UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET

Zadaća 3

Tehnike programiranja

Student: **Daris Mujkić**

Grupa: RI1-2b

Broj indeksa: 19413

Ocjena: 4.5

Potpis:

Potpis tutora:

Zadatak . 1.cpp

```
//TP 2022/2023: Zada?a 3, Zadatak 1
#include <iostream>
#include <cmath>
#include <stdexcept>
#include <vector>
#include <utility>
#include <functional>
#include <algorithm>

const double eps=0.0000001;

std::function<double(double)> LagrangeovaInterpolacija (std::vector<std::pair<
double,double>> cvorovi){
    for (int i=0;i<cvorovi.size();i++)
    {
        for (int j=i+1;j<cvorovi.size();j++)
        {
            if (std::fabs(cvorovi[i].first-cvorovi[j].first)<eps) throw std::
domain_error("Neispravni cvorovi");
        }
    }
    return [cvorovi](double x)
    {
        double rezultat=0;
        for (int i=0;i<cvorovi.size();i++)
        {
            double brojnik=1, nazivnik=1;
            for (int j=0;j<cvorovi.size();j++)
            {
                if (j!=i)
                {
                    brojnik=brojnik*(x-cvorovi[j].first);
                    nazivnik=nazivnik*(cvorovi[i].first-cvorovi[j].first);
                }
            }
            rezultat=rezultat+(cvorovi[i].second*brojnik/nazivnik);
        }
        return rezultat;
    };
}

double f(double x)
{
    return x*x+std::sin(x)+std::log(x+1);
}

std::function<double(double)> LagrangeovaInterpolacija (std::function<double(
double)>f, double Xmin, double Xmax, double deltaX)
{
    if (Xmin>Xmax || deltaX<=0) throw std::domain_error("Nekorektni parametri");
    int brojclanova=0;
    double rezultat = (Xmax-Xmin)/deltaX;
    if (rezultat==int(rezultat)) // djeljivi su
    {
```

```
        double izraz=Xmin;
        int i=0;
        while (izraz<=Xmax)
        {
            izraz=Xmin+i*deltaX;
            brojclanova++;
            i++;
        }
    }
    else
    {
        double izraz=Xmin;
        int i=0;
        while (izraz<Xmax)
        {
            izraz=Xmin+i*deltaX;
            brojclanova++;
            i++;
        }
    }
    std::vector<std::pair<double,double>> vektor;
    double prvi=Xmin;
    double drugi=0;
    for (int i=0;i<brojclanova-1;i++)
    {
        prvi=Xmin+i*deltaX;
        drugi=f(prvi);
        vektor.push_back(std::make_pair(prvi, drugi));
    }
    auto RJESENJE=LagrangeovaInterpolacija(vektor);
    return RJESENJE;
}

int main ()
{
    try {
    std::cout<<"Odaberite opciju (1 - unos cvorova, 2 - aproksimacija): "<<std::endl;
    int opcija;
    std::cin>>opcija;
    if (opcija==1)
    {
        std::cout<<"Unesite broj cvorova: "<<std::endl;
        int brcvorova;
        std::cin>>brcvorova;
        std::cout<<"Unesite cvorove kao parove x y: "<<std::endl;
        std::vector<std::pair<double,double>>cvor;
        for (int i=0;i<brcvorova;i++)
        {
            double x,y;
            std::cin>>x>>y;
            cvor.push_back(std::make_pair(x,y)); //probati emplace_back
        }

        double min=cvor[0].first;
        double max=cvor[0].first;
```

```
        for (int i=0;i<cvor.size();i++)
        {
            if (cvor[i].first>max) max=cvor[i].first;
            else if (cvor[i].first<min) min=cvor[i].first;
        }
        auto p=LagrangeovaInterpolacija(cvor);
        double argument;
        std::cout<<"Unesite argument (ili \"kraj\" za kraj): ";
        while (std::cin>>argument)
        {
            if (argument<min || argument >max) std::cout<<"f("<<argument<<") = "<<p(
argument)<<" [ekstrapolacija]";
            else std::cout<<"f("<<argument<<") = "<<p(argument);
            std::cout<<"\nUnesite argument (ili \"kraj\" za kraj): ";
        }
        return 0;
    }
    else if (opcija==2)
    {
        std::cout<<"Unesite krajeve intervala i korak: ";
        double kraj1, kraj2, korak;
        std::cin>>kraj1>>kraj2>>korak;
        auto p=LagrangeovaInterpolacija(f,kraj1,kraj2,korak);
        std::cout<<"Unesite argument (ili \"kraj\" za kraj): ";
        double argument;
        while (std::cin>>argument)
        {
            if (argument<kraj1 || argument>kraj2) std::cout<<"f("<<argument<<
") = "<<f(argument)<<" P("<<argument<<") = "<<p(argument)<<" [ekstrapolacija]";
            else std::cout<<"f("<<argument<<") = "<<f(argument)<<" P("<<
argument<<") = "<<p(argument);
            std::cout<<"\nUnesite argument (ili \"kraj\" za kraj): ";
        }
        return 0;
    }
    }
    catch(std::domain_error izuzetak)
    {
        std::cout<<izuzetak.what();
        return 0;
    }
        return 0;
}
```

Zadatak .

2.cpp

```
//TP 2022/2023: Zada?a 3, Zadatak 2
#include <iostream>
#include <cmath>
#include <tuple>
#include <vector>
#include <map>
#include <string>
#include <algorithm>
#include <set>
#include <stdexcept>

typedef std::map<std::string, std::vector<std::string>> Knjiga;
//KreirajIndeksPojmova
std::map<std::string, std::set<std::tuple<std::string, int, int>>>
 KreirajIndeksPojmova(const Knjiga &knjiga){
    std::map<std::string, std::set<std::tuple<std::string, int, int>>> indeks;
    int brojacStranica=0;
    for (const auto &poglavlje : knjiga)
    {
        const std::string &oznakaPoglavlja = poglavlje.first;
        const std::vector<std::string> &stranice = poglavlje.second;
        int brojacStranica=0; //DODANO
        for (int i=0;i<stranice.size();i++)
        {
            const std::string &sadrzajStranice = stranice[i];
            std::string rijec;
            int pozicija=0;
            for (char c : sadrzajStranice)
            {
                if (std::isalnum(c)) rijec.push_back(std::tolower(c));
                else
                {
                    if (!rijec.empty())
                    {
                        indeks[rijec].insert(std::make_tuple(
oznakaPoglavlja, brojacStranica+1, pozicija-rijec.length()));
                        rijec.clear();
                    }
                }
                pozicija++;
            }
            if (!rijec.empty()) indeks[rijec].insert(std::make_tuple(
oznakaPoglavlja, brojacStranica+1, pozicija-rijec.length()));
            brojacStranica++;
        }
    }
    return indeks;
}

//PretraziIndeksPojmova
std::set<std::tuple<std::string, int, int>>
 PretraziIndeksPojmova(const std::string &rijec, const std::map<std::string, std::
set<std::tuple<std::string,int,int>>> &indeks){
    std::string lowercaseRijec = rijec;
```

```
    std::transform(lowercaseRijec.begin(), lowercaseRijec.end(),
 lowercaseRijec.begin(), [](unsigned char c){return std::tolower(c);});
//pretvaranje u lowercase
    auto it = indeks.find(lowercaseRijec);
    if (it!=indeks.end()) return it->second;
    else throw std::logic_error("Pojam nije nadjen");
}

//IspisiIndeksPojmova
void IspisiIndeksPojmova(const std::map<std::string, std::set<std::tuple<std::
string,int,int>>> &indeks){
    for (const auto &unos : indeks)
    {
        std::cout<<unos.first<<": ";

//for (const auto &pozicija : unos.second) std::cout<<std::get<0>(pozicija)<<"/"<<std:
:get<1>(pozicija)<<"/"<<std::get<2>(pozicija)<<", ";
        //std::cout<<std::endl;
        auto it = unos.second.begin();
        if (it!=unos.second.end())
        {
            std::cout<<std::get<0>(*it)<<"/"<<std::get<1>(*it)<<"/"<<std::get<2>(*it);
            it++;
        }
        while (it != unos.second.end())
        {
            std::cout<<", "<<std::get<0>(*it)<<"/"<<std::get<1>(*it)<<"/"<<std::get<2
>(*it);
            it++;
        }
        std::cout<<std::endl;
    }
}


int main ()
{
    Knjiga knjiga;
    std::string oznakaPoglavlja;
    std::string sadrzajStranice;
    int brojStranice=1;

    for(;;)
    {
        std::cout<<"Unesite naziv poglavlja: \n";
        std::getline(std::cin, oznakaPoglavlja);
        if (oznakaPoglavlja==".") break;
        brojStranice=1; //resetuje ga
        for (;;)
        {
            std::cout<<"Unesite sadrzaj stranice "<< brojStranice<<": \n";
            std::getline(std::cin, sadrzajStranice);
            if (sadrzajStranice == ".") break;
            knjiga[oznakaPoglavlja].push_back(sadrzajStranice);
            brojStranice++;
        }
```

```
    }

    std::map<std::string,std::set<std::tuple<std::string, int, int>>> indeks=
KreirajIndeksPojmova(knjiga);
    std::cout<<"Kreirani indeks pojmova: \n";
    IspisiIndeksPojmova(indeks);
    std::string rijec;
    for (;;)
    {
        std::cout<<"Unesite rijec: \n";
        std::cin>>rijec;
        if (rijec==".") break;
        try
        {
            std::set<std::tuple<std::string, int, int>> rezultati =
 PretraziIndeksPojmova(rijec, indeks);
            for (const auto &pozicija : rezultati) std::cout<<std::get<0>(
pozicija)<<"/"<<std::get<1>(pozicija)<<"/"<<std::get<2>(pozicija)<<" ";

        }
        catch (const std::logic_error e)
        {
            std::cout <<"Unesena rijec nije nadjena!"<<std::endl;
        }
    }
        return 0;
}
```

Zadatak .

3.cpp

```
//TP 2022/2023: Zada?a 3, Zadatak 3
#include <iostream>
#include <iomanip>
#include <stdexcept>
#include <new>
template <typename TipElemenata>
 struct Matrica {
    char ime_matrice; // Koristi se samo u funkciji "UnesiMatricu"
    int br_redova, br_kolona;
    TipElemenata **elementi = nullptr; // VEOMA BITNA INICIJALIZACIJA!!!
 };

template <typename TipElemenata>
 Matrica<TipElemenata> StvoriMatricu(int br_redova, int br_kolona, char ime = 0) {
    Matrica<TipElemenata> mat;
    mat.br_redova = br_redova; mat.br_kolona = br_kolona; mat.ime_matrice = ime;
    mat.elementi = new TipElemenata*[br_redova]{};
    try
    {
        for(int i = 0; i < br_redova; i++)
        mat.elementi[i] = new TipElemenata[br_kolona]{};
    }
    catch(...)
    {
        UnistiMatricu(mat);
        throw;
    }
    return mat;
 }

template<typename TipElemenata>
Matrica<TipElemenata> ProduktMatrica(const Matrica<TipElemenata>&mat1, const
 Matrica<TipElemenata>&mat2){
    if (mat1.br_kolona!=mat2.br_redova) throw std::domain_error(
"Matrice nisu saglasne za mnozenje");
    Matrica<TipElemenata> pomnozenamatrica=StvoriMatricu<TipElemenata>(mat1.
br_redova, mat2.br_kolona);
    pomnozenamatrica.br_redova=mat1.br_redova;
    pomnozenamatrica.br_kolona=mat2.br_kolona;
    for (int i=0;i<pomnozenamatrica.br_redova;i++)
    {
        for (int j=0;j<pomnozenamatrica.br_kolona;j++)
        {
            for (int k=0;k<mat2.br_redova;k++)
            {
                pomnozenamatrica.elementi[i][j]+=mat1.elementi[i][k]*mat2.
elementi[k][j];
            }
        }
    }
    return pomnozenamatrica;
}

template<typename TipElemenata>
```

```
Matrica<TipElemenata> MatricniHarmonijskiPolinom (Matrica<TipElemenata>mat, int n){
    if (mat.br_redova!=mat.br_kolona) throw std::domain_error(
"Matrica nije kvadratna");
    if (n<0) throw std::domain_error("Pogresan parametar n");
    Matrica<TipElemenata> rezultat;
    try
    {
        rezultat=StvoriMatricu<TipElemenata>(mat.br_redova, mat.br_kolona);
    }
    catch(...)
    {
        UnistiMatricu(rezultat);
        throw std::bad_alloc();
    }
    for (int i=0;i<mat.br_redova;i++)
    {
        for (int j=0;j<mat.br_kolona;j++)
        {
            rezultat.elementi[i][j]=mat.elementi[i][j];
        }
    }
    double clan;
    Matrica<TipElemenata> faktor; //000
    Matrica<TipElemenata> o; //000
    Matrica<TipElemenata> temp; //000
    bool pocetak=true;
    for (int i=2;i<=n;i++)
    {
        if (!pocetak) temp=o;
        else temp=mat; //898
        try
        {
            faktor=ProduktMatrica(mat, temp); //934 nova matrica
        }
        catch(...)
        {
            UnistiMatricu(faktor);
            UnistiMatricu(rezultat);
            throw std::bad_alloc();
        }
        clan=1./i;
        for (int j=0;j<mat.br_redova;j++)
        {
            for (int k=0;k<mat.br_redova;k++)
            {
                rezultat.elementi[j][k]+=clan*faktor.elementi[j][k];
            }
        }
        if (!pocetak) UnistiMatricu(o);
        o=faktor;
        if (i==n) UnistiMatricu(o);
        pocetak=false;
    }
    return rezultat;
}
```

```
template <typename TipElemenata>
 void UnistiMatricu(Matrica<TipElemenata> &mat) {
    if(!mat.elementi) return;
    for(int i = 0; i < mat.br_redova; i++) delete[] mat.elementi[i];
    delete[] mat.elementi;
    mat.elementi = nullptr;
 }

template <typename TipElemenata>
 void UnesiMatricu(Matrica<TipElemenata> &mat) {
    for(int i = 0; i < mat.br_redova; i++)
    for(int j = 0; j < mat.br_kolona; j++)
    {
        std::cout << mat.ime_matrice << "(" << i + 1 << "," << j + 1 << ") = ";
        std::cin >> mat.elementi[i][j];
    }
 }

template <typename TipElemenata>
 void IspisiMatricu(const Matrica<TipElemenata> &mat, int sirina_ispisa, int
 preciznost=6, bool treba_brisati=false) {

    for(int i = 0; i < mat.br_redova; i++)
    {
        for(int j = 0; j < mat.br_kolona; j++)
        std::cout << std::setw(sirina_ispisa) << std::setprecision(
preciznost) << mat.elementi[i][j];
        std::cout << std::endl;
    }
    if (treba_brisati)
    {
        Matrica<TipElemenata> temp=mat;
        UnistiMatricu(temp);
    }
 }
template <typename TipElemenata>
 Matrica<TipElemenata> ZbirMatrica(const Matrica<TipElemenata> &m1,
 const Matrica<TipElemenata> &m2) {
    if(m1.br_redova != m2.br_redova || m1.br_kolona != m2.br_kolona)
    throw std::domain_error("Matrice nemaju jednake dimenzije!");
    auto m3 = StvoriMatricu<TipElemenata>(m1.br_redova, m1.br_kolona);
    for(int i = 0; i < m1.br_redova; i++)
    for(int j = 0; j < m1.br_kolona; j++)
    m3.elementi[i][j] = m1.elementi[i][j] + m2.elementi[i][j];
    return m3;
 }

 int main() {
    Matrica<double> A; // AUTOMATSKA INICIJALIZACIJA!!!
    Matrica<double> B;
    int dimenzija;
    std::cout << "Unesite dimenziju kvadratne matrice: ";
    std::cin >> dimenzija;
    std::cout<<"Unesite elemente matrice A: \n";
    try
    {
```

```
        A = StvoriMatricu<double>(dimenzija, dimenzija, 'A');
        UnesiMatricu(A);
        std::cout<<"Unesite red polinoma: ";
        int n;
        std::cin>>n;
        B=MatricniHarmonijskiPolinom(A, n);
    }
    catch(std::bad_alloc)
    {
        std::cout << "Nema dovoljno memorije!\n";
    }
    std::cout<<"Matricni harmonijski polinom:\n";
    IspisiMatricu(B, 10);
    UnistiMatricu(A);
    UnistiMatricu(B);
    return 0;
}
```

Zadatak . 4.cpp

```
//TP 2022/2023: Zada?a 3, Zadatak 4
#include <iostream>
#include <cmath>
#include <list>

int Poglavica(int N, int M, bool ispis=false){
    if (N<=0 || M<=0) return 0;
    std::list<int> clanovi;
    for (int i=1;i<=N;i++) clanovi.push_back(i);

    auto it=clanovi.begin();
    while (clanovi.size()>1)
    {
        for (int i=0;i<M-1;i++)
        {
            it++;
            if (it==clanovi.end()) it=clanovi.begin();
        }

        if (ispis) std::cout<<*it<<", ";
        it=clanovi.erase(it);
        if (it==clanovi.end()) it=clanovi.begin();
    }
    return *clanovi.begin();
}

int SigurnoMjesto(int M, int N1, int N2){
    if (M<=0 || N1<=0 || N2<=0) return 0;
    for (int mjesto=N1; mjesto<=N2; mjesto++)
    {
        bool jeSigurnoMjesto = true;
        for (int N=N1; N<=N2; N++)
        {
            if (Poglavica(N,M) == mjesto)
            {
                jeSigurnoMjesto = false;
                break;
            }
        }
        if (jeSigurnoMjesto) return mjesto;
    }
    return 0;
}


int main ()
{
    int N, M, N1, N2;
    std::cout<<
"Unesite broj punoljetnih clanova za odabir poglavice plemena Wabambe: ";
    std::cin>>N;
    std::cout<<"Unesite korak razbrajanja: ";
    std::cin>>M;
    std::cout<<"Unesite raspon za odabir (N1 i N2): ";
```

```
    std::cin>>N1>>N2;

    int poglavica = Poglavica(N,M);
    std::cout<<"\nRedni broj osobe koja postaje poglavica: "<<poglavica<<std::endl;

    int sigurnoMjesto = SigurnoMjesto(M,N1,N2);
    if (sigurnoMjesto == 0) std::cout<<
"Zao mi je Mbebe Mgogo, ali nema sigurnog mjesta."<<std::endl;
    else std::cout<<
"Mbebe Mgogo, stani na sigurno mjesto "<<sigurnoMjesto<<
" da ne bi bio poglavica!"<<std::endl;

        return 0;
}
```

Zadatak .

5.cpp

```
//TP 2022/2023: Zada?a 3, Zadatak 5
#include <iostream>
#include <cmath>

/*struct Clan{
    int redni_broj;
    Clan *sljedeci;
};*/

struct Cvor{
    int redni_broj;
    Cvor *sljedeci;
};

int Poglavica(int N, int M, bool ispis=false){
    if (N<=0 || M<=0) return 0;

    Cvor *prvi = new Cvor;
    prvi->redni_broj = 1;
    Cvor *trenutni = prvi;

    for (int i=2;i<=N;i++)
    {
        trenutni->sljedeci = new Cvor;
        trenutni = trenutni->sljedeci;
        trenutni->redni_broj=i;
    }
    trenutni->sljedeci=prvi;

    while (N>1)
    {
        for (int i=1;i<M;i++) trenutni=trenutni->sljedeci;

        if (ispis) std::cout<<trenutni->sljedeci->redni_broj<<", ";
        Cvor *izbaceni = trenutni->sljedeci;
        trenutni->sljedeci=izbaceni->sljedeci;
        delete izbaceni;
        N--;
    }
    int poglavica=trenutni->redni_broj;
    delete trenutni;
    return poglavica;
}

int SigurnoMjesto(int M, int N1, int N2){
    if (M<=0 || N1<=0 || N2<=0) return 0;

    for (int mjesto=N1; mjesto<=N2; mjesto++)
    {
        bool jeSigurnoMjesto=true;
        for (int N=N1; N<=N2; N++)
        {
            if (Poglavica(N,M)==mjesto)
            {
```

```
                jeSigurnoMjesto=false;
                break;
            }
        }
        if (jeSigurnoMjesto) return mjesto;
    }
    return 0;
}

int main ()
{
    int N,M,N1,N2;
    std::cout<<
"Unesite broj punoljetnih clanova za odabir poglavice plemena Wabambe: ";
    std::cin>>N;
    std::cout<<"Unesite korak razbrajanja: ";
    std::cin>>M;
    std::cout<<"Unesite raspon za odabir (N1 i N2): ";
    std::cin>>N1>>N2;

    int poglavica=Poglavica(N,M);
    std::cout<<"\nRedni broj osobe koja postaje poglavica: "<<poglavica<<std::endl;
    int sigurnoMjesto=SigurnoMjesto(M,N1,N2);
    if (sigurnoMjesto==0) std::cout<<
"Zao mi je Mbebe Mgogo, ali nema sigurnog mjesta."<<std::endl;
    else std::cout <<
"Mbebe Mgogo, stani na sigurno mjesto "<<sigurnoMjesto<<
" da ne bi bio poglavica!"<<std::endl;

        return 0;
}
```

Zadatak .

6.cpp

```
//TP 2022/2023: Zada?a 3, Zadatak 6
#include <iostream>
#include <cmath>
#include <memory>

/*struct Clan{
    int redni_broj;
    std::shared_ptr<Clan> sljedeci;
};*/

struct Cvor{
    int redni_broj;
    std::shared_ptr<Cvor> sljedeci;
};

int Poglavica(int N, int M, bool ispis=false){
    if (N<=0 || M<=0) return 0;

    std::shared_ptr<Cvor> prvi =std::make_shared<Cvor>();
    prvi->redni_broj=1;
    std::shared_ptr<Cvor> trenutni=prvi;

    for (int i=2;i<=N;i++)
    {
        trenutni->sljedeci = std::make_shared<Cvor>();
        trenutni=trenutni->sljedeci;
        trenutni->redni_broj=i;
    }
    trenutni->sljedeci = prvi;

    while (N>1)
    {
        for (int i=1; i<M; i++) trenutni=trenutni->sljedeci;

        if (ispis) std::cout<<trenutni->sljedeci->redni_broj<<", ";
        std::shared_ptr<Cvor> izbaceni=trenutni->sljedeci;
        trenutni->sljedeci=izbaceni->sljedeci;
        N--;
    }
    int poglavica=trenutni->redni_broj;
    trenutni->sljedeci=nullptr;
//prekid veze na posljednjem cvoru, oslobadjanje pametnog
    return poglavica;
}

int SigurnoMjesto(int M, int N1, int N2){
    if(M<=0 || N1<=0 || N2<=0) return 0;

    for (int mjesto=N1; mjesto<=N2; mjesto++)
    {
        bool jeSigurnoMjesto = true;
        for (int N=N1; N<=N2; N++)
        {
            if (Poglavica(N,M)==mjesto)
```

```
            {
                jeSigurnoMjesto=false;
                break;
            }
        }
        if (jeSigurnoMjesto) return mjesto;
    }
    return 0;
}

int main ()
{
    int N,M,N1,N2;
    std::cout<<
"Unesite broj punoljetnih clanova za odabir poglavice plemena Wabambe: ";
    std::cin>>N;
    std::cout<<"Unesite korak razbrajanja: ";
    std::cin>>M;
    std::cout<<"Unesite raspon za odabir (N1 i N2): ";
    std::cin>>N1>>N2;

    int poglavica=Poglavica(N,M);
    std::cout<<"\nRedni broj osobe koja postaje poglavica: "<<poglavica<<std::endl;
    int sigurnoMjesto=SigurnoMjesto(M,N1,N2);
    if (sigurnoMjesto==0) std::cout<<
"Zao mi je Mbebe Mgogo, ali nema sigurnog mjesta."<<std::endl;
    else std::cout<<
"Mbebe Mgogo, stani na sigurno mjesto "<<sigurnoMjesto<<
" da ne bi bio poglavica!"<<std::endl;

        return 0;
}
```