**Міністерство освіти та науки України**

**Національний технічний університет України**

**“Київський політехнічний інститут імені Ігоря Сікорського”**

**Факультет прикладної математики**

**Кафедра системного програмування і спеціалізованих комп’ютерних систем**

**ЛАБОРАТОРНА РОБОТА №»3**

з дисципліни

“ Програмування”

Тема: «Опис масивів. Робота з масивами»

Виконав:Кулик Н.Я.

Студент групи КВ-31

Варіант 11

Перевірив(ла): Кучмій О.О.

**Київ 2023**

**Постановка задачі**

1.Заповнити квадратну матрицю А={aij}, aij=f(i,j) при (i, j=0, …, n-1), де f(i,j) – задана функція (таблиця 1).

2.Взяти за вхідні дані елементи матриці А, за вказаною умовою отримати елементи вектора Х={х0, х1 , ..., хn-1}.

3.Взявши як вхідні дані елементи вектора Х, обчислити значення функції U=g(x), де g(x) – задана функція (таблиця 2).

**Умова отримання вектора Х за матрицею А**

11.За елементи вектора *Х* від *0-го* до *(n-2)-го* взяти мінімальні елементи відповідних стовпців матриці *А*. За останній *хn-1*-ий елемент взяти суму елементів останнього стовпця матриці.

**Варіант 11**

Обчислити: 

Текст програми

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

double f(int i, int j, int n)

{

return 4 \* fabs(8.5 - i) - 1.5 \* j \* j + 2;

}

double g(double \*\*A, int n)

{

double max\_ratio = 0;

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

{

double min\_xj = fabs(A[i][0]);

for (int j = 0; j < n - 1; j++)

{

double abs\_xj = fabs(A[i][j]);

if (abs\_xj < min\_xj)

{

min\_xj = abs\_xj;

}

}

double abs\_xi = fabs(A[i][i]);

double ratio = min\_xj / (abs\_xi + 0.5);

if (ratio > max\_ratio)

{

max\_ratio = ratio;

}

}

return max\_ratio;

}

int main()

{

int n;

printf("Enter the size of the matrix and vector: ");

scanf("%d", &n);

double \*\*A = (double \*\*)malloc(n \* sizeof(double \*));

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

{

A[i] = (double \*)malloc(n \* sizeof(double));

}

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

{

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

{

A[i][j] = f(i, j, n);

}

}

printf("Matrix A:\n");

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

{

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

{

printf("%.2f\t", A[i][j]);

}

printf("\n");

}

double \*X = (double \*)malloc(n \* sizeof(double));

for (int j = 0; j < n - 1; j++)

{

double min\_xj = fabs(A[0][j]);

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

{

double abs\_xj = (A[i][j]);

if (abs\_xj < min\_xj)

{

min\_xj = abs\_xj;

}

}

X[j] = min\_xj;

}

double sum\_last\_column = 0;

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

{

sum\_last\_column += A[i][n - 1];

}

X[n - 1] = sum\_last\_column;

printf("\nVector X:\n");

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

{

printf("%.2f\t", X[i]);

}

double max\_min\_ratio\_result = g(A, n);

printf("\n\nMax{(min{|xj|})/(|xi+0.5|)} = %.2f\n", max\_min\_ratio\_result);

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

{

free(A[i]);

}

free(A);

free(X);

return 0;

}

Тестування:

