**Лабораторная работа №4**

**Задача.**

Разработать программу численного решения СЛАУ методом

Якоби и методом релаксации (n – порядок матрицы системы Ax=f)

Матрица системы имеет диагональное преобладание, для первого уравнения преобладание строгое.

Недиагональные элементы ai,j, i≠j, выбираются из чисел 0, –1, –2, –3, –4 произвольным образом;



Задать правую часть f умножением матрицы A на вектор x=(m, m+1, ... , n+m–1): f=Ax.

Метод Якоби:



Метод релаксации:



Метод Гаусса-Зейделя – частный случай метода релаксации при w = 1.

**Входные данные.**

Size = 10, m = 14, w = 0.5, 1, 1.5

**Листинг программы.**

#include <iostream>  
#include <vector>  
#include <cmath>  
  
int size = 10**;**int m = 14**;**float eps = 0.0001**;**int kmax = 1000**;**std::vector<float> generateVectorX() {  
 std::vector<float> x(size**,** 0)**;** for (int i = 0**;** i < size**;** ++i) {  
 x[i] = (float)i + (float)m**;** }  
 return x**;**}  
  
std::vector<float> multiply(std::vector<std::vector<float>> A**,** std::vector<float> x) {  
 std::vector<float> ans(size**,** 0)**;** {  
 for (int i = 0**;** i < size**;** i++) {  
 float sum = 0**;** for (int j = 0**;** j < size**;** j++) {  
 sum += A[i][j] \* x[j]**;** }  
 ans[i] = sum**;** }  
 }  
 return ans**;**}  
  
std::vector<std::vector<float>> generateMatrix() {  
 std::vector<std::vector<float>> A(size**,** std::vector<float>(size**,** 0))**;** for (int i = 0**;** i < size**;** ++i) {  
 for (int j = 0**;** j < size**;** ++j) {  
 if (i != j) {  
 A[i][j] = rand() % 5 - 4**;** A[i][i] -= A[i][j]**;** }  
 }  
  
 }  
 A[0][0] +=1**;** return A**;**}  
  
void printResults(std::vector<float> vector**,** int kNum) {  
 if (kNum == kmax)  
 std::cout << "Number of iterations has been exceeded" << std::endl**;** else  
 std::cout << "Iteration " << kNum << std::endl**;** for (int i = 0**;** i < size**;** i++)  
 std::cout << vector[i] << " "**;** std::cout << std::endl**;**}  
  
void JakobiMethod(std::vector<std::vector<float>> &A**,** std::vector<float>& f)  
{  
 std::vector<float> ans(size**,** 0)**;** std::vector<float> copy(size**,** 0)**;** //первая итерация  
 for (int i = 0**;** i < size**;** ++i) {  
 ans[i] = f[i]/A[i][i]**;** copy[i] = ans[i]**;** }  
  
 int iterNumber = 0**;** for (int k = 1**;** k <= kmax**;** ++k) {  
 for (int i = 0**;** i < size**;** ++i) {  
 float sum = f[i]**;** for (int j = 0**;** j < i**;** ++j) {  
 sum-= A[i][j]\*copy[j]**;** }  
 for (int j = i+1**;** j < size**;** ++j) {  
 sum -= A[i][j]\*copy[j]**;** }  
 copy[i] = ans[i]**;** ans[i] = sum/A[i][i]**;** }  
  
 float maxConf = 0**;** for (int i = 0**;** i < size**;** ++i){  
 float temp = std::abs(ans[i] - copy[i])**;** if (temp > maxConf)  
 maxConf = temp**;** }  
 iterNumber = k**;** if(maxConf < eps)  
 break**;** }  
  
 printResults(ans**,** iterNumber)**;**}  
  
void RelaxationMethod(std::vector<std::vector<float>> &A**,** std::vector<float>& f**,** float w)  
{  
 std::vector<float> ans(size**,** 0)**;** std::vector<float> copy(size**,** 0)**;** for (int i = 0**;** i < size**;** ++i) {  
 ans[i] = f[i]/A[i][i]**;** copy[i] = ans[i]**;** }  
  
 int iterNumber = 0**;** for (int k = 1**;** k <= kmax**;** ++k) {  
 for (int i = 0**;** i < size**;** ++i) {  
 float sum = f[i]**;** for (int j = 0**;** j < i**;** ++j) {  
 sum-= A[i][j]\*ans[j]**;** }  
 for (int j = i+1**;** j < size**;** ++j) {  
 sum -= A[i][j]\*ans[j]**;** }  
 copy[i] = ans[i]**;** ans[i] = (1-w)\*ans[i] + w\*(sum/A[i][i])**;** }  
  
 float maxConf = 0**;** for (int i = 0**;** i < size**;** ++i){  
 float temp = std::abs(ans[i] - copy[i])**;** if (temp > maxConf)  
 maxConf = temp**;** }  
 iterNumber = k**;** if(maxConf < eps)  
 break**;** }  
  
 printResults(ans**,** iterNumber)**;**}  
  
  
int main()  
{  
 std::vector<std::vector<float>> matrix = generateMatrix()**;** std::vector<float> x = generateVectorX()**;** std::vector<float> f = multiply(matrix**,** x)**;** //метод якоби  
 JakobiMethod(matrix**,** f)**;** //метод Гаусса-Зейделя  
 RelaxationMethod(matrix**,** f**,** 1)**;** //метод релаксации  
 RelaxationMethod(matrix**,** f**,** 0.5)**;** RelaxationMethod(matrix**,** f**,** 1.5)**;** return 0**;**}

**Выходные данные.**

Метод Якоби:

Number of iterations has been exceeded

13.7923 14.7768 15.7781 16.7788 17.7789 18.7769 19.7788 20.7787 21.7801 22.7783

Метод Гаусса-Зейделя:

Number of iterations has been exceeded

13.7893 14.7735 15.7749 16.7756 17.7757 18.7737 19.7756 20.7755 21.7769 22.7751

Метод релаксации:

Iteration 603

13.9931 14.9926 15.9927 16.9927 17.9927 18.9926 19.9927 20.9927 21.9928 22.9927

Iteration 239

13.9977 14.9975 15.9975 16.9976 17.9976 18.9975 19.9976 20.9976 21.9976 22.9976

**Выводы.**

Метод Гаусса-Зейделя наименее точный из всех. Для уточнения результатов метода Якоби и Гаусса-Зейделя необходимо больше 1000 итераций. Метод релаксации с параметром релаксации w=1.5 наиболее точный и требует меньше всего итераций.