;
        for (size_t i = 0; i < obj._pitanjaOdgovori.size(); i++)
            COUT << *obj._pitanjaOdgovori[i];
        return COUT;
    }
};
class Korisnik {
    char* _imePrezime;
    string _emailAdresa;
    string _lozinka;
public:
    Korisnik(const char* imePrezime, string emailAdresa, string lozinka)
    {
        _imePrezime = GetNizKaraktera(imePrezime);
        _emailAdresa = emailAdresa;
        _lozinka = "";//<== na opisani nacin inicijalizovati lozinku
    }
    virtual ~Korisnik() { delete[] _imePrezime; _imePrezime = nullptr; }
    string GetEmail() { return _emailAdresa; }
    string GetLozinka() { return _lozinka; }
    char* GetImePrezime() { return _imePrezime; }
    virtual void Info() {}
};
class Kandidat {
    vector<Ispit> _polozeniPredmeti;
public:
    Kandidat(const char* imePrezime, string emailAdresa, string lozinka) {
    }
    ~Kandidat() {
        cout << crt << "DESTRUKTOR -> Kandidat" << crt;
    }
    friend ostream& operator<< (ostream& COUT, Kandidat& obj) {
        COUT << obj.GetImePrezime() << " " << obj.GetEmail() << " " <<
obj.GetLozinka() << endl;
        for (size_t i = 0; i < obj._polozeniPredmeti.size(); i++)
            COUT << obj._polozeniPredmeti[i];
        return COUT;
    }
    vector<Ispit>& GetPolozeniPredmeti() { return _polozeniPredmeti; }
};
const char* GetOdgovorNaPrvoPitanje() {
    cout << "Pitanje -> Pojasnite prednosti i nedostatke visestrukog
nasljedjivanja.\n";
    return "Odgovor -> OVDJE UNESITE VAS ODGOVOR";
}
const char* GetOdgovorNaDrugoPitanje() {
    cout << "Pitanje -> Ukratko opisite znacaj i nacin koristanje pametnih
pokazivaca?\n";
    return "Odgovor -> OVDJE UNESITE VAS ODGOVOR";
}
void main() {

    cout << PORUKA;
    cin.get();

    cout << GetOdgovorNaPrvoPitanje() << endl;
    cin.get();
    cout << GetOdgovorNaDrugoPitanje() << endl;
    cin.get();

    Datum   datum19062020(19, 6, 2020),
        datum20062020(20, 6, 2020),
        datum30062020(30, 6, 2020),
        datum05072020(5, 7, 2020);
```

```
    int kolekcijaTestSize = 10;

    Kolekcija<int, int> kolekcija1;
    for (int i = 0; i < kolekcijaTestSize; i++)
        kolekcija1.AddElement(i, i);

    cout << kolekcija1 << endl;

    try {
        /*metoda AddElement baca izuzetak u slucaju da se pokusa
        dodati vise od maksimalnog broja elemenata*/
        kolekcija1.AddElement(11, 11);
    }
    catch (exception& err) {
        cout << crt << "Greska -> " << err.what() << crt;
    }
    cout << kolekcija1 << crt;


    kolekcija1.RemoveAt(2);
    /*uklanja par (T1 i T2) iz kolekcije koji se nalazi na lokaciji sa
proslijedjenim indeksom.
    nakon uklanjanja vrijednosti onemoguciti pojavu praznog prostora unutar
kolekcije tj.
    pomjeriti sve elemente koji se nalaze nakon proslijedjene lokacije za
jedno mjesto unazad
    npr. ako unutar kolekcije postoje elementi
    0 0
    1 1
    2 2
    3 3
    nakon uklanjanja vrijednosti na lokaciji 1, sadrzaj kolekcije ce biti
sljedeci
    0 0
    2 2
    3 3
    */

    cout << kolekcija1 << crt;

    kolekcija1.AddElement(9, 9, 2);
    /*funkciji AddElement se, kao treci parametar, moze proslijediti i
lokacija na koju se dodaju
    nove vrijednosti pri cemu treba zadrzati postojece vrijednosti pomjerene
za jedno mjesto unaprijed
    u odnosu na definisanu lokaciju npr. ako unutar kolekcije postoje
elementi
    0 0
    1 1
    2 2
    3 3
    nakon dodavanja vrijednosti 9 i 9 na lokaciju 1, sadrzaj kolekcije ce
biti sljedeci
    0 0
    9 9
    1 1
    2 2
    3 3
    */

    cout << kolekcija1 << crt;

    Kolekcija<int, int> kolekcija2 = kolekcija1;
```

```
    cout << kolekcija1 << crt;

    //na osnovu vrijednosti T1 mijenja vrijednost T2.
    kolekcija1[9] = 2;
    /* npr.ako unutar kolekcije postoje elementi:
    0 0
    9 9
    1 1
    2 2
    3 3
    nakon promjene vrijednosti sadrzaj kolekcije ce biti sljedeci
    0 0
    9 2
    1 1
    2 2
    3 3
    */

    Pitanje sortiranjeNiza("Navedite algoritme za sortiranje clanova niza."),
        dinamickaMemorija("Navedite pristupe za upravljanje dinamickom
memorijom."),
        visenitnoProgramiranje("Na koji se sve nacin moze koristiti veci broj
niti tokom izvrsenja programa."),
        regex("Navedite par primjera regex validacije podataka.");

    /*svako pitanje moze imati vise ocjena tj. razlicita rjesenja/odgovori se
mogu postaviti u vise navrata.Drugim rijecima, ocjena, rjesenje i odgovor se
mogu posmatrati kao sinonimi.
        -   razmak izmedju postavljanja dva rjesenja mora biti najmanje 3
dana
        -   nije dozvoljeno dodati ocjenu sa ranijim datumom u odnosu na vec
evidentirane (bez obzira sto je stariji od 3 dana)
    */
    if (sortiranjeNiza.AddOcjena(1, datum19062020))
        cout << "Ocjena evidentirana!" << endl;
    if (!sortiranjeNiza.AddOcjena(5, datum20062020))
        cout << "Ocjena NIJE evidentirana!" << endl;
    if (sortiranjeNiza.AddOcjena(5, datum30062020))
        cout << "Ocjena evidentirana!" << endl;

    // ispisuje sadrzaj/tekst pitanja, ocjene (zajedno sa datumom) i
prosjecnu ocjenu za sve odgovore/rjesenja
    // ukoliko pitanje nema niti jednu ocjenu prosjecna treba biti 0
    cout << sortiranjeNiza << endl;

    if (ValidirajLozinku("john4Do*e"))
        cout << "OK" << crt;
    if (!ValidirajLozinku("john4Doe"))
        cout << "Specijalni znak?" << crt;
    if (!ValidirajLozinku("jo*4Da"))
        cout << "7 znakova?" << crt;
    if (!ValidirajLozinku("4jo-hnoe"))
        cout << "Veliko slovo?" << crt;
    if (ValidirajLozinku("@john2Doe"))
        cout << "OK" << crt;

    /*
    za autentifikaciju svaki korisnik mora posjedovati lozinku koja sadrzi:
    -   najmanje 7 znakova
    -   velika i mala slova
    -   najmanje jedan broj
    -   najmanje jedan specijalni znak
```

```
    za provjeru validnosti lozinke koristiti globalnu funkciju
ValidirajLozinku, a unutar nje regex metode.
    validacija lozinke se vrsi unutar konstruktora klase Korisnik, a u
slucaju da nije validna
    postaviti je na podrazumijevanu vrijednost: <VRIJEDNOST_NIJE_VALIDNA>
    */

    Korisnik* jasmin = new Kandidat("Jasmin Azemovic", "jasmin@kursevi.ba",
"j@sm1N*");
    Korisnik* adel = new Kandidat("Adel Handzic", "adel@edu.kursevi.ba",
"4Ade1*H");
    Korisnik* lozinkaNijeValidna = new Kandidat("John Doe",
"john.doe@google.com", "johndoe");

    /*
    svi odgovori na nivou jednog predmeta (PRI, PRII... ) se evidentiraju
unutar istog objekta tipa Ispit tj. pripadajuceg objekta tipa Pitanje,
    tom prilikom onemoguciti:
    - dodavanje istih (moraju biti identicne vrijednosti svih atributa)
odgovora na nivou jednog predmeta,
    - dodavanje odgovora za viši predmet ako prethodni predmet nema
evidentirana najmanje 3 pitanja ili nema prosjecnu ocjenu svih pitanja vecu
od 3.5
    (onemoguciti dodavanje pitanja za PRII ako ne postoje najmanje 3 pitanja
za predmet PRI ili njihov prosjek nije veci od 3.5)
    funkcija vraca true ili false u zavisnosti od (ne)uspjesnost izvrsenja
    */

    //doraditi klase da nacin da omoguce izvrsenje naredne linije koda
    Kandidat* jasminPolaznik = dynamic_cast<Kandidat*>(jasmin);

    if (jasminPolaznik != nullptr) {
        if (jasminPolaznik->AddPitanje(PRI, dinamickaMemorija))
            cout << "Pitanje uspjesno dodano!" << crt;
        //ne treba dodati visenitnoProgramiranje jer ne postoje evidentirana
3 pitanja za predmet PRI
        if (!jasminPolaznik->AddPitanje(PRII, visenitnoProgramiranje))
            cout << "Pitanje NIJE uspjesno dodano!" << crt;
        if (jasminPolaznik->AddPitanje(PRI, visenitnoProgramiranje))
            cout << "Pitanje uspjesno dodano!" << crt;
        if (jasminPolaznik->AddPitanje(PRI, regex))
            cout << "Pitanje uspjesno dodano!" << crt;
        if (jasminPolaznik->AddPitanje(PRI, sortiranjeNiza))
            cout << "Pitanje uspjesno dodano!" << crt;
        //ne treba dodati sortiranjeNiza jer je vec dodana za predmet PRI
        if (!jasminPolaznik->AddPitanje(PRI, sortiranjeNiza))
            cout << "Pitanje NIJE uspjesno dodano!" << crt;

        //ispisuje sve dostupne podatke o kandidatu
        cout << *jasminPolaznik << crt;
    }
    /*nakon evidentiranja ocjene na bilo kojem odgovoru, kandidatu se salje
email sa porukom:

    FROM:info@kursevi.ba
    TO: emailKorisnika

    Postovani ime i prezime, evidentirana vam je ocjena X za odgovor na
pitanje Y. Dosadasnji uspjeh (prosjek ocjena)
    za pitanje Y iznosi F, a ukupni uspjeh (prosjek ocjena) na svim
predmetima iznosi Z.
    Pozdrav.

    EDUTeam.
```

```
    slanje email poruka implemenitrati koristeci zasebne thread-ove.
    */

    //osigurati da se u narednim linijama poziva i destruktor klase Kandidat
    delete jasmin;
    delete adel;
    delete lozinkaNijeValidna;

    cin.get();
    system("pause>0");
}
```