МИНОБРНАУКИ РОССИИ САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра САПР

ОТЧЕТ

по практической работе № 3

по дисциплине «Программирование»

Тема: «Отладка»

Студент гр. 3351 _____

Преподаватель _____

Санкт-

Петербург

2023

Цель работы:

приобретение навыков работы в отладочном режиме и изучение команд отладчика.

Задание: &&&1

Пользователь задает натуральные числа:

r – радиус окружностей, **n** – исходное количество окружностей;

{X_1,Y_1}, …, {Xn, Yn} – последовательность задаваемых генератором случайных числе координат точек на экране, представляющих собой центры окружностей радиуса **r**.

Проверить на примере следующего задания работу отладчика Си++:

1. Подсчитать количество непересекающихся окружностей.
2. Сформировать из исходной последовательности новую последовательность координат центров окружностей, которые не имеют общих точек с другими окружностями. Вывести результат на экран.
3. Нарисовать на экране непересекающиеся окружности, соединив их каждую с каждой линиями кратчайшей длины. Линии должны быть нарисованы от края окружности, а не от центра. Если линия проходит через другую окружность, то ее участок, проходящий внутри окружности, не отображается.
4. В разделе Руководство пользователя помимо окна результатов приведите скриншоты окна программы с установленными точками останова, а также окна Контрольные значения со значениями произвольно выбранных переменных.

Требование: для задания исходной и новой последовательности координат точек сформировать массивы. На примере этой программы изучите различные режимы и команды отладчика.

Текст программы «Отображение окружностей и откладка»

```
#include <SFML/Graphics.hpp>
#include <iostream>
#include <cstdlib>
using namespace std;

// Максимальное количество окружностей, которое можно сгенерировать
const int maxCircles = 999;
// Максимальные значения случайных координат для генерации центров окружностей
const int maxRandom = 600;
const int minRandom = 200;
const int windowSize = 1000;// Размер окна SFML, в котором будут отображаться окружности
const sf::Color backgroundColor = sf::Color::White;// Цвет заднего фона окна SFML
const sf::Color lineColor = sf::Color::Black;// Цвет линий, соединяющих центры непересекающихся окружностей
const sf::Color fillColor = sf::Color::White;// Цвет заливки непересекающихся окружностей
const sf::Color outlineColor = sf::Color::Black;// Цвет контура непересекающихся окружностей
const int outlineThickness = 1; // Толщина контура непересекающихся окружностей


int main() {
   setlocale(LC_ALL, "rus");
   srand(time(NULL)); // Инициализация генератора случайных чисел

   // Переменные для ввода пользователя и отслеживания новых окружностей
   int radius, numCircles, newNumCircles = 0;
   int centr[maxCircles][2];
   cout << endl << "Введите количество окружностей: ";
   cin >> numCircles;
   cout << "Введите радиус данных окружностей: ";
   cin >> radius;

   // Динамическое выделение памяти для хранения центров окружностей
   int** circleCenters = new int* [numCircles];
   for (int i = 0; i < numCircles; i++) {
      //Указание на массив, который содержит две координаты (x и y) центра i-й окружности.
      circleCenters[i] = new int[2];
      // Генерация случайных координат x и y для центров окружностей
      circleCenters[i][0] = rand() % maxRandom + minRandom;
      circleCenters[i][1] = rand() % maxRandom + minRandom;
   }
   // Вывод сгенерированных центров окружностей
   cout << "Сгенерированные центры окружностей: " << endl;
   for (int i = 0; i < numCircles; i++) {
      cout << circleCenters[i][0] << "\t" << circleCenters[i][1] << endl;
   }
   // Проверка на пересекающиеся окружности и сохранение непересекающихся
   for (int i = 0, j = 0; i < numCircles; i++) {
      bool intersects = false;
      int x_1 = circleCenters[i][0], y_1 = circleCenters[i][1];
      double distance;

      // Проверка на пересечение с другими окружностями
      for (int k = 0; k < numCircles; k++) {
         if (k == i) continue;
         int x_2 = circleCenters[k][0], y_2 = circleCenters[k][1];
         distance = sqrt(pow(x_2 - x_1, 2) + pow(y_2 - y_1, 2));
         if (distance <= 2 * radius) {
```

```
            intersects = true;
        }
    }

    // Сохранение центров непересекающихся окружностей
    for (; !intersects; j++) {
        //Это бесконечный цикл (for (;;)), который выполняется до тех пор, пока intersects равно false
        centr[j][0] = circleCenters[i][0];
        centr[j][1] = circleCenters[i][1];
        newNumCircles++;
        intersects = true;
    }
}

// Вывод центров непересекающихся окружностей
cout << "Центры непересекающихся окружностей: " << endl;
for (int i = 0; i < newNumCircles; i++) {
    cout << centr[i][0] << "\t" << centr[i][1] << endl;
}

// Создание окна SFML
sf::RenderWindow window(sf::VideoMode(windowSize, windowSize),
    L"Непересекающиеся окружности");

// Очистка окна с указанным цветом фона
window.clear(backgroundColor);

// Рисование линий между непересекающимися окружностями
for (int i = 0; i < newNumCircles - 1; i++) {
    for (int j = i + 1; j < newNumCircles; j++) {
        // Создание объекта для отрисовки линии между центрами окружностей
        sf::VertexArray line(sf::Lines, 2);
        // Установка цвета линии
        line[0].color = lineColor;
        line[1].color = lineColor;
        // Установка начальной точки линии в координаты центра первой окружности
        line[0].position = sf::Vector2f(centr[i][0], centr[i][1]);
        // Установка конечной точки линии в координаты центра второй окружности
        line[1].position = sf::Vector2f(centr[j][0], centr[j][1]);
        // Отрисовка линии в окне SFML
        window.draw(line);
    }
}
// Рисование непересекающихся окружностей
for (int i = 0; i < newNumCircles; i++) {
    sf::CircleShape circle(radius);// Создание объекта окружности
    circle.setPosition(centr[i][0] - radius, centr[i][1] - radius);// Создание объекта окружности
    circle.setFillColor(fillColor);  // Установка цвета заливки и контура окружности
    circle.setOutlineColor(outlineColor);
    circle.setOutlineThickness(outlineThickness);// Установка толщины контура окружности
    window.draw(circle); // Отрисовка окружности в окне
}


// Отображение содержимого окна
window.display();

// Обработка событий: окно остается открытым до закрытия
while (window.isOpen()) {
```

```
        sf::Event event;
        while (window.pollEvent(event)) {
            if (event.type == sf::Event::Closed) window.close();
        }
    }

    // Освобождение динамически выделенной памяти
    for (int i = 0; i < numCircles; i++) {
        delete[] circleCenters[i];
    }
    delete[] circleCenters;

    return 0;
}
```

Результат работы программы.

Программа запрашивает на ввод у пользователя радиус окружностей и их количество, после этого помещает значения центров в массив (рис.

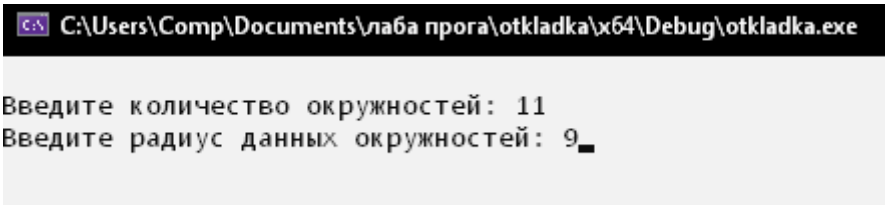

Рисунок 1

Далее программа ищет пересекающиеся окружности и удаляет их координаты центра из массива и выводит на экран исходную и новую последовательность из координат центров (рис. 2)

Рисунок 2

После рисуются непересекающиеся окружности, соединенные линиями (рис. 3)

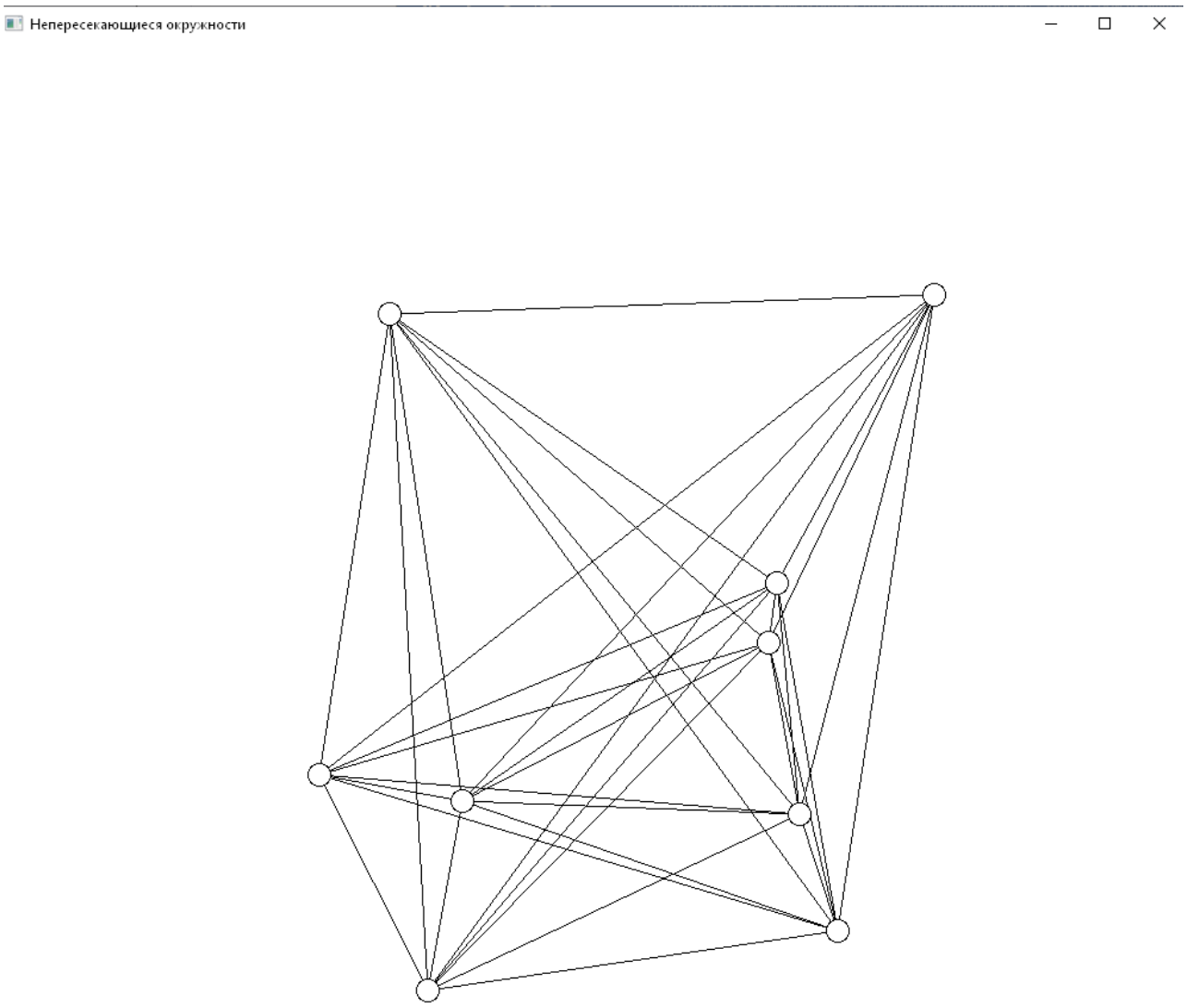

Рисунок 3

Работа с отладчиком:

- Скриншоты с установленными точками останова (рис. 4, рис. 5)

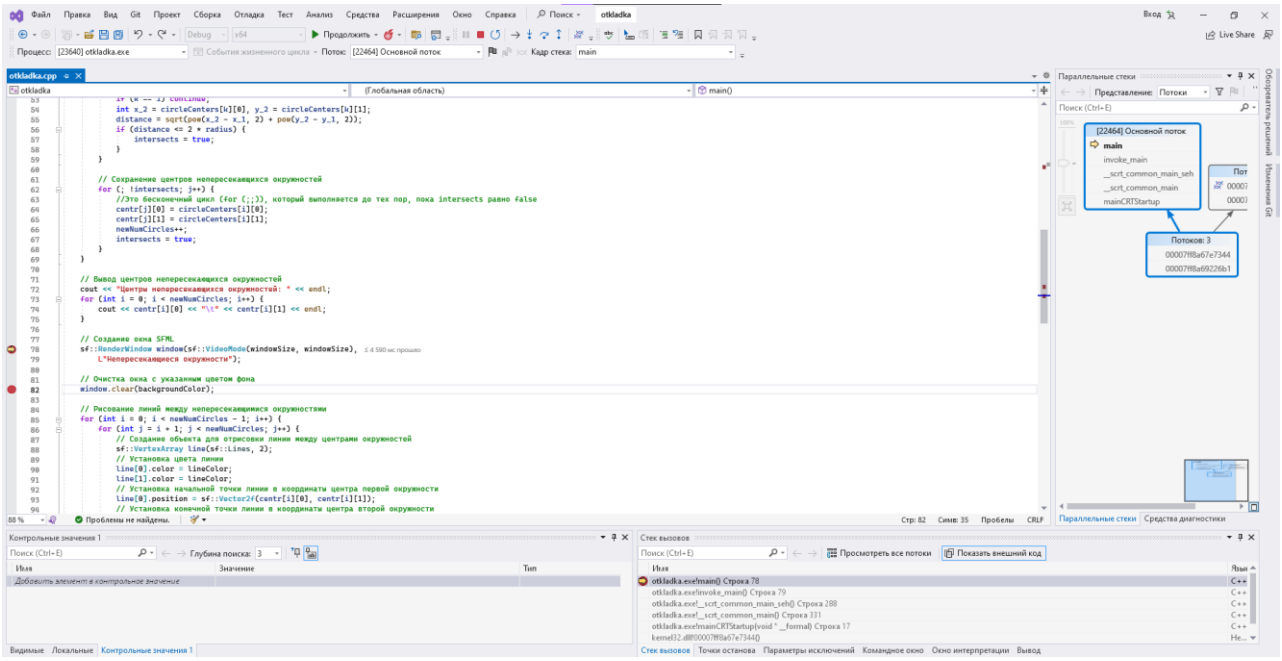

Рисунок 4

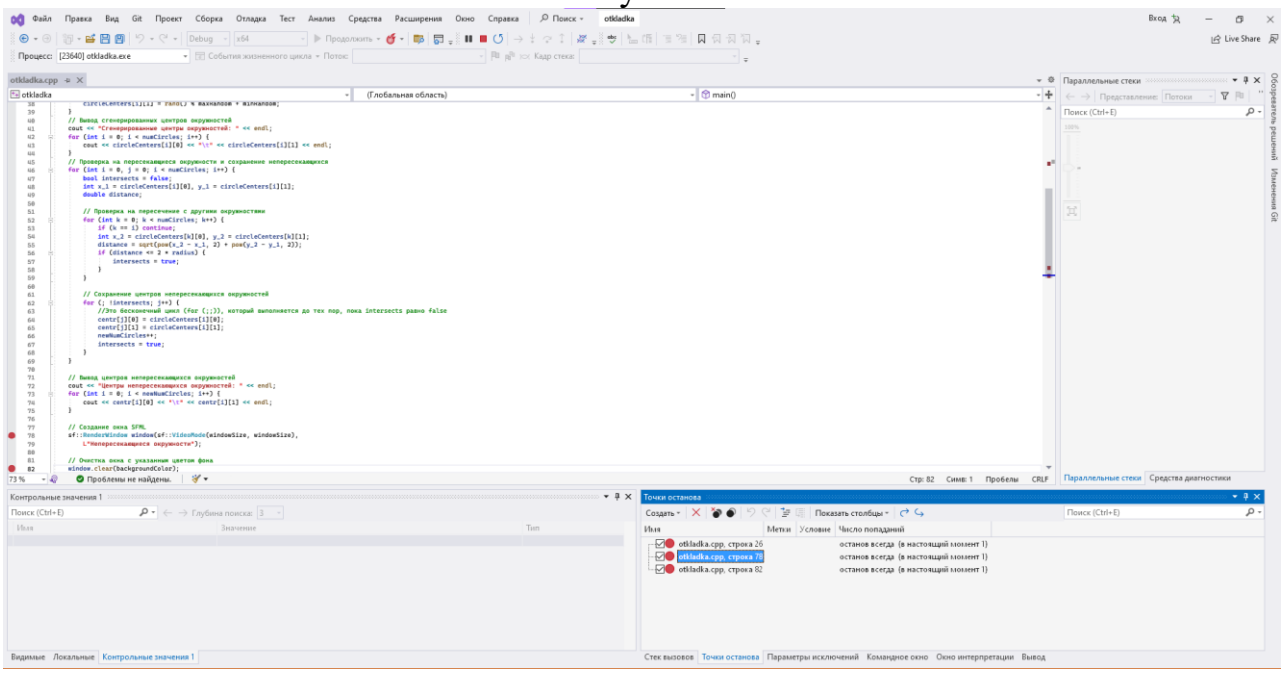

Рисунок 5

- Окно Контрольных значений (рис. 6)

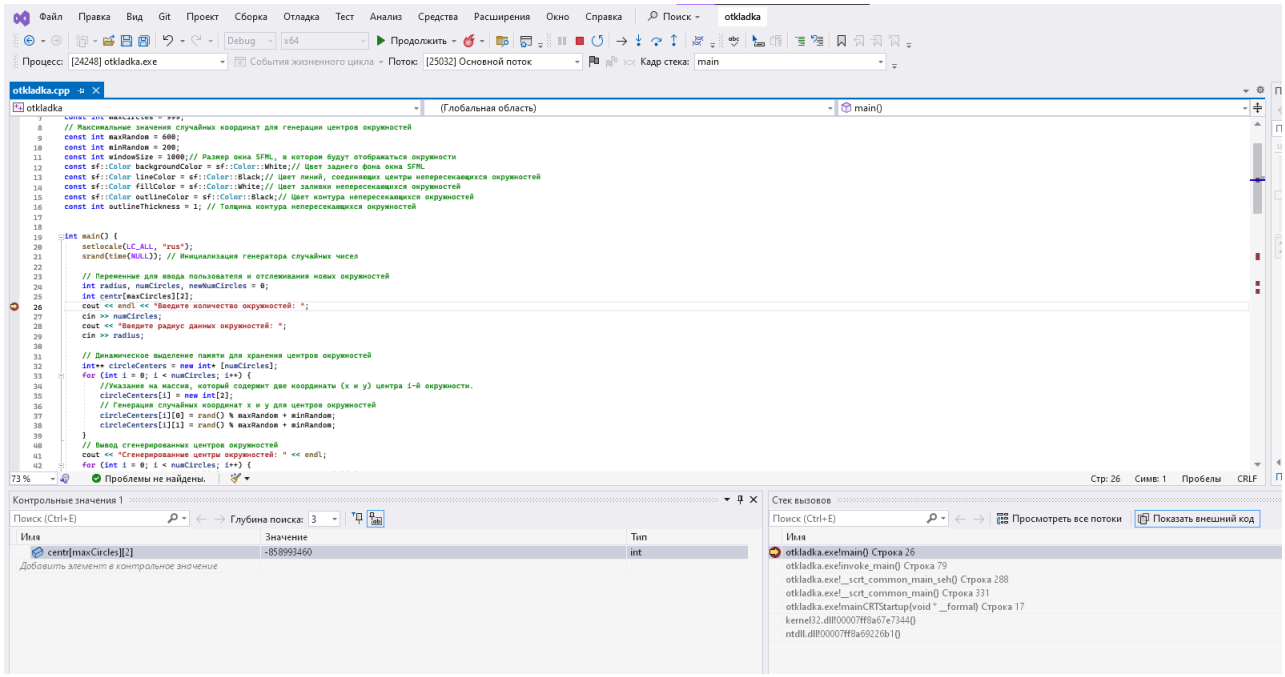
Файл Правка Вид Git Проект Сборка Отладка Тест Анализ Средства Расширения Окно Справка Поиск otkladka
Debug x64 Продолжить
Процесс: [24248] otkladka.exe События жизненного цикла Поток: [25032] Основной поток Кадр стека: main
otkladka.cpp
otkladka (Глобальная область) main()
// Максимальные значения случайных координат для генерации центров окружностей
const int maxRandom = 600;
const int minRandom = 200;
const int windowSize = 1000;// Размер окна SFML, в котором будут отображаться окружности
const sf::Color backgroundColor = sf::Color::White;// Цвет заднего фона окна SFML
const sf::Color lineColor = sf::Color::Black;// Цвет линий, соединяющих центры непересекающихся окружностей
const sf::Color fillColor = sf::Color::White;// Цвет заливки непересекающихся окружностей
const sf::Color outlineColor = sf::Color::Black;// Цвет контура непересекающихся окружностей
const int outlineThickness = 1; // Толщина контура непересекающихся окружностей
int main() {
setlocale(LC_ALL, "rus");
srand(time(NULL)); // Инициализация генератора случайных чисел
// Переменные для ввода пользователя и отслеживания новых окружностей
int radius, numCircles, newNumCircles = 0;
int centr[maxCircles][2];
cout << endl << "Введите количество окружностей: ";
cin >> numCircles;
cout << "Введите радиус данных окружностей: ";
cin >> radius;
// Динамическое выделение памяти для хранения центров окружностей
int** circleCenters = new int* [numCircles];
for (int i = 0; i < numCircles; i++) {
//Указание на массив, который содержит две координаты (x и y) центра i-й окружности.
circleCenters[i] = new int[2];
// Генерация случайных координат x и y для центров окружностей
circleCenters[i][0] = rand() % maxRandom + minRandom;
circleCenters[i][1] = rand() % maxRandom + minRandom;
}
// Вывод сгенерированных центров окружностей
cout << "Сгенерированные центры окружностей: " << endl;
for (int i = 0; i < numCircles; i++) {
73 % Проблемы не найдены. Стр: 26 Симв: 1 Пробелы CRLF
Контрольные значения 1
Поиск (Ctrl+E) Глубина поиска: 3
Имя Значение Тип
centr[maxCircles][2] -858993460 int
Добавить элемент в контрольное значение
Стек вызовов
Поиск (Ctrl+E) Просмотреть все потоки Показать внешний код
Имя
otkladka.exe!main() Строка 26
otkladka.exe!invoke_main() Строка 79
otkladka.exe!__scrt_common_main_seh() Строка 288
otkladka.exe!__scrt_common_main() Строка 331
otkladka.exe!mainCRTStartup(void * __formal) Строка 17
kernel32.dll!00007ff8a67e7344()
ntdll.dll!00007ff8a69226b1()

Рисунок 6

Пути дальнейшего улучшения программы.

1) Добавление более удобного интерфейса