МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



**ОТЧЁТ**

**по лабораторной работе № 2**

**«Рисование линий и окружностей*»***

по дисциплине:

**«Компьютерная графика»**

|  |  |
| --- | --- |
| Выполнил(а):  Дашкин С.М. Группа АТ-34 | Проверил:  Смагин С.М. |

Новосибирск

2024

**ЦЕЛЬ РАБОТЫ**

Получить навыки применения алгоритмов дифференциального анализатора, Брезенхема для рисования линий и Брезенхема для рисования окружностей.

**Задание**

1. Разработать программу, которая будет рисовать окружность по алгоритму Брезенхема на сетке с заданными параметрами (как в Лабораторной работе №1).
2. Разработать программу, которая будет рисовать линию, построенную по алгоритму цифрового дифференциального анализатора и по алгоритму Брезенхема на сетке с заданными параметрами (как в Лабораторной работе №1).

**ХОД РАБОТЫ:**

1. **createGrid**:

- Создает сетку на экране.

- Пользователь вводит размеры сетки (ширину и высоту).

- Рисует вертикальные и горизонтальные линии, формирующие сетку.

2. **fillCell**:

- Закрашивает ячейку сетки в зависимости от переданного символа (`'r'` для прямоугольника, `'c'` для круга).

- Проверяет, что координаты ячейки находятся в пределах сетки.

3. **DDA**:

- Рисует линию с использованием алгоритма цифрового дифференциального анализатора (DDA).

- Пользователь вводит начальные и конечные координаты линии.

- Линия рисуется путем последовательного закрашивания ячеек сетки.

4. **BresenhamLine**:

- Рисует линию с использованием алгоритма Брезенхема.

- Пользователь вводит начальные и конечные координаты линии.

- Линия рисуется путем последовательного закрашивания ячеек сетки.

5. **BresenhamCircle**:

- Рисует окружность с использованием алгоритма Брезенхема.

- Пользователь вводит координаты центра и радиус окружности.

- Окружность рисуется путем последовательного закрашивания ячеек сетки.

6. **main**:

- Основная функция программы.

- Устанавливает локаль для русского языка.

- Получает дескриптор окна консоли и контекст устройства (HDC).

- Создает перо и кисть для рисования.

- Вызывает функцию `createGrid` для создания сетки.

- Пользователь выбирает действие: рисование линии по DDA, линии по Брезенхему или окружности по Брезенхему.

- В зависимости от выбора пользователя вызывает соответствующую функцию.

- Освобождает ресурсы и ждет ввода пользователя перед завершением программы.

**ЛИСТИНГ:**

#include <windows.h>

#include <windowsx.h>

#include <locale.h>

#include <iostream>

#include <cmath>

#define X\_OFFSET 250

#define Y\_OFFSET 80

#define CELL\_SIDE\_LEN 17

using namespace std;

void createGrid(HDC hdc, int& width, int& height) {

cout << "Введите размер сетки(ширина высота): ";

cin >> width >> height;

for (int i = 1; i <= width; ++i) {

MoveToEx(hdc, X\_OFFSET + i \* CELL\_SIDE\_LEN, Y\_OFFSET, NULL);

LineTo(hdc, X\_OFFSET + i \* CELL\_SIDE\_LEN, Y\_OFFSET + height \* CELL\_SIDE\_LEN);

}

for (int i = 0; i <= height; ++i) {

MoveToEx(hdc, X\_OFFSET, Y\_OFFSET + i \* CELL\_SIDE\_LEN, NULL);

LineTo(hdc, X\_OFFSET + width \* CELL\_SIDE\_LEN, Y\_OFFSET + i \* CELL\_SIDE\_LEN);

}

}

void fillCell(HDC hdc, int width, int height, int x, int y, char shape) {

if (x < 1 || x > width || y < 1 || y > height) return;

int cellX = X\_OFFSET + (x - 1) \* CELL\_SIDE\_LEN;

int cellY = Y\_OFFSET + (height - y) \* CELL\_SIDE\_LEN;

if (shape == 'r') {

Rectangle(hdc, cellX, cellY, cellX + CELL\_SIDE\_LEN, cellY + CELL\_SIDE\_LEN);

}

else if (shape == 'c') {

Ellipse(hdc, cellX, cellY, cellX + CELL\_SIDE\_LEN, cellY + CELL\_SIDE\_LEN);

}

}

void DDA(HDC hdc, int width, int height) {

int x1, y1, x2, y2;

cout << "Введите начальные и конечные точки линии(x1 y1 x2 y2): ";

cin >> x1 >> y1 >> x2 >> y2;

SetDCBrushColor(hdc, RGB(255, 255, 0));

double Dx, Dy, Length;

if (abs(x2 - x1) >= abs(y2 - y1)) {

Length = abs(x2 - x1);

}

else {

Length = abs(y2 - y1);

}

Dx = (x2 - x1) / Length;

Dy = (y2 - y1) / Length;

double x = x1;

double y = y1;

// Рисуем начальную точку

fillCell(hdc, width, height, static\_cast<int>(x), static\_cast<int>(y), 'c');

int i = 1;

while (i <= Length) {

x += Dx;

y += Dy;

fillCell(hdc, width, height, static\_cast<int>(x), static\_cast<int>(y), 'c');

i++;

}

}

void BresenhamLine(HDC hdc, int x0, int y0, int x1, int y1, int c, int width, int height) {

int dx, dy, ch = 0, e, dx2, dy2;

int x, y;

dx = abs(x1 - x0);

dy = abs(y1 - y0);

if (!dx && !dy) return; // Если начало совпадает с концом отрезка

int sx = (x0 < x1) ? 1 : -1;

int sy = (y0 < y1) ? 1 : -1;

if (dy > dx) {

int t = dy;

dy = dx;

dx = t;

ch = 1;

} // меняем местами x и y

dx2 = dx << 1;

dy2 = dy << 1; // dx2 = 2\*dx; dy2 = 2\*dy;

e = dy2 - dx; // Начальное значение ошибки;

x = x0;

y = y0;

while (x != x1 || y != y1) {

fillCell(hdc, width, height, x, y, 'r');

if (e > 0) {

if (ch) x += sx;

else y += sy;

e -= dx2;

}

else {

if (ch) y += sy;

else x += sx;

e += dy2;

}

}

fillCell(hdc, width, height, x1, y1, 'r'); // Убедимся, что конечная точка закрашена

}

void BresenhamCircle(HDC hdc, int width, int height) {

int cx, cy, r;

cout << "Введите центр и радиус окружности(cx cy r): ";

cin >> cx >> cy >> r;

SetDCBrushColor(hdc, RGB(0, 0, 255));

int x = 0;

int y = r;

int d = 3 - 2 \* r;

while (x <= y) {

fillCell(hdc, width, height, cx + x, cy + y, 'c');

fillCell(hdc, width, height, cx + x, cy - y, 'c');

fillCell(hdc, width, height, cx - x, cy + y, 'c');

fillCell(hdc, width, height, cx - x, cy - y, 'c');

fillCell(hdc, width, height, cx + y, cy + x, 'c');

fillCell(hdc, width, height, cx + y, cy - x, 'c');

fillCell(hdc, width, height, cx - y, cy + x, 'c');

fillCell(hdc, width, height, cx - y, cy - x, 'c');

if (d < 0) {

d += 4 \* x + 6;

}

else {

d += 4 \* (x - y) + 10;

y--;

}

x++;

}

}

int main() {

setlocale(LC\_ALL, "Rus");

HWND hwnd = GetConsoleWindow();

HDC hdc = GetDC(hwnd);

HPEN pen = CreatePen(PS\_SOLID, 1, RGB(255, 255, 255));

HBRUSH brush = (HBRUSH)::GetStockObject(DC\_BRUSH);

SelectObject(hdc, pen);

SelectObject(hdc, brush);

int width, height, function;

createGrid(hdc, width, height);

cout << "Выберите действие: 1)Нарисовать линию по ДДА; 2)Нарисовать линию по алгоритму Брезенхэма; 3)Нарисовать круг по алгоритму Брезенхэма: ";

cin >> function;

switch (function) {

case 1:

DDA(hdc, width, height);

break;

case 2:

int x1, y1, x2, y2;

cout << "Введите начальные и конечные точки линии(x1 y1 x2 y2): ";

cin >> x1 >> y1 >> x2 >> y2;

BresenhamLine(hdc, x1, y1, x2, y2, RGB(255, 0, 0), width, height);

break;

case 3:

BresenhamCircle(hdc, width, height);

break;

default:

cout << "Функция выбрана неправильно" << endl;

}

DeleteObject(pen);

DeleteObject(brush);

ReleaseDC(hwnd, hdc);

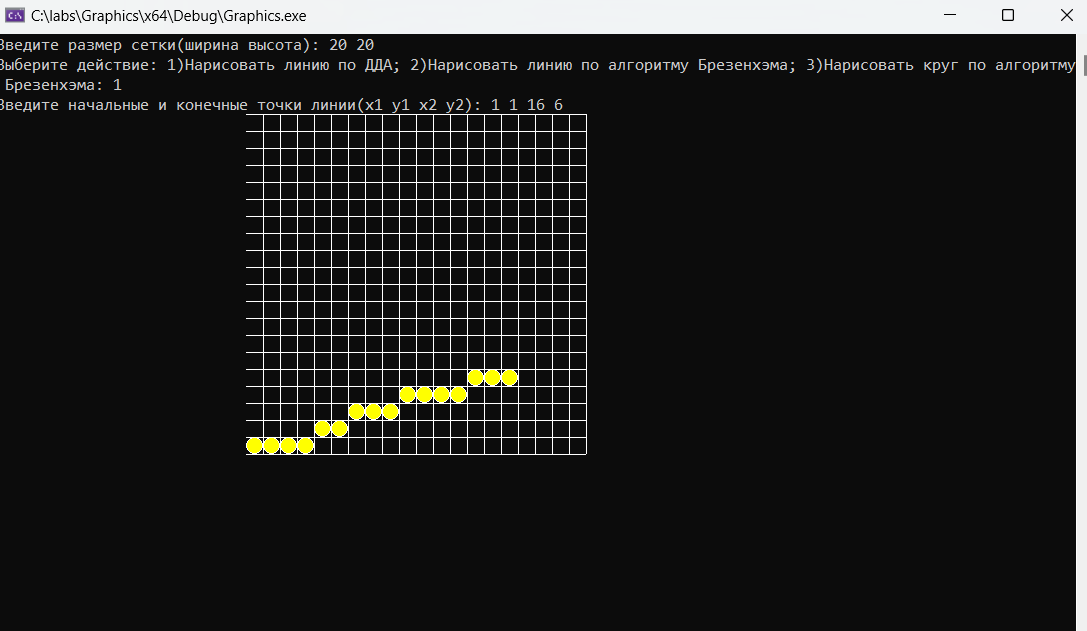
int pause;

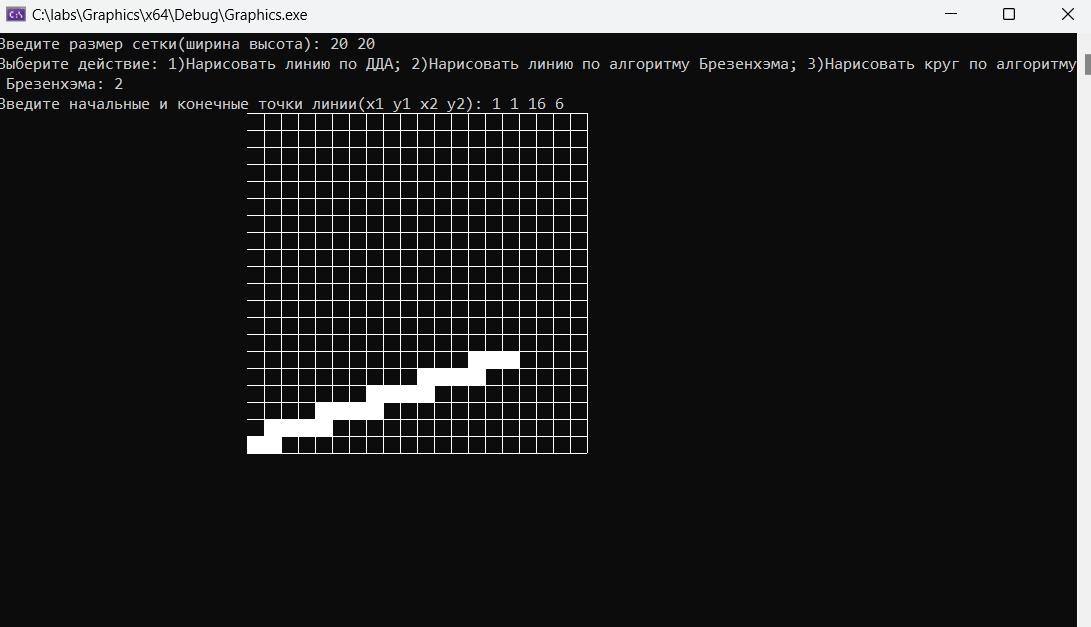
cin >> pause;

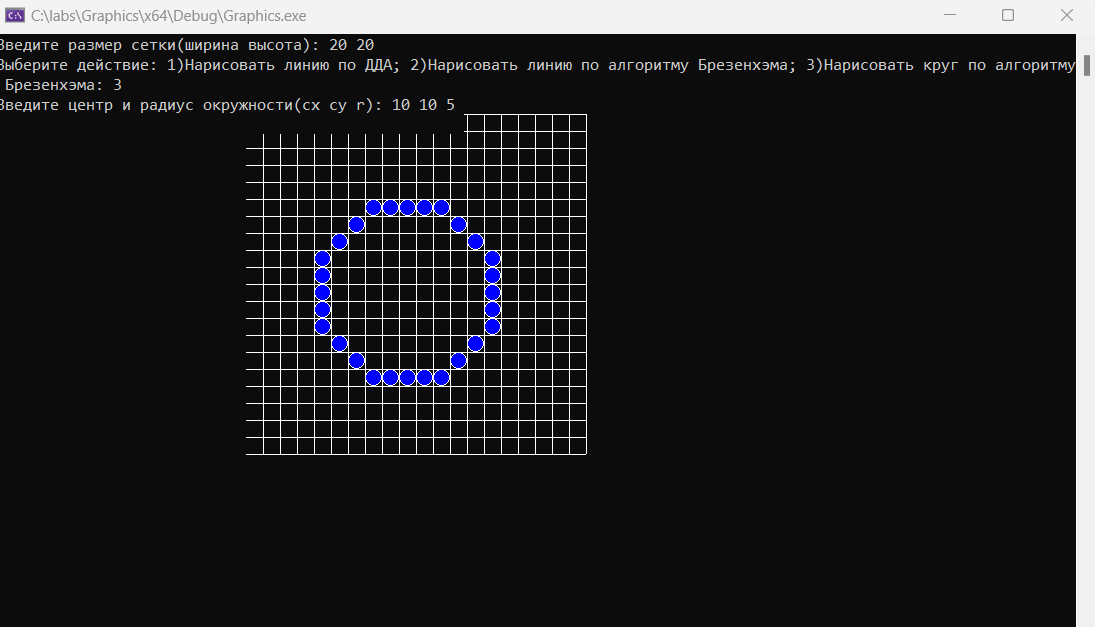
return 0;

}

**ТЕСТИРОВАНИЕ:**

****

****

****