Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №3

з дисципліни

«Алгоритми і структури даних»

Виконала:

студентка групи ІМ-21

Рабійчук Дар’я Олександрівна

номер у списку групи: 18

Перевірила:

Молчанова А.А.

Київ 2023

Постановка задачі:

1. Представити у програмі напрямлений і ненапрямлений графи з заданими параметрами: — число вершин n; — розміщення вершин; — матриця суміжності A. Параметри задаються на основі номера групи, представленого десятковими цифрами п1, п2 та номера студента у списку групи — десяткового числа п3, п4. Число вершин п дорівнює 10 + п3. Розміщення вершин: — колом при п4 = 0,1; — прямокутником (квадратом) при п4 = 2,3; — трикутником при п4 = 4,5; — колом з вершиною в центрі при п4 = 6,7; — прямокутником (квадратом) з вершиною в центрі при п4 = 8,9. Наприклад, при п4 = 10 розміщення вершин прямокутником з вершиною в центрі повинно виглядати так, як на прикладі графа рис.4. Матриця A напрямленого графа за варіантом формується за функціями: srand(n1 n2 n3 n4); T = randm(n,n);

A = mulmr((1.0 - n3*0.02 - n4*0.005 - 0.25),T); де randm(n,n) – розроблена функція, яка формує матрицю розміром n·n, що складається з випадкових чисел у діапазоні (0, 2.0); mulmr() — розроблена функція множення матриці на коефіцієнт та округлення результату до 0 чи 1 (0, якщо результат менший за 1.0 і 1 — якщо більший за 1.0).

2. Створити програму для формування зображення напрямленого і ненапрямленого графів у графічному вікні.

Варіант 18 :

N1 = 2

N2 = 1

N3 = 1

N4 = 8

Число вершин n дорівнює 11

Текст програми напрямленого графа:

```
#include<windows.h>
#include<math.h>
#include <stdio.h>
#include <stdlib.h>

double** random(int rows, int cols){
    double** matrix = (double**)malloc(rows * sizeof(double*));

    for (int i = 0; i < rows; i++)
        matrix[i] = (double*)malloc(cols * sizeof(double));

    for (int i = 0; i < rows; i++){
        for (int j = 0; j < cols; j++){
          double temp = rand() % 21;
                        matrix[i][j]=temp/10;
        }
    }

    return matrix;
}

double** mulmr(double num, double **mat, int rows, int cols){
    for (int i = 0; i < rows; i++){
        for (int j = 0; j < cols; j++){
            mat[i][j] = mat[i][j] * num;

            if(mat[i][j] > 1.0){
                mat[i][j] = 1;
```

```
            } else mat[i][j] = 0;
        }
    }

    return mat;
}

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

char ProgName[]="P›P°P±PsCЪP°C,PsCЪPSP° CЪPsP±PsC,P° 3";

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpszCmdLine, int
nCmdShow){
    HWND hWnd;
    MSG lpMsg;

    WNDCLASS w;
    w.lpszClassName=ProgName;
    w.hInstance=hInstance;
    w.lpfnWndProc=WndProc;
    w.hCursor=LoadCursor(NULL, IDC_ARROW);
    w.hIcon=0;
    w.lpszMenuName=0;
    w.hbrBackground = WHITE_BRUSH;
    w.style=CS_HREDRAW|CS_VREDRAW;
    w.cbClsExtra=0;
    w.cbWndExtra=0;

    if(!RegisterClass(&w))
        return 0;

    hWnd = CreateWindow(ProgName,
        "LAB3 RABIICHUK IM-21",
        WS_OVERLAPPEDWINDOW,
        0,
        0,
        600,
        600,
        (HWND)NULL,
        (HMENU)NULL,
        (HINSTANCE)hInstance,
        (HINSTANCE)NULL);

    ShowWindow(hWnd, nCmdShow);

    while(GetMessage(&lpMsg, hWnd, 0, 0)) {
            TranslateMessage(&lpMsg);
            DispatchMessage(&lpMsg);
        }
    return(lpMsg.wParam);
    }

LRESULT CALLBACK WndProc(HWND hWnd, UINT messg, WPARAM wParam, LPARAM lParam){
    HDC hdc;
    PAINTSTRUCT ps;

    void arrow(float fi, int px,int py){
            fi = 3.1416*(180.0 - fi)/180.0;
            int lx,ly,rx,ry;
            lx = px+15*cos(fi+0.3);
            rx = px+15*cos(fi-0.3);
            ly = py+15*sin(fi+0.3);
            ry = py+15*sin(fi-0.3);
            MoveToEx(hdc, lx, ly, NULL);
            LineTo(hdc, px, py);
            LineTo(hdc, rx, ry);
      }

    switch(messg){

        case WM_PAINT :

            hdc=BeginPaint(hWnd, &ps);
            int n = 11;
            char *nn[11] = {"1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11"};
            int nx[11] = {};
            int ny[11] = {};
            int num = 120;
             for(int i = 0; i < n; i++)
        {
```

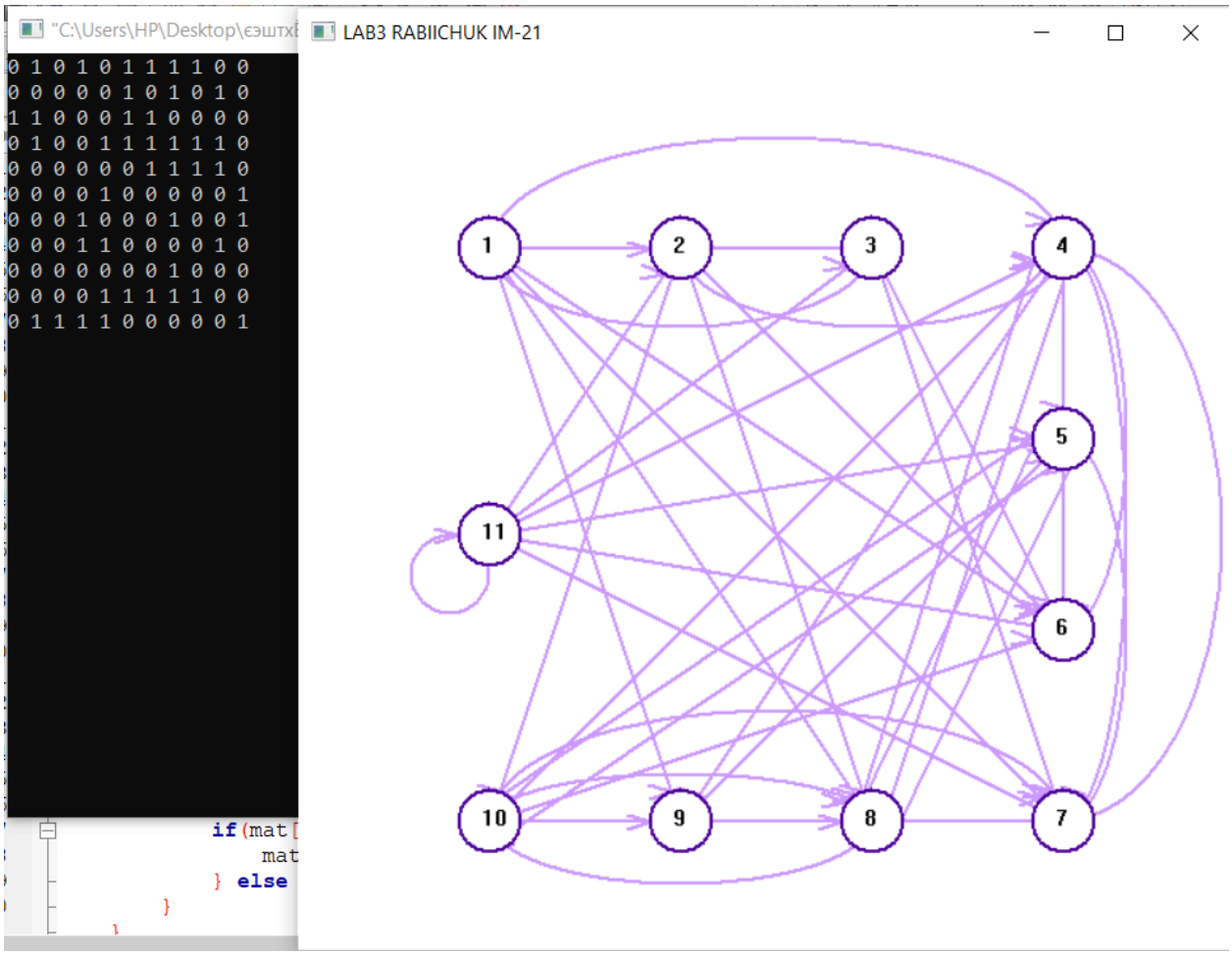
```
            if(i == 0)
            {
                nx[i] = num;
                ny[i] = num;
            } else if(i < 4)
            {
                nx[i] = nx[i - 1] + num;
                ny[i] = ny[i - 1];
            } else if(i < 7)
            {
                nx[i] = nx[i - 1];
                ny[i] = ny[i - 1] + num;
            } else if(i < 10)
            {
                nx[i] = nx[i - 1] - num;
                ny[i] = ny[i - 1];
            } else
            {
            nx[i] = nx[i - 1];
            ny[i] = ny[i - 1] - num*1.5;
            }
        }
            int dx = 20, dy = 20, dtx = 5;
            int i;
            HPEN BPen = CreatePen(PS_SOLID, 2, RGB(76, 0, 153));
            HPEN KPen = CreatePen(PS_SOLID, 2, RGB(204, 153, 255));

        srand(2118);
        double** T = random(11, 11);
        double coefficient = 1.0 - 1*0.02 - 3*0.005 - 0.25;
        double** A = mulmr(coefficient, T, 11, 11);

             for (int i = 0; i < 11; i++){
            for (int j = 0; j < 11; j++){
                printf("%g ",A[i][j]);
            }
            printf("\n");

        }

SelectObject(hdc, KPen);

        for(int i = 0; i < n; i++){
            for(int j = 0; j < n; j++){
                if(A[i][j] == 1){
                    MoveToEx(hdc, nx[i], ny[i], NULL);

                    if(i == j){
                        if(i < n*0.25){
                            Arc(hdc, nx[j], ny[j], nx[j]-50, ny[j]-50, nx[j], ny[j], nx[j],
ny[j]);
                            arrow((-90*3.1416)/180, nx[j], ny[j]-dy);
                        }
                            else if(i < n*0.5){
                            Arc(hdc, nx[j], ny[j], nx[j]+50, ny[j]-50, nx[j], ny[j], nx[j],
ny[j]);
                            arrow((0*3.1416)/180, nx[j]+dx, ny[j]);
                        }
                            else if(i < n*0.75){
                            Arc(hdc, nx[j], ny[j], nx[j]+50, ny[j]+50, nx[j], ny[j], nx[j],
ny[j]);
                            arrow((90*3.1416)/180, nx[j], ny[j]+dy);
                        }
                            else{
                            Arc(hdc, nx[j], ny[j], nx[j]-50, ny[j]+50, nx[j], ny[j], nx[j],
ny[j]);
                            arrow((180*3.1416)/180, nx[j]-dx, ny[j]);
                        }
                    }
                    if((ny[i] == ny[j]) && (nx[j] != nx[i] + num) && (nx[j] != nx[i] - num)){
                        if(i <= 4){
                            if(nx[i] < nx[j]){
                                if(nx[i]+3*num==nx[j]){
                                        Arc(hdc, nx[i], ny[i]-70, nx[j], ny[j]+70, nx[j], ny[j],
nx[i], ny[i]);
                                        arrow((-140*3.1416)/180, nx[j]-16*cos(-45),
ny[j]+16*sin(-45)-3);
                                }else{
                                 Arc(hdc, nx[i], ny[i]-50, nx[j], ny[j]+50, nx[j], ny[j], nx[i],
ny[i]);
```

```
                                    arrow((-145*3.1416)/180, nx[j]-16*cos(-45)-2, ny[j]+16*sin(-45)-
2);
                                }

                            }
                            else if(nx[i] > nx[j]){
                                if(nx[i]==nx[j]+3*num){
                                    Arc(hdc, nx[i], ny[i]-70, nx[j], ny[j]+70, nx[j], ny[j],
nx[i], ny[i]);
                                arrow((40*3.1416)/180, nx[j]+dx*cos(-145)-7, ny[j]+dy*sin(-
145)+27);
                                }else{
                                 Arc(hdc, nx[i], ny[i]-50, nx[j], ny[j]+50, nx[j], ny[j], nx[i],
ny[i]);
                                arrow((25*3.1416)/180, nx[j]+dx*cos(-145)-7, ny[j]+dy*sin(-
145)+25);
                                }
                            }
                        }
                        else if(i >= n*0.5 && i <= 10){
                            if(nx[i] < nx[j]){
                                if((nx[i]+3*num == nx[j])||(nx[i] == nx[j]+3*num)){
                                    Arc(hdc, nx[i], ny[i]-70, nx[j], ny[j]+70, nx[j], ny[j],
nx[i], ny[i]);
                                arrow((-140*3.1416)/180, nx[j]+dx*cos(-145)-28, ny[j]+dy*sin(-
145)-8);
                                }else{
                                Arc(hdc, nx[i], ny[i]-30, nx[j], ny[j]+30, nx[j], ny[j], nx[i],
ny[i]);
                                arrow((-160*3.1416)/180, nx[j]+dx*cos(-145)-34, ny[j]+dy*sin(-
145)-2);
                                }
                            }
                            else if(nx[i] > nx[j]){
                                Arc(hdc, nx[i], ny[i]-40, nx[j], ny[j]+40, nx[j], ny[j], nx[i],
ny[i]);
                                if((nx[i]+3*num == nx[j])||(nx[i] == nx[j]+3*num)){
                                    arrow((20*3.1416)/180, nx[j]+dx*cos(-145)-2, ny[j]+dy*sin(-
145)+22);
                                }else{
                                arrow((25*3.1416)/180, nx[j]+dx*cos(-145)-2, ny[j]+dy*sin(-
145)+24);
                                }
                            }
                        }
                    }
                    else if(((nx[i] == nx[j]) && (ny[j] != ny[i] + num) && (ny[j] != ny[i] -
num)&&(nx[i]==num || nx[i]== num*4))&&
                            ((nx[i] == nx[j]) && (ny[j] != ny[i] + num*1.5) && (ny[j] != ny[i] -
num*1.5))){


                        if(i >= n*0.25 && i <= 7){
                            if(ny[i] < ny[j]){
                                Arc(hdc, nx[i]-40, ny[i], nx[j]+40, ny[j], nx[j], ny[j], nx[i],
ny[i]);
                                if(ny[i]+2*num == ny[j]){
                                    arrow((-70*3.1416)/180, nx[j]+dx*cos(-145)-2, ny[j]+dy*sin(-
145)-5);
                                }else{
                                 arrow((-70*3.1416)/180, nx[j]+dx*cos(-145)-4, ny[j]+dy*sin(-
145)-7);
                                }
                            }
                            else if(ny[i] > ny[j]){

                                Arc(hdc, nx[j]-100, ny[j], nx[i]+100, ny[i], nx[i], ny[i], nx[j],
ny[j]);
                                if(ny[i] == ny[j]+2*num){
                                arrow((30*3.1416)/180, nx[j]+dx*cos(-145)+1, ny[j]+dy*sin(-
145)+13);
                                }else{
                                arrow((40*3.1416)/180, nx[j]+dx*cos(-145)+2, ny[j]+dy*sin(-
145)+15);
                                }
                            }
                        }
                        else if((i >= 9)||(j >= 9)){
                            if(ny[i] < ny[j]){
                                Arc(hdc, nx[j]-80, ny[j], nx[i]+80, ny[i], nx[i], ny[i], nx[j],
```

```
ny[j]);
                                    arrow((-130*3.1416)/180, nx[j]+dx*cos(-145)-36, ny[j]+dy*sin(-
145)+2);
                                }
                                else if(ny[i] > ny[j]){
                                    Arc(hdc, nx[i]-40, ny[i], nx[j]+40, ny[j], nx[j], ny[j], nx[i],
ny[i]);
                                    arrow((110*3.1416)/180, nx[j]+dx*cos(-145)-30, ny[j]+dy*sin(-
145)+27);
                                }
                            }
                        }
                        else{
                            double fi = 3.141  + acos((nx[j]-nx[i])/(sqrt((nx[j]-nx[i])*(nx[j]-
nx[i])+(ny[j]-ny[i])*(ny[j]-ny[i]))));
                            if(ny[j] < ny[i]) fi *= -1;

                            if(A[i][j] == A[j][i] && i < j){
                                if((ny[i]+3*num == ny[j])||(ny[i] == ny[j]+3*num)){
                                MoveToEx(hdc, nx[i]+5, ny[i]+5, NULL);
                                LineTo(hdc, nx[j]+5, ny[j]+5);
                                if(nx[i]==nx[j]+3*num){
                                arrow(fi, nx[j]+dx*cos(fi)+5, ny[j]+dy*sin(fi)+7);
                                }else{
                                    if(nx[i]==nx[j]){
                                     arrow(fi, nx[j]+dx*cos(fi)+7, ny[j]+dy*sin(fi));
                                    }else{
                                        if(nx[i]<nx[j]){
                                            arrow(fi, nx[j]+dx*cos(fi)+2, ny[j]+dy*sin(fi));
                                        }else{
                                            arrow(fi, nx[j]+dx*cos(fi)+6, ny[j]+dy*sin(fi)+3);
                                        }
                                    }
                                }
                                }else{
                                MoveToEx(hdc, nx[i]+10, ny[i]+5, NULL);
                                LineTo(hdc, nx[j]+15, ny[j]+5);
                                if(ny[i]+num == ny[j]){
                                arrow(fi+0.1, nx[j]+dx*cos(fi)+5, ny[j]+15+dy*sin(fi));
                                }else if((ny[i]==ny[j]+num*1.5)||(ny[i]+num*1.5==ny[j])){
                                if(nx[i] == nx[j]){
                                arrow(fi, nx[j]+dx*cos(fi)+7, ny[j]+5+dy*sin(fi)-4);
                                }else{
                                    if(nx[i]==nx[j]+3*num){
                                    arrow(fi, nx[j]+dx*cos(fi)+2, ny[j]+dy*sin(fi)+12);
                                    }else{
                                     arrow(fi, nx[j]+dx*cos(fi)-2, ny[j]+dy*sin(fi)+2);
                                    }
                                }

                                }else{
                                    if((nx[i]==nx[j]+num)||(nx[i]+num==nx[j])){
                                    arrow(fi, nx[j]+dx*cos(fi), ny[j]+5+dy*sin(fi));
                                    }else{
                                        if(ny[i]+2*num==ny[j]){
                                         arrow(fi, nx[j]+dx*cos(fi)+3, ny[j]+dy*sin(fi)+14);
                                        }else{
                                         arrow(fi, nx[j]+dx*cos(fi), ny[j]+3+dy*sin(fi));
                                        }
                                    }
                                }
                                }
                            }
                            else if(A[i][j] == A[j][i] && i > j){
                                if((ny[i]+3*num == ny[j])||(ny[i] == ny[j]+3*num)){
                                MoveToEx(hdc, nx[i]-10, ny[i]-5, NULL);
                                LineTo(hdc, nx[j]-15, ny[j]-5);
                                if(nx[i]+3*num==nx[j]){
                                arrow(fi, nx[j]+dx*cos(fi)-7, ny[j]-5+dy*sin(fi)-7);
                                }else{
                                    if(nx[i]==nx[j]){
                                    arrow(fi, nx[j]+dx*cos(fi)-14, ny[j]-5+dy*sin(fi));
                                    }else{
                                        if(nx[i]>nx[j]){
                                            arrow(fi, nx[j]+dx*cos(fi)-10, ny[j]-5+dy*sin(fi)+7);
                                        }else{
                                            arrow(fi, nx[j]+dx*cos(fi)-12, ny[j]-5+dy*sin(fi)-3);
                                        }
                                    }
                                }
```

```
                        }else{
                            MoveToEx(hdc, nx[i]-10, ny[i]-5, NULL);
                            LineTo(hdc, nx[j]-15, ny[j]-5);
                            if(ny[i] == ny[j]+num){
                            arrow(fi-0.1, nx[j]+dx*cos(fi)-5, ny[j]+dy*sin(fi)-15);
                            }else if((ny[i]==ny[j]+num*1.5)||(ny[i]+num*1.5==ny[j]) ){
                                if(nx[i]== nx[j]){
                            arrow(fi, nx[j]+dx*cos(fi)-4, ny[j]-5+dy*sin(fi)+5);
                                }else{
                                     if(nx[i]+3*num==nx[j]){
                            arrow(fi, nx[j]+dx*cos(fi)-4, ny[j]+dy*sin(fi)-10);
                                     }else{
                            arrow(fi, nx[j]+dx*cos(fi)+1, ny[j]-5+dy*sin(fi)+2);
                                     }
                                }
                            }else{
                                if((nx[i]==nx[j]+num)||(nx[i]+num==nx[j])){
                            arrow(fi, nx[j]+dx*cos(fi), ny[j]-5+dy*sin(fi));
                                }else{
                                    if(ny[i]==ny[j]+2*num){
                                     arrow(fi, nx[j]+dx*cos(fi)-2, ny[j]+dy*sin(fi)-13);
                                    }else{
                                     arrow(fi, nx[j]+dx*cos(fi), ny[j]-2+dy*sin(fi));
                                    }
                            }
                            }
                            }
                        }
                        else
                        {
                            LineTo(hdc, nx[j], ny[j]);
                            arrow(fi, nx[j]+dx*cos(fi)+1, ny[j]+dy*sin(fi));
                        }
                    }
                }
            }
        }

        SelectObject(hdc, BPen);
        for(i = 0; i < n; i++){
            Ellipse(hdc, nx[i]-dx, ny[i]-dy, nx[i]+dx, ny[i]+dy);
            if(i < 9){
                TextOut(hdc, nx[i]-dtx, ny[i]-dy/2, nn[i], 1);
            } else TextOut(hdc, nx[i]-dtx, ny[i]-dy/2, nn[i], 2);
        }

            EndPaint(hWnd, &ps);
            break;

        case WM_DESTROY:
            PostQuitMessage(0);
            break;

        default:
            return(DefWindowProc(hWnd, messg, wParam, lParam));
    }
    return 0;
}
```

Результат тестування програми:



Текст програми ненапрямленого графа:

```
#include<windows.h>
#include<math.h>

double** random(int rows, int cols){
    double** matrix = (double**)malloc(rows * sizeof(double*));

    for (int i = 0; i < rows; i++)
        matrix[i] = (double*)malloc(cols * sizeof(double));

    for (int i = 0; i < rows; i++){
        for (int j = 0; j < cols; j++){
                double temp = rand() % 21;
                        matrix[i][j]=temp/10;
        }
    }

    return matrix;
}

             double** mirrorMatrix(double **matrix, int rows, int cols){
                for (int i = 0; i < rows; i++)
                {
                    for (int j = 0; j < cols; j++)
                    {
                    if(matrix[i][j] == 1){
                            matrix[j][i]=1;
                        }
                        }
                }

                return matrix;
            }


           double** mulmr(double num, double **mat, int rows, int cols){
    for (int i = 0; i < rows; i++){
        for (int j = 0; j < cols; j++){
            mat[i][j] = mat[i][j] * num;

            if(mat[i][j] > 1.0){
                mat[i][j] = 0;
            } else mat[i][j] = 1;
        }
    }

    return mat;
}

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

char ProgName[]="P›P°P±PsCЂP°C,PsCЂPSP° CЂPsP±PsC,P° 3";

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpszCmdLine, int nCmdShow)
{
    HWND hWnd;
    MSG lpMsg;

    WNDCLASS w;
    w.lpszClassName=ProgName;
    w.hInstance=hInstance;
    w.lpfnWndProc=WndProc;
    w.hCursor=LoadCursor(NULL, IDC_ARROW);
    w.hIcon=0;
    w.lpszMenuName=0;
    w.hbrBackground = WHITE_BRUSH;
    w.style=CS_HREDRAW|CS_VREDRAW;
    w.cbClsExtra=0;
    w.cbWndExtra=0;

    if(!RegisterClass(&w))
        return 0;

    hWnd=CreateWindow(ProgName,
        "LAB3 RABIICHUK IM-21",
        WS_OVERLAPPEDWINDOW,
        0,
        0,
```

```
        600,
        600,
        (HWND)NULL,
        (HMENU)NULL,
        (HINSTANCE)hInstance,
        (HINSTANCE)NULL);

    ShowWindow(hWnd, nCmdShow);

    while(GetMessage(&lpMsg, hWnd, 0, 0)) {
            TranslateMessage(&lpMsg);
            DispatchMessage(&lpMsg);
        }
    return(lpMsg.wParam);
    }

LRESULT CALLBACK WndProc(HWND hWnd, UINT messg,
                         WPARAM wParam, LPARAM lParam){
    HDC hdc;
    PAINTSTRUCT ps;

    void arrow(float fi, int px,int py){
            fi = 3.1416*(180.0 - fi)/180.0;
            int lx,ly,rx,ry;
            lx = px+15*cos(fi+0.3);
            rx = px+15*cos(fi-0.3);
            ly = py+15*sin(fi+0.3);
            ry = py+15*sin(fi-0.3);
            MoveToEx(hdc, lx, ly, NULL);
            LineTo(hdc, px, py);
            LineTo(hdc, rx, ry);
      }

    switch(messg){

        case WM_PAINT :

            hdc=BeginPaint(hWnd, &ps);
            int n = 11;
            char *nn[11] = {"1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11"};
            int nx[11] = {};
            int ny[11] = {};
            int num = 120;
             for(int i = 0; i < n; i++){
            if(i == 0){
                nx[i] = num;
                ny[i] = num;
            } else if(i < 4){
                nx[i] = nx[i - 1] + num;
                ny[i] = ny[i - 1];
            } else if(i < 7){
                nx[i] = nx[i - 1];
                ny[i] = ny[i - 1] + num;
            } else if(i < 10){
                nx[i] = nx[i - 1] - num;
                ny[i] = ny[i - 1];
            } else{
            nx[i] = nx[i - 1];
            ny[i] = ny[i - 1] - num*1.5;
            }
        }
            int dx = 20, dy = 20, dtx = 5;
            int i;
            HPEN BPen = CreatePen(PS_SOLID, 2, RGB(76, 0, 153));
            HPEN KPen = CreatePen(PS_SOLID, 2, RGB(204, 153, 255));

        srand(2118);
        double** T = random(11, 11);
        double coefficient = 1.0 - 1*0.02 - 3*0.005 - 0.25;
        double** A = mulmr(coefficient, T, 11, 11);

             for (int i = 0; i < 11; i++){
            for (int j = 0; j < 11; j++){
                printf("%g ",A[i][j]);
            }
            printf("\n");

        }

SelectObject(hdc, KPen);
```

```
        for(int i = 0; i < 11; i++){
            for(int j = 0; j < 11; j++){
                if(A[i][j] == 1){
                    MoveToEx(hdc, nx[i], ny[i], NULL);

                    if((ny[i] == ny[j]) && (nx[j] != nx[i] + num) && (nx[j] != nx[i] - num)){

                        if(i < 4){
                            if(nx[i] < nx[j]){
                                Arc(hdc, nx[i], ny[i]-50, nx[j], ny[j]+50, nx[j], ny[j], nx[i],
ny[i]);
                            }
                            else if(nx[i] > nx[j]){
                                Arc(hdc, nx[j], ny[j]-40, nx[i], ny[i]+40, nx[i], ny[i], nx[j],
ny[j]);
                            }
                        }
                        else if(i > 5 && i < 10){
                            if(nx[i] < nx[j]){
                                Arc(hdc, nx[j], ny[j]-50, nx[i], ny[i]+50, nx[i], ny[i], nx[j],
ny[j]);
                            }
                            else if(nx[i] > nx[j]){
                                Arc(hdc, nx[i], ny[i]-40, nx[j], ny[j]+40, nx[j], ny[j], nx[i],
ny[i]);
                            }
                        }

                    }
                    else if(((nx[i] == nx[j]) && (ny[j] != ny[i] + num) && (ny[j] != ny[i] -
num)&&(nx[i]==num || nx[i]== num*4))&&
                            ((nx[i] == nx[j]) && (ny[j] != ny[i] + num*1.5) && (ny[j] != ny[i] -
num*1.5) )){

                        if(i > 2 && i < 7){
                            if(ny[i] < ny[j]){
                                Arc(hdc, nx[i]-40, ny[i], nx[j]+40, ny[j], nx[j], ny[j], nx[i],
ny[i]);
                            }
                            else if(ny[i] > ny[j]){
                                Arc(hdc, nx[j]-50, ny[j], nx[i]+50, ny[i], nx[i], ny[i], nx[j],
ny[j]);
                            }
                        }
                        else if(i > 8){
                            if(ny[i] < ny[j])
                            {
                                Arc(hdc, nx[j]-40, ny[j], nx[i]+40, ny[i], nx[i], ny[i], nx[j],
ny[j]);
                            }
                            else if(ny[i] > ny[j]){
                                Arc(hdc, nx[i]-40, ny[i], nx[j]+40, ny[j], nx[j], ny[j], nx[i],
ny[i]);
                            }
                        }

                    }
                    else{
                        LineTo(hdc, nx[j], ny[j]);
                    }
                }
            }
        }

          SelectObject(hdc, BPen);
        for(i = 0; i < n; i++){
            Ellipse(hdc, nx[i]-dx, ny[i]-dy, nx[i]+dx, ny[i]+dy);
            if(i < 9){
                TextOut(hdc, nx[i]-dtx, ny[i]-dy/2, nn[i], 1);
            } else TextOut(hdc, nx[i]-dtx, ny[i]-dy/2, nn[i], 2);
        }

            EndPaint(hWnd, &ps);
            break;

        case WM_DESTROY:
            PostQuitMessage(0);
            break;
```

```
        default:
            return(DefWindowProc(hWnd, messg, wParam, lParam));
    }
    return 0;
}
```

Результат тестування програми: