

1. ZADATAK

PRII – G1

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 ili metode koje će vam olakšati implementaciju programa.

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

const char* PORUKA = "\n-----\n"
-----\n"
"0. PROVJERITE DA LI PREUZETI ZADACI PRIPADAJU VASOJ GRUPI (G1/G2)\n"
"1. SVE KLASJE 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 METODA (SVE ISTO VAZI I ZA FUNKCIJE), 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 METODE MOZETE IMENOVATI I DODAVATI PO ZELJI.\n"
"5. IZUZETAK BACITE SAMO U METODAMA U KOJIMA JE TO NAZNACENO.\n"
"6. SVE METODE POZVANE U MAIN-U ZADATKA TREBAJU POSTOJATI. UKOLIKO
NISTE ZADOVOLJNI IMPLEMENTACIJOM "
"POTREBNO JE DA IMPLEMENTIRATE BAREM TIJELO TIH METODA (METODA MOZE
BITI PRAZNA), "
"A AKO METODA TREBA VRATITI NEKI PODATAK ONDA MOZETE VRATITI BILO KOJU
TJ. ZELJENU VRIJEDNOST ZAHTIJEVANOG TIPA.!\n"
"7. NA KRAJU ISPITA SVOJE RJESENJE KOPIRAJTE U .DOCX FAJL (IMENOVAN
BROJEM INDEKSA npr. IB150051.docx)!\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 MAIN-U, BUDITE SLOBODNI DODATI TESTNIH
PODATAKA (POZIVA METODA) KOLIKO GOD SMATRATE DA JE POTREBNO!\n"
"12.ZA IZRADU ISPITNOG RJESENJA KORISTITI VISUAL STUDIO 2022 I
RJESENJE TESTIRAJTE U OBA MODA (F5 i Ctrl+F5)!\n"
"13.NA KRAJU ISPITA PROVJERITE DA LI STE RJEŠENJE KOPIRALI U ADEKVATAN
FOLDER NA FTP SERVERU\n"
"-----\n"
-----\n";

const char* crt = "\n-----\n";
enum Drzava { ENGLJESKA, SPANIJA, HOLANDIJA, FRANCUSKA,
BOSNA_I_HERCEGOVINA };

char* GetNizKaraktera(const char* sadrzaj, bool dealociraj = false) {
    if (sadrzaj == nullptr) return nullptr;
    int vel = strlen(sadrzaj) + 1;
    char* temp = new char[vel];
    strcpy_s(temp, vel, sadrzaj);
    if (dealociraj)
        delete[] sadrzaj;
    return temp;
}
```

```

}

template<class T1, class T2, int max>
class Kolekcija {
    T1 _elementi1[max];
    T2 _elementi2[max];
    int* _trenutno;
public:
    ~Kolekcija() {
        delete _trenutno; _trenutno = nullptr;
    }
    T1& getElement1(int lokacija){ return _elementi1[lokacija]; }
    T2& getElement2(int lokacija){ return _elementi2[lokacija]; }
    int getTrenutno() const { return *_trenutno; }
    friend ostream& operator<< (ostream& COUT, Kolekcija& obj) {
        for (size_t i = 0; i < obj._trenutno; i++)
            COUT << obj.getElement1(i) << " " << obj.getElement2(i) <<
endl;
        return COUT;
    }
};

class Vrijeme {
    int* _sat, * _minuta, * _sekunda;
public:
    Vrijeme(int sat = 10, int minuta = 0, int sekunda = 0) {
        _sat = new int(sat);
        _minuta = new int(minuta);
        _sekunda = new int(sekunda);
    }
    ~Vrijeme() {
        delete _sat; _sat = nullptr;
        delete _minuta; _minuta = nullptr;
        delete _sekunda; _sekunda = nullptr;
    }
    friend ostream& operator<< (ostream& COUT, const Vrijeme& obj) {
        COUT << obj._sat << ":" << obj._minuta << ":" << obj._sekunda;
        return COUT;
    }
};

class Pogodak {
    Vrijeme * _vrijemePogotka;
    char* _napomena;
public:
    Pogodak(Vrijeme vrijeme, const char* napomena) {
        _napomena = GetNizKaraktera(napomena);
        _vrijemePogotka = new Vrijeme(vrijeme);
    }
    ~Pogodak(){
        delete[] _napomena; _napomena = nullptr;
        delete _vrijemePogotka; _vrijemePogotka = nullptr;
    }
    Vrijeme GetVrijemePogotka() { return *_vrijemePogotka; }
    char* GetNapomena() { return _napomena; }
    friend ostream& operator<< (ostream& COUT, const Pogodak& obj) {
        COUT << *obj._vrijemePogotka << " -> " << obj._napomena;
        return COUT;
    }
};

```

```

    }
};

class Igrac {
    static int _id;
    char* _ID; // za inicijalizaciju _ID-a iskoristiti funkciju
    GenerisiID tj. atributa _imePrezime (inicijala imena i prezimena) i
    statickog clana _id
    char* _imePrezime;
    vector<Pogodak*> _pogoci;
public:
    Igrac(const char* imePrezime){
        _imePrezime = GetNizKaraktera(imePrezime);
    }
    ~Igrac() {
        delete[] _ID; _ID = nullptr;
        delete[] _imePrezime; _imePrezime = nullptr;
        for (size_t i = 0; i < _pogoci.size(); i++) {
            delete _pogoci[i];
            _pogoci[i] = nullptr;
        }
    }
    char* GetImePrezime() { return _imePrezime; }
    char* GetID() { return _ID; }
    vector<Pogodak*>& GetPogoci() { return _pogoci; }

    friend ostream& operator<< (ostream& COUT, Igrac& obj) {
        COUT << *obj._ID << " -> " << obj._imePrezime;
        for (size_t i = 0; i < obj._pogoci.size(); i++)
            cout << *obj._pogoci[i] << endl;
        return COUT;
    }
};

class Reprezentacija {
    Drzava _drzava;
    vector<Igrac> _igraci;
public:
    Reprezentacija(Drzava drzava = BOSNA_I_HERCEGOVINA) {
        _drzava = drzava;
    }
    Drzava GetDrzava() { return _drzava; }
    vector<Igrac>& GetIgraci() { return _igraci; }
};

class Prventstvo {
    Kolekcija<Reprezentacija, Reprezentacija, 20> _utakmice;
public:
    Kolekcija<Reprezentacija, Reprezentacija, 20>& GetUtakmice() {
        return _utakmice; }
};

const char* GetOdgovorNaPrvoPitanje() {
    cout << "Pitanje -> Pojasnite osnovne preduslove koji moraju biti
    ispunjeni da bi se realizovao polimorfizam (navesti kratki
    primjer)?\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();

    /*
    Globalna funkcija GenerisiID vraca ID igraca na osnovu vrijednosti
    prosljedjenih parametara. Funkcija generise
    ID sa maksimalno 4 cifre, ne racunajuci inicijale tj. prva slova
    imena i prezimena.
    Podrazumijeva se da ce uvijek biti prosljedjene validne
    vrijednosti parametara. Ukoliko se prosljede
    mala pocetna slova imena ili prezimena, potrebno ih je
    konvertovati u velika.
    U okviru generisanog ID-a, prosljedjena cjelobrojna vrijednost
    treba biti u obrnutom redoslijedu cifara.
    */
    cout << GenerisiID("Denis Music", 3) << endl;//treba vratiti
D000M3
    cout << GenerisiID("Jasmin Azemovic", 14) << endl;//treba vratiti
J00A41
    cout << GenerisiID("Goran Skondric",156) << endl;//treba vratiti
G0S651
    cout << GenerisiID("emina junuz",1798) << endl;//treba vratiti
EJ8971

    //Za validaciju ID-a koristiti funkciju ValidirajID koja treba,
    koristeci regex, osigurati postivanje osnovnih pravila
    //vezanih za format koja su definisana u prethodnom dijelu
    zadatka.
    if (ValidirajID("D000M3"))
        cout << "ID VALIDAN" << endl;
    if (ValidirajID("J00A41"))
        cout << "ID VALIDAN" << endl;
    if (!ValidirajID("G00S651"))
        cout << "ID NIJE VALIDAN" << endl;
    if (!ValidirajID("Ej8971"))
        cout << "ID NIJE VALIDAN" << endl;

    int kolekcijaTestSize = 9;

    Kolekcija<int, int, 10> kolekcija1;

    for (int i = 0; i < kolekcijaTestSize; i++)
        kolekcija1.AddElement(i, i);//dodaje vrijednosti u kolekciju
```

```
cout << kolekcija1 << crt;

/* metoda InsertAt treba da doda vrijednosti prvog i drugog
parametra na lokaciju koja je definisana trecim parametrom. Povratna
vrijednost metode
je objekat (pozivaoc metode, u konkretnom slucaju objekat
kolekcija1) u okviru koga su, na definisanu lokaciju, dodati
zahtijevani parametri.
Nakon izvršenja metode InsertAt, oba objekta, kolekcija1 i
kolekcija2, bi trebali posjedovati sljedeci sadrzaj:
10 10
0 0
1 1
2 2
* ....
*/
Kolekcija<int, int, 10> kolekcija2 = kolekcija1.InsertAt(10, 10,
0);
cout << kolekcija2 << crt;

/*Metoda RemoveRange prihvata lokacija OD i DO, te u tom opsegu
uklanja sve elemente iz kolekcije. U slucaju da zahtijevani opseg ne
postoji u kolekciji
metoda treba baciti izuzetak. Na kraju, metoda treba da vrati
pokazivac na novi objekat tipa kolekcija koji sadrzi samo uklonjene
elemente*/

Kolekcija<int, int, 10> kolekcija3 = kolekcija1.RemoveRange(1, 3);
cout << kolekcija3 << endl;

cout << kolekcija1 << crt;

/*kolekcija3 bi trebala sadrzavati sljedece elemente:
0 0
1 1
2 2

dok bi kolekcija1 trebala sadrzavati sljedece elemente:
10 10
3 3
4 4
.....
*/

kolekcija1 = kolekcija3;
cout << kolekcija1;

Vrijeme
prviPogodak201633(20, 16, 33),
drugiPogodak202319(20, 23, 19),
treciPogodak205108(20, 51, 8),
cetvrtiPogodak210654(21, 6, 54);

Igrac denis("Denis Music"), jasmin("Jasmin Azemovic"),
goran("Goran Skondric"), adil("Adil Joldic");

if (strcmp(denis.GetID(), "D000M1") == 0 &&
    strcmp(jasmin.GetID(), "J000A2") == 0)
```

```
        cout << "ID se uspjesno generise!" << endl;

    Pogodak prviPogodak(prviPogodak201633, "podaci o prvom pogotku"),
        drugiPogodak(drugiPogodak202319, "podaci o drugom pogotku"),
        treciPogodak(treciPogodak205108, "podaci o trecem pogotku"),
        cetvrtiPogodak(cetvrtiPogodak210654, "podaci o cetvrtom
pogotku");

    Reprezentacija BIH(BOSNA_I_HERCEGOVINA), ENG(ENGLESKA);
    BIH.AddIgrac(denis);
    BIH.AddIgrac(jasmin);
    ENG.AddIgrac(goran);
    ENG.AddIgrac(adil);

    try
    {
        BIH.AddIgrac(denis); //onemoguciti dodavanje istih igraca -
provjeravati ID
    }
    catch (exception& obj)
    {
        cout << obj.what();
    }

    Prventstvo euro2024;
    euro2024.AddUtakmicu(BIH, ENG);

    try
    {
        euro2024.AddUtakmicu(BIH, ENG); //onemoguciti ponovne susrete
drzava tokom istog prvenstva
    }
    catch (exception& obj)
    {
        cout << obj.what();
    }

    //omoguciti dodavanje pogotka po ID-u ili imenu i prezimenu
    if (euro2024.AddPogodak(BOSNA_I_HERCEGOVINA, ENGLESKA, "D000M1",
prviPogodak))
        cout << "Pogodak uspjesno dodat" << endl;
    //onemoguciti dodavanje istih pogodaka
    if (!euro2024.AddPogodak(BOSNA_I_HERCEGOVINA, ENGLESKA, "Denis
Music", prviPogodak))
        cout << "Pogodak NIJE uspjesno dodat" << endl;
    if (euro2024.AddPogodak(BOSNA_I_HERCEGOVINA, ENGLESKA, "J000A2",
drugiPogodak))
        cout << "Pogodak uspjesno dodat" << endl;
    if (euro2024.AddPogodak(BOSNA_I_HERCEGOVINA, ENGLESKA, "Jasmin
Azemovic", treciPogodak))
        cout << "Pogodak uspjesno dodat" << endl;
    if (euro2024.AddPogodak(BOSNA_I_HERCEGOVINA, ENGLESKA, "Goran
Skondric", cetvrtiPogodak))
        cout << "Pogodak uspjesno dodat" << endl;

    //nakon svakog evidentiranog pogotka, svim igracima te utakmice
(pod pretpostavkom da su validne email adrese sa ID-ovima igraca),
```

```
//u zasebnom thread-u, poslati email. u razmaku od 2 sekunde, sa
sljedećim sadržajem:
/*
    To: D000M1@euro2024.com
    From: info@euro2024.com
    Subject: Informacija

    Postovani,

    U 20:35:16 sati igrac Jasmin Azemovic je zabiljezio svoj 1
pogodak na ovoj utakmici.
    Trenutni rezultat je:

    BOSNA_I_HERCEGOVINA 2 : 0 ENGLESKA

    Puno sreće u nastavku susreta.
    Neka bolji tim pobijedi.
*/

//ispisuje detaljnije informacije o susretu, kako je navedeno u
narednom ispisu
cout << euro2024;

/*
-----
(3) BOSNA_I_HERCEGOVINA : ENGLESKA (1)
-----
Denis Music                Goran Skondric
Jasmin Azemovic
Jasmin Azemovic
-----
*/

//vraca sve igrace koji su na takmicenju postigli broj pogodaka
koji je veci ili jednak prosljedjenoj vrijednosti
vector<Igrac*> igraci = euro2024(2);
for (size_t i = 0; i < igraci.size(); i++)
    cout << igraci[i]->GetImePrezime() << endl;

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