Федеральное агентство по образованию

Арзамасский политехнический институт (филиал)

Государственное образовательное учреждение высшего профессионального образования

"Нижегородский государственный технический университет" им. Р.Е. Алексеева

Кафедра "Прикладная математика"

Курсовая работа

по дисциплине

"Алгоритмические языки и программирование"

Выполнил студент

группы АСПМ 09-1

Балахонова О.В.

Проверил

Преподаватель

Лазарева А.Б

Арзамас 2010

1. Введение

2. Задача № 1

2.1 Блок – схема для ввода данных

2.2 Блок – схема для удаления элемента

2.3 Блок – схема для добавления элемента

* 1. 2.4 Блок – схема главной функции
  2. 2.5 Листинг программы
  3. 2.6 Результаты работы программы

3. Задача № 2

3.1 Постановка задачи

3.2 Блок – схема для конструктора

3.3 Блок – схема для конструктора копий

3.4 Блок – схема для деструктора

3.5 Блок – схемы для доступа к закрытым полям

3.6 Блок – схема для перестановки

3.7 Блок – схема для приведения к треугольному виду

3.8 Блок – схема для аппроксимации

3.9 Блок-схема для обратной подстановки

3.10 Блок-схема для вычисления интеграла методом левых прямоугольников

3.11 Блок-схема для вычисления интеграла методом правых прямоугольников

3.12 Блок-схема для вычисления интеграла методом центральных прямоугольников

3.13 Блок-схема для вычисления интеграла методом Симпсона с автоматическим выбором шага

3.14 Листинг программы

3.15 Результаты работы программы

3.16 Теоретические сведения

Список использованных источников

1. Введение

В начале 80–х годов в фирме Bell Telephone Laboratories Бьерном Страуструпом (США) в результате дополнения и расширения языка С был создан язык, получивший название "С с классами". В 1983г. Это название было изменено на С++. В 1998г. Был утвержден окончательный стандарт языка С++.

В настоящее время С++ является одним из наиболее распространенных и востребованных языков программирования. Этот язык представляет собой фундамент, на котором строится современное программирование, поэтому он является обязательным элементом подготовки современных программистов, именно поэтому для студентов специальностей "Прикладная математика" и "Информационные системы и технологии" выбран этот язык.

2. Задача № 1

Постановка задачи

Создать файл и записать в него объекты структуры. Структура должна быть со следующими полями:

* Название команды
* Город
* Количество игроков
* Количество набранных очков

Из объектов структуры создается массив, заносится в файл. Вывести созданный файл на монитор, удалить из файла все элементы с количеством очков меньше заданного. Измененную базу данных записать в другой файл. Добавить К элементов в начало файла.

2.1 Блок – схема для ввода данных



2.2 Блок – схема для удаления элемента



2.3 Блок – схема для добавления элемента



2.4 Блок – схема главной функции



2.5 Листинг программы

#include <iostream>

#include <string>

#include <cstdio>

#include <windows.h>

#include <fstream>

using namespace std;

struct set

{char name[50];

char city[20];

int pl;

int point;

};

void data(set a1[50],int& kol );

void udal(set a[50],ofstream& f,char str[15],int& kol);

void vvodf(char str[15],ofstream& f,set a[50],int& kol);

void dobav(set a[50],ofstream& f,char str[15],int& kol);

void vivod(set a[50],int& kol);

//главная функция

void main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

set a[50];

int d;

ofstream f1;

char str1[15];

cout<<"Введите полное имя файла \n"; cin>>str1;

data(a,d);

vvodf(str1,f1,a,d);

udal(a,f1,str1,d);

dobav(a,f1,str1,d);

}

//ввод данных

void data(set a1[50],int& kol)

{ cout<<"\n Сколько записей вы хотите ввести? \n";

cin>>kol;

for(int i=0;i<kol;i++)

{

cout<<"\n Введите название команды "<<i+1<<"\n";

gets(a1[i].name);

cin.getline(a1[i].name,sizeof(a1[i].name));

cout<<"\n Введите название города \n";

cin>>a1[i].city;

cout<<"\n Введите количество игроков \n";

cin>>a1[i].pl;

cout<<"\n Введите набранных очков \n";

cin>>a1[i].point;

}

vivod(a1,kol);

}

//ввод в файл

void vvodf(char str[15],ofstream& f,set a[50],int& kol)

{ f.open(str);

if(f.fail()) {cout<<"Ошибка при вводе файла"; exit(1);}

for(int i=0;i<kol;i++)

{ f<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

f<<"Название команды "<<i+1<<":"<<a[i].name<<endl;

f<<"Город: "<<a[i].city<<endl;

f<<"Количество игроков: "<<a[i].pl<<endl;

f<<"Количество очков: "<<a[i].point<<endl;

f<<endl<<endl;

}

f.close();

}

//удаление элемента

void udal(set a[50],ofstream& f,char str[15],int& kol)

{ cout<<"Введите количество очков и все команды с меньшим значением будут удалены ";

int ef; cin>>ef; int z=0;int q=0;int m=0;

for(int i=0;i<kol;i++)

{if(a[i].point<ef) {z=i;

for(int i=z;i<kol;i++)

a[i]=a[i+1];

kol--;

m=1;

}

if (m==1) {i--;m=0;}

}

vvodf(str,f,a,kol);

vivod(a,kol);

}

//добавление элемента

void dobav(set a[50],ofstream& f,char str[15],int& kol)

{

int koldob,nom=0;

cout<<"\n Сколько записей вы хотите добавить? \n"; cin>>koldob;

for(int i=kol-1;i>=nom;i--) a[i+koldob]=a[i];

f.open(str);

if(f.fail()) {cout<<"Ошибка открытии файла"; exit(1);}

for(int i=nom;i<nom+koldob;i++)

{

cout<<"\n Введите название команды "<<i+1<<" которую вы хотите добавить\n";

gets(a[i].name);

cin.getline(a[i].name,sizeof(a[i].name));

cout<<"\n Введите название города \n";

cin>>a[i].city;

cout<<"\n Введите количество игроков\n";

cin>>a[i].pl;

cout<<"\n Количество очков \n";

cin>>a[i].point;

}

kol=kol+koldob;

f.close();

vvodf(str,f,a,kol);

vivod(a,kol);

}

void vivod(set a[50],int& kol)

{cout<<endl;

for(int i=0;i<kol;i++)

{cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<"Название команды"<<i+1<<":"<<a[i].name<<endl;

cout<<"Город:"<<a[i].city<<endl;

cout<<"Количество игроков:"<<a[i].pl<<endl;

cout<<"Количество очков:"<<a[i].point<<endl;

cout<<endl<<endl;

}

}

2.6 Результаты работы программы

Введите полное имя файла

Команда

Сколько записей вы хотите ввести?

4

Введите название команды 1

Спартак

Введите название города

Москва

Введите количество игроков

21

Введите набранных очков

42

Введите название команды 2

ЦСКА

Введите название города

Москва

Введите количество игроков

21

Введите набранных очков

38

Введите название команды 3

Локомотив

Введите название города

Москва

Введите количество игроков

21

Введите набранных очков

30

Введите название команды 4

Томь

Введите название города

Томь

Введите количество игроков

21

Введите набранных очков

26

Название команды1:Спартак

Город:Москва

Количество игроков:21

Количество очков:42

Название команды2:ЦСКА

Город:Москва

Количество игроков:21

Количество очков:38

Название команды3:Локомотив

Город:Москва

Количество игроков:21

Количество очков:30

Название команды4:Томь

Город:Томь

Количество игроков:21

Количество очков:26

Введите количество очков и все команды с меньшим значением будут удалены 38

Название команды1:Спартак

Город:Москва

Количество игроков:21

Название команды2:ЦСКА

Город:Москва

Количество игроков:21

Количество очков:38

Сколько записей вы хотите добавить?

1

Введите название команды 1 которую вы хотите добавить

Спартак

Введите название города

Нальчик

Введите количество игроков

21

Количество очков

28

Название команды1:Спартак

Город:Нальчик

Количество игроков:21

Количество очков:28

Название команды2:Спартак

Город:Москва

Количество игроков:21

Количество очков:42

Название команды3:ЦСКА

Город:Москва

Количество игроков:21

Количество очков:38

Для продолжения нажмите любую клавишу . . .

3. Задача № 2

3.1 Постановка задачи

Дана таблица значений функции. Требуется аппроксимировать ее полиномом 6й степени.

Для решения задачи необходимо решать систему линейных уравнений методом Гаусса или итераций.

Для решения системы линейных уравнений методом Гаусса необходимо создать классы с одномерным динамическим массивом и двумерным динамическим массивом, в которых должны быть:

* Конструктор;
* Конструктор по умолчанию;
* Конструктор копий;
* Деструктор;
* Методы доступа к закрытым полям;
* Метод для инициализации закрытого поля.

Вне классов должны перегружаться операции вставки и извлечения.

Должны быть добавлены функции или дружественные функции для:

1. Левых прямоугольников;
2. Правых прямоугольников;
3. Центральных прямоугольников;
4. Методом Симпсона с автоматическим выбором шага.

3.2 Блок – схема для конструктора



3.3 Блок – схема для конструктора копий



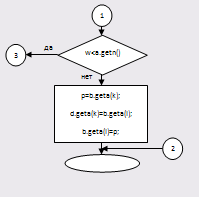
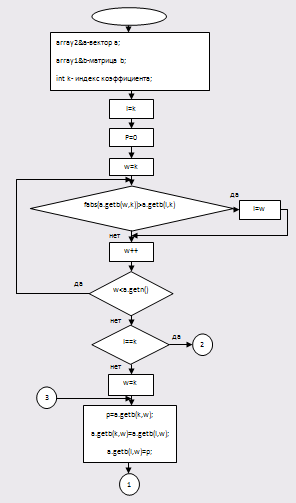
3.4 Блок – схема для деструктора



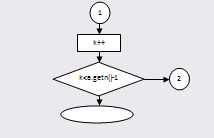
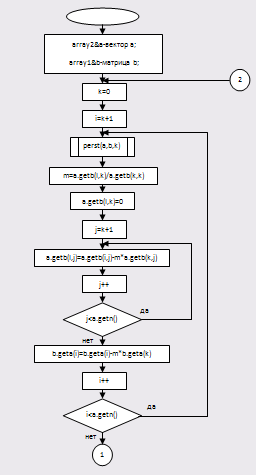
3.5 Блок – схемы для доступа к закрытым полям



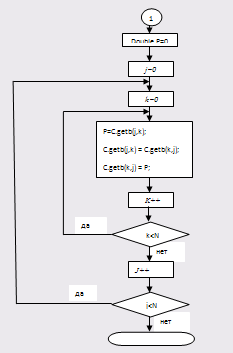
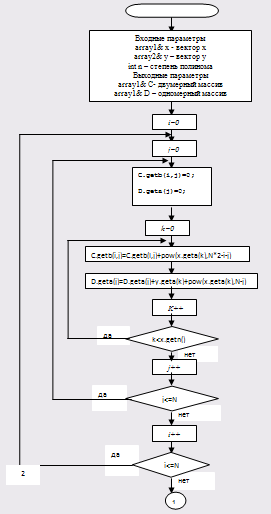
3.6 Блок – схема для перестановки



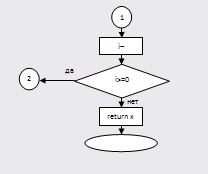
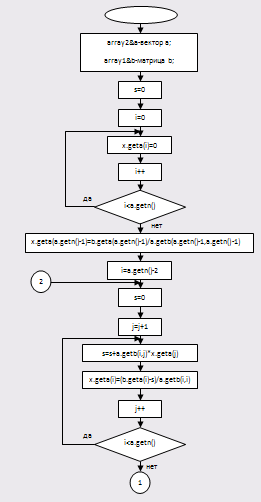
3.7 Блок – схема для приведения к треугольному виду



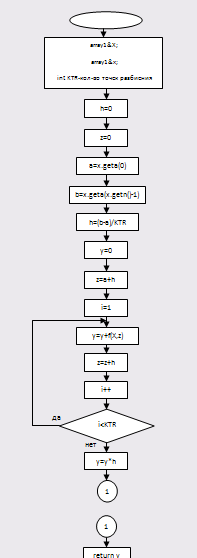
3.8 Блок – схема для аппроксимации



3.9 Блок-схема для обратной подстановки



3.10 Блок-схема для вычисления интеграла методом левых прямоугольников



1

y=y\*h

нет

да

i<KTR

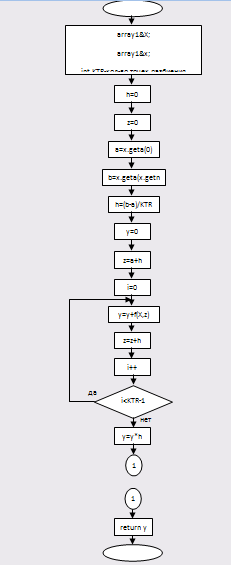
i++

z=z+h

y=y+f(X,z)

i=1

3.11 Блок-схема для вычисления интеграла методом правых прямоугольников



1

y=y\*h

i<KTR-1

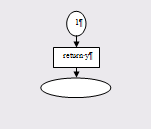
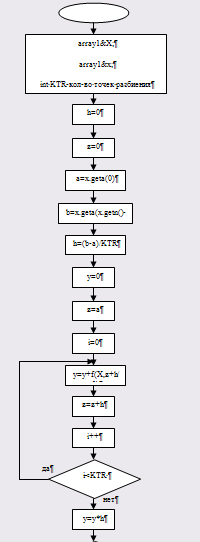
i++

z=z+h

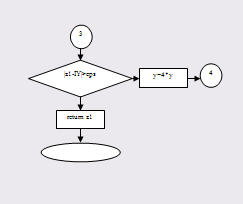
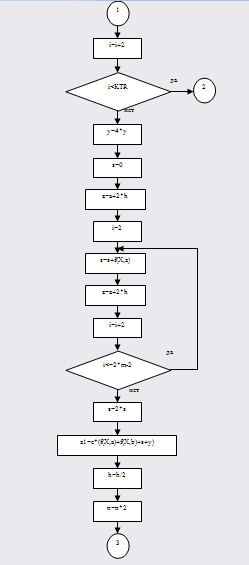
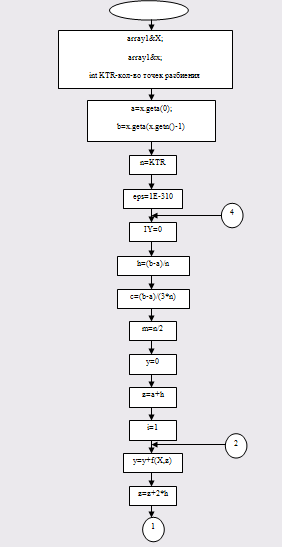
y=y+f(X,z)

i=0

3.12 Блок-схема для вычисления интеграла методом центральных прямоугольников



3.13 Блок-схема для вычисления интеграла методом Симпсона с автоматическим выбором шага



3.14 Листинг программы

#include <iostream>

#include <cstdlib>

#include <cstddef>

#include <iomanip>

#include <cmath>

#include <fstream>

using namespace std;

class array1

{double\* a;

int n;

public:

array1(int nn);

array1();

array1(const array1& ob);

~array1();

int getn();

double& geta(int n);

void set1(int i,double v);

};

class array2

{double\*\* b;

int n,m;

public:

array2(int nn,int nm);

array2();

array2(const array2& ob);

~array2();

int getn();

int getm();

double& getb(int k,int l);

void set(int i,int j,double v);

};

//конструктор array1

array1::array1(int nn)

{n=nn;

a=new double [n];

if (a==NULL){cout<<"\n нет оп ";

exit(1);

}

}

//конструктор по умолчанию array1

array1::array1()

{

}

//деструктор array1

array1::~array1()

{

}

//конструктор копий array1

array1::array1(const array1 &ob)

{n=ob.n;

a=new double[n];

if (a==NULL){cout<<"\n нет оп " ;

exit(1);

}

for (int i=0;i<n;i++)

a=ob.a;

}

//методы доступа array1

int array1::getn()

{return n;

}

double& array1::geta(int r)

{return a[r];

}

void array1::set1(int i,double v)

{a[i]=v;

}

//операция извлечения

istream& operator>>(istream& stream, array1& ob)

{int i; double temp;

for(i=0;i<ob.getn();i++)

{stream>>temp;

ob.set1(i,temp);

}

return stream;

}

//операция вставки

ostream& operator<<(ostream& stream, array1& ob)

{int i;

for(i=0;i<ob.getn();i++)

stream<<setw(10)<<ob.geta(i);

return stream;

}

//конструктор array2

array2::array2(int nn,int nm)

{ n=nn;

m=nm;

b=new double\* [n];

if(b==NULL) {cout<<"\n НеТ ОП";exit(1); }

for(int i=0;i<n;i++)

{b[i]=new double [m];

if(b[i]==NULL) {cout<<"\n НеТ ОП";exit(1);}

}

}

//конструктор по умолчанию array2

array2::array2()

{

}

//конструктор копий array2

array2::array2(const array2 &ob)

{n=ob.n;

m=ob.m;

b=new double\* [n];

if(b==NULL) {cout<<"\n НеТ ОП";

exit(1);

}

for(int i=0;i<n;i++)

for(int j=0;j<m;j++)

b[i][j]=ob.b[i][j];

}

//деструктор array2

array2::~array2()

{//delete[] a;

}

//методы доступа array2

int array2::getn()

{return n;

}

int array2::getm()

{return m;

}

double& array2::getb(int k,int l)

{return b[k][l];

}

void array2::set(int i,int j,double v)

{b[i][j]=v;

}

//операция извлечения

istream& operator>>(istream& stream, array2& ob)

{int i,j;

double temp;

for(i=0;i<ob.getn();i++)

for(j=0;j<ob.getm();j++)

{stream>>temp;

ob.set(i,j,temp);

}

return stream;

}

//операция вставки

ostream& operator<<(ostream& stream, array2& ob)

{int i,j;

for(i=0;i<ob.getn();i++)

{stream<<endl;

for(j=0;j<ob.getm();j++)

stream<<setw(10)<<ob.getb(i,j);

}

return stream;

}

//перестановка уравнений

void Perestanovka(array2& a,array1& b,int k)

{int l=k;

double p=0;

for(int w=k;w<a.getn();w++)

if(fabs(a.getb(w,k))>a.getb(l,k)) l=w;

if(l!=k) {for(int w=k;w<a.getn();w++)

{p=a.getb(k,w);

a.getb(k,w)=a.getb(l,w);

a.getb(l,w)=p;

}

p=b.geta(k);

b.geta(k)=b.geta(l);

b.geta(l)=p;

}

}

//привидение системы к треугольному виду

void Treugol(array2& a,array1& b)

{double m;

for(int k=0;k<a.getn()-1;k++)

{for(int i=k+1;i<a.getn();i++)

{Perest(a,b,k);

m=a.getb(i,k)/a.getb(k,k);

a.getb(i,k)=0;

for(int j=k+1;j<a.getn();j++)

a.getb(i,j)=a.getb(i,j)-m\*a.getb(k,j);

b.geta(i)=b.geta(i)-m\*b.geta(k);

}

}

}

//получение неизвестных

array1 Pods(array2& a,array1& b)

{array1 x(a.getn());

double s=0;

for(int i=0;i<a.getn();i++)

x.geta(i)=0;

x.geta(a.getn()-1)=b.geta(a.getn()-1)/a.getb(a.getn()-1,a.getn()-1);

for(int i=a.getn()-2;i>=0;i--)

{s=0;

for(int j=i+1;j<a.getn();j++)

s=s+a.getb(i,j)\*x.geta(j);

x.geta(i)=(b.geta(i)-s)/a.getb(i,i);

}

return x;

}

void vivod(array1& X)

{cout<<endl;

for(int i=0;i<X.getn();i++)

{cout<<"="<<"("<<X.geta(i)<<"\*x^"<<X.getn()-i-1<<")"; }

}

//аппроксимация

void Appr(array1 &x,array1& y,int N,array2& C,array1& D)

{array1 X(N+1);

for(int i=0;i<=N;i++)

for(int j=0;j<=N;j++)

{C.getb(i,j)=0;

D.geta(j)=0;

for(int k=0;k<x.getn();k++)

{C.getb(i,j)=C.getb(i,j)+pow(x.geta(k),N\*2-i-j);

D.geta(j)=D.geta(j)+y.geta(k)\*pow(x.geta(k),N-j);

}

}

double p=0;

for(int j=0;j<N;j++)

for(int k=0;k<N;k++)

{p=C.getb(j,k);

C.getb(j,k)=C.getb(k,j);

C.getb(k,j)=p;

}

}

//полученная фун-я

double f(array1& X,double z)

{double y=0;

for(int i=0;i<X.getn();i++)

y=y+X.geta(i)\*pow(z,X.getn()-i-1);

return y;

}

//метод левых прямоугольников

double LevPR(array1& X,int KTR,array1& x)

{ double h=0;

double z=0;

double a=x.geta(0);

double b=x.geta(x.getn()-1);

h=(b-a)/KTR;

double y=0;

z=a+h;

for(int i=1;i<KTR;i++)

{y=y+f(X,z);

z=z+h;

}

y=y\*h;

return y;

}

//метод правых прямоугольников

double PravPr(array1& X,int KTR,array1& x)

{double h=0;

double z=0;

double a=x.geta(0);

double b=x.geta(x.getn()-1);

h=(b-a)/KTR;

double y=0;

z=a+h;

for(int i=0;i<KTR-1;i++)

{y=y+f(X,z);

z=z+h;

}

y=y\*h;

return y;

}

//метод центральных прямоугольников

double CentrPr(array1& X,int KTR,array1& x)

{double h=0;

double z=0;

double a=x.geta(0);

double b=x.geta(x.getn()-1);

h=(b-a)/KTR;

double y=0;

z=a;

for(int i=0;i<KTR;i++)

{y=y+f(X,z+h/2);

z=z+h;

}

y=y\*h;

return y;

}

//Метод Симпсона с автомат. выбором шага

double Simpson(array1& X,int KTR,array1& x)

{double z1, h, c, s, y, IY, eps,z,n,m;

int i;

double a=x.geta(0);

double b=x.geta(x.getn()-1);

n=KTR;

eps=1E-310;

IY=0;

h=(b-a)/n;

do

{c=(b-a)/(3\*n);

m=n/2;

y=0;

z=a+h;

for(i=1;i<=(2\*m-1);i=i+2)

{y=y+f(X,z);

z=z+2\*h;

}

y=4\*y;

s=0;

z=a+2\*h;

for(i=2;i<=(2\*m-2);i=i+2)

{s=s+f(X,z);

z=z+2\*h;

}

s=2\*s;

z1=c\*(f(X,a)+f(X,b)+s+y);

h=h/2;

n=n\*2;

if(fabs(z1-IY)>eps) IY=z1;

}

while(fabs(z1-IY)>eps);

return z1;

}

//вывод из файла

void outputf(ifstream& f,char S[40],array1& v)

{double next;int i=0;

f.open(S);

if(f.fail()) {cout<<" Error";

exit(1);

}

while(f>>next)

{v.geta(i)=next;

i++;

}

f.close();

}

//Главная функция

void main()

{cout.setf(ios::fixed);

cout.setf(ios::showpoint);

cout.precision(2);

setlocale(LC\_ALL,"Russian");

int n,N;

cout<<"\n Введите количество элементов векторов x и y \n";

cin>>n;

array1 x(n),y(n);

cout<<"\n \*\*Введите степень полинома\*\*\n";

cin>>N;

char S[40];

char S1[40];

cout<<"\n Введите путь к вектору x \n";

cin>>S;

cout<<"\n Введите путь к вектору y \n";

cin>>S1;

ifstream f,f1;

cout<<"\*\*\*\*\*Вектор x\*\*\*\*\*";

outputf(f,S,x);

cout<<x<<endl;

cout<<endl<<"\*\*\*\*\*Вектор y\*\*\*\*\*"<<endl;

outputf(f1,S1,y);

cout<<y<<endl;

array2 C(N+1,N+1);array1 D(N+1);

Appr(x,y,N,C,D);

Treugol(C,D);

cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Матрица C\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"<<C<<endl;

cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Вектор D\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"<<D<<endl;

array1 X(N+1);

X=Pods(C,D);cout<<"\n \*\*\*\*Коэффициенты\*\*\*\* \n"<<X<<endl;

vivod(X);

double I=0;

double I2=0;

double I3=0;

double I4=0;

cout<<"\n Введите количество точек разбиения \n";

int KTR=0;

cin>>KTR;

I=LevPr(X,KTR,x);

cout<<"\n Интеграл методом левых прямоугольников \n"<<I;

I2=PravPr(X,KTR,x);

cout<<"\n Интеграл методом правых прямоугольников\n"<<I2;

I3=CentrPr(X,KTR,x);

cout<<"\n Интеграл методом центральных прямоугольников\n"<<I3;

I4=Simpson(X,KTR,x);

cout<<"\n Интеграл методом Симпсона с автоматическим выбором шага\n"<<I4;

cout<<endl;

}

3.15 Результаты работы программы

Введите количество элементов векторов x и y

51

\*\*Введите степень полинома\*\*

6

Введите путь к вектору x

D:\c++\x.txt

Введите путь к вектору y

D:\c++\y.txt

\*\*\*\*\*Вектор x\*\*\*\*\* -5.00 -4.80 -4.60 -4.40 -4.20 -4.00

-3.80 -3.60 -3.40 -3.20 -3.00 -2.80 -2.60 -2.40

-2.20 -2.00 -1.80 -1.60 -1.40 -1.20 -1.00 -0.80

-0.60 -0.40 -0.20 0.00 0.20 0.40 0.60 0.80

1.00 1.20 1.40 1.60 1.80 2.00 2.20 2.40

2.60 2.80 3.00 3.20 3.40 3.60 3.80 4.00

4.20 4.40 4.60 4.80 5.00

\*\*\*\*\*Вектор y\*\*\*\*\*

123863.00 96837.80 74907.15 57273.05 43235.06 32182.00 23584.04 16985.15

11995.90 8286.62 5581.00 3649.95 2305.94 1397.61 804.81 434.00

214.01 92.15 30.74 3.94 -5.00 -6.11 -4.77 -3.34

-2.47 -2.00 -1.53 -0.59 1.52 6.31 17.00 39.84

85.73 172.28 326.19 586.00 1005.27 1656.04 2632.71 4056.29

6079.00 8889.25 12716.97 17839.36 24586.94 33350.00 44585.45 58823.97

76677.60 98847.65 126133.00

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Матрица C\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1202617144.06 -0.0054804623.21 0.002581573.91 0.00 127856.81

0.0054804623.21 0.002581573.91 0.00 127856.81 0.00

0.00 0.00 84065.25 -0.00 10211.57 -0.00 1065.08

0.00 0.00 0.00 6251.67 0.00 868.96 0.00

0.00 0.00 0.00 0.00 109.56 -0.00 38.16

0.00 0.00 0.00 0.00 0.00 22.93 0.00

0.00 0.00 0.00 0.00 0.00 0.00 10.62

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Вектор D\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

9620681443.2223489892.10 -2130.89 58003.22 -76.32 45.85 -21.24

\*\*\*\*Коэффициенты\*\*\*\*

8.00 -0.00 -0.00 9.00 -0.00 2.00 -2.00

+(8.00\*x^6)+(-0.00\*x^5)+(-0.00\*x^4)+(9.00\*x^3)+(-0.00\*x^2)+(2.00\*x^1)+(-2.00\*x^0)

Введите количество точек разбиения

10000000

Интеграл методом левых прямоугольников

178551.30

Интеграл методом правых прямоугольников

178551.30

Интеграл методом центральных прямоугольников

178551.43

Интеграл методом Симпсона с автоматическим выбором шага

178551.43

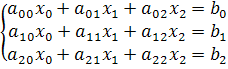
Для продолжения нажмите любую клавишу

3.16 Теоретические сведения

Решение системы линейных уравнений методом Гаусса

Проиллюстрируем метод на системе из трех линейных уравнений с тремя неизвестными

(1)



В такой системе, по крайней мере, один из коэффициентов отличен от нуля. Уравнения необходимо переставить таким образом, чтобы на месте первого уравнения было уравнение с максимальным отличным от нуля коэффициентом при Х=0. Далее вводится множитель , на него умножается первое уравнение системы и вычитается из второго уравнения. При этом мы получим

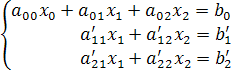


Третье уравнение системы (1) умножим на коэффициент и вычтем из второго уравнения первое уравнение, умноженное на этот коэффициент. В результате получим



Таким образом система уравнений (1) приводится к виду:

(2)

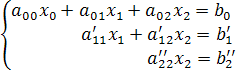


Для системы уравнений (2) введем множитель , умножим на него второе уравнение системы (2) и вычтем из третьего уравнения. В результате третье уравнение системы (2) примет вид:



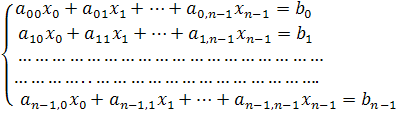
Тогда система уравнений (2) примет вид:

(3)



Мы получили треугольную систему уравнений и необходимо выполнить обратную подстановку для вычисления неизвестных.

Обобщим метод на случай системы из n уравнений с n неизвестными.



На k-ом этапе мы исключаем из системы уравнений с помощью множителей

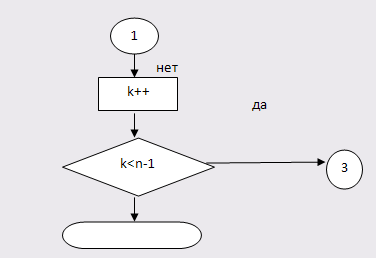


, где i=k+1, k+2,…,n-1



J=k,k+1,…,n-1

Индекс k принимает значения k=0,1,…,n-2 включительно. При k=n-2 происходит исключение из последнего уравнения и в результате получится треугольная система уравнений.



В представленной блок-схеме все множители, на которые нужно умножать уравнения, обозначаем буквой m, т. к. на каждом этапе требуется не более 1-го множителя.

k-номер уравнения, который вычитается из остальных, а также неизвестного, который исключается из остальных n-k уравнений

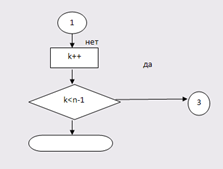
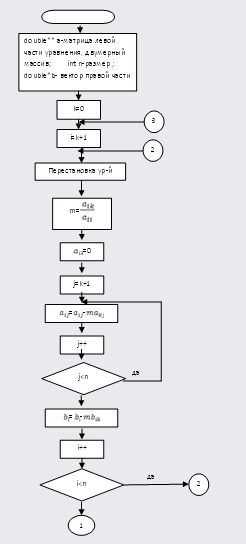
i-номер уравнения, из которого в данный момент исключается неизвестное, j-номер столбца

После приведения системы к ∆-му виду необходимо сделать обратный ход для нахождения неизвестных, значения которых будут вычисляться по формуле:

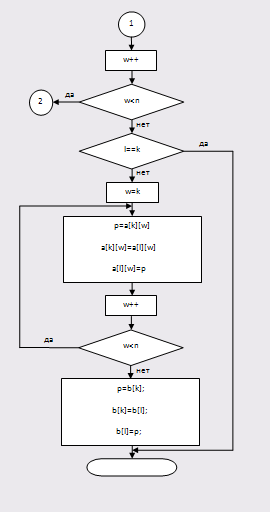


Где j-n-2, n-3,…,0

Начертим блок-схему для функции обратной подстановки



Начертим блок-схему для функции определения наибольшего коэффициента и перестановки уравнений при необходимости.



Аппроксимация функций методом наименьших квадратов

Пусть в результате эксперимента получается следующая таблица значений:

|  |  |  |  |
| --- | --- | --- | --- |
| x0 | x1 | … | xn-1 |
| y0 | y1 | … | yn-1 |

Требуется найти аналитический вид функции, которая в точках x0, x1,…,xn-1 будет иметь значения достаточно близкие к y0, y1,…,yn-1.

Рассмотрим наиболее простой вид аппроксимации, т.е. замены таблицы значений аналитическим видом функции – аппроксимацию линейной функцией, т.е. будем находить функцию y=a0\*x+a1, из которой в точках x0, x1,…,xn-1 значения будут близки к y0, y1,...,yn-1. В качестве критерия близости будем рассматривать квадратичный критерий качества

(1)



Необходимо найти коэффициенты a0 и a1, чтобы критерий качества стремился к минимуму. Как известно, необходимое условие минимума – равенство 0 первых производных функции по соответствующим переменным. Найдем производные критерия качества (1) по переменным a0 и a1

(2)



Разделим каждое из уравнений системы (2) на -2 и поставим неизвестные перед коэффициентами

(3)



Система (3) – система из двух линейных уравнений с двумя неизвестными. Будем решать его методом алгебраического сложения. Первое уравнение системы (3) умножим на n, а второе уравнение на и вычтем из первого уравнения второе, из второго уравнения системы (3) получим выражение для a1



Аппроксимация квадратичной функции:

Пусть дана таблица значений

|  |  |  |  |
| --- | --- | --- | --- |
| x0 | x1 | ……… | xn-1 |
| y0 | y1 | ……… | yn-1 |

Требуется аппроксимировать эту таблицу полиномом третьего порядка, т.е. требуется найти коэффициенты полинома y=a0x2+a1x+a2 , который в точках x0 , x1 , … , xn-1 имеет значения достаточно близкие к y0 , y1 , … , yn-1.

В качестве критерия близости выберем квадратичный критерий качества, который должен стремиться к минимуму.

= (1)



Найдем производные функции (1) по переменным а0 , а1 и а2.

2\*



(2) 2\*



2\*



Разделим каждое уравнение системы (2) на (-2) и поставим коэффициенты после неизвестных



(3)



Система уравнений (3) это система из трех линейных уравнений с тремя неизвестными, которую необходимо решать методом Гаусса или итерации.

Для того, чтобы решить систему методом Гаусса необходимо вычислить коэффициенты при неизвестных а0 , а1 и а2 и записать их в соответствующие переменные и, может быть, выполнить переобозначение , так как при решении системы уравнений методом Гаусса систему уравнений мы записывали в виде:

a00\*x0+a01\*x1+…+a0,n-1=b0

a10\*x0+a11\*x1+…+a1,n-1=b1

………………………………………...

an-1,0\*x0+an-1,1\*x1+…+an-1,n-1=bn-1

или систему уравнений рассмотрим в виде:

x00\*a0+x01\*a1+…+x0n-1=b0

x10\*a0+x11\*a1+…+x1n-1=b1

…………………………..

Список использованных источников

1. Программирование на С++ в среде Visual Studio C++. NET: учеб. пособие / А. Б. Лазарева, А. В. Троицкий, С. Н. Митяков – Н. Новгород, 2008.-334
2. Алгоритмические языки и программирование: методические указания к курсовой работе для студентов специальностей 230401.65 / А. Б. Лазарева, Т. Е. Жилина – АПИ НГТУ 2010.-43