

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №6
дисциплины «Алгоритмизация»
Вариант ____

Выполнил:
Иващенко Олег Андреевич
2 курс, группа ИВТ-б-о-22-1,
09.03.02 «Информационные и
вычислительные машины»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем»

(подпись)

Руководитель практики:
Доцент кафедры инфокоммуникации
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: «Жадные алгоритмы»

Порядок выполнения работы:

Задание 1. Написать программу, принимающая на вход список из точек и выводит количество и точки, нужные для того, чтобы покрыть все входные точки.

Таблица 1 – Код программы

```
using System;
using System.Collections.Generic;
using System.Linq;

class Program
{
    static void Main()
    {
        int N = 10;
        List<Segment> SegmentList = new List<Segment>();
        Random _rnd = new Random();
        Console.WriteLine("[Program] Сгенерированные точки:");
        for (int i = 0; i < N; i++)
        {
            int X = _rnd.Next(0, 9);
            int Y = 0;
            while (Y < X) Y = _rnd.Next(0, 9);
            SegmentList.Add(new Segment(X, Y));
            Console.WriteLine($"[{i}] {SegmentList[i].WriteSegment()}");
        }
        Console.WriteLine("\n[Program] Отсортированный список точек:");
        SegmentList = SegmentList.OrderBy(s => s.X).ThenBy(s => s.Y).ToList(); // Сортировка
        foreach (Segment segment in SegmentList) Console.WriteLine(segment.WriteSegment());

        List<Segment> minSegments = FindMinSegments(SegmentList);

        Console.WriteLine($"[Program] Минимальное количество отрезков: {minSegments.Count}");
        foreach (Segment segment in minSegments) Console.WriteLine(segment.WriteSegment());
        Console.ReadKey();
    }

    static List<Segment> FindMinSegments(List<Segment> segmentList)
    {
        List<Segment> minSegments = new List<Segment>();

        Segment currentSegment = segmentList.First();

        foreach (Segment segment in segmentList.Skip(1))
        {
            if (segment.X > currentSegment.Y)
            {
                // Текущий отрезок не покрывает текущую точку, добавляем новый отрезок
                minSegments.Add(currentSegment);
                currentSegment = segment;
            }
        }
        minSegments.Add(currentSegment);
    }
}
```

```

    }
    else
    {
        // Обновляем текущий отрезок, если текущая точка входит в него
        currentSegment.Y = Math.Max(currentSegment.Y, segment.Y);
    }
}

// Добавляем последний отрезок
minSegments.Add(currentSegment);

return minSegments;
}
}

class Segment
{
    public int X;
    public int Y;

    public Segment(int X, int Y)
    {
        this.X = X;
        this.Y = Y;
    }

    public string WriteSegment() { return $"({X}, {Y})"; }
}

```

```

[Program] Сгенерированные точки:
[0] (6, 8)
[1] (5, 5)
[2] (7, 7)
[3] (8, 8)
[4] (2, 4)
[5] (5, 6)
[6] (8, 8)
[7] (1, 1)
[8] (6, 6)
[9] (0, 0)

[Program] Отсортированный список точек:
(0, 0)
(1, 1)
(2, 4)
(5, 5)
(5, 6)
(6, 6)
(6, 8)
(7, 7)
(8, 8)
(8, 8)

[Program] Минимальное количество отрезков: 4
(0, 0)
(1, 1)
(2, 4)
(5, 8)

```

Рисунок 1 – Результат выполнения программы

Задание 2. Написать программу, принимающую список точек и выводящую максимальное количество не пересекающихся между собой отрезков.

Таблица 2 – Код программы

```
using System;
using System.Collections.Generic;
using System.Linq;

namespace algorithm6._1
{
    internal class Program
    {
        static public List<Segment> OutSegmentList = new List<Segment>();

        static void Main(string[] args)
        {
            int N = 0;
            List<Segment> SegmentList = new List<Segment>();
            Console.WriteLine("[Program] Введите количество точек");
            Console.Write(">>> "); N = int.Parse(Console.ReadLine()); // Ввод количества сегментов
            Random _rnd = new Random();

            for (int i = 0; i < N; i++) // Генерация координат сегментов
            {
                int X = _rnd.Next(0, 10);
                int Y = 0;
                while (Y < X) Y = _rnd.Next(0, 10); // Координата Y не может быть меньше или равной X
                SegmentList.Add(new Segment(X, Y));
            }

            SegmentList = SegmentList.OrderBy(s => s.X).ThenBy(s => s.Y).ToList(); // Сортировка
            сначала по X, затем по Y
            Console.WriteLine("[Program] Отсортированный список точек");
            foreach (Segment segment in SegmentList) Console.WriteLine(segment.WriteSegment()); //
            Вывод отсортированного списка
            OutSegmentList.Add(SegmentList[0]); // Добавление первого сегмента в итоговый список

            for (int i = 1; i < SegmentList.Count - 1; i++)
            {
                /*
                Проверка на то, если у последнего сегмента итогового списка равный X с текущим
                проверяемым сегментом,
                но у второго Y больше. Если условие выполняется, то последний сегмент итогового
                списка заменяется
                на текущий проверяемый сегмент входного списка
                */
                if (OutSegmentList[OutSegmentList.Count - 1].X == SegmentList[i].X &&
                    OutSegmentList[OutSegmentList.Count - 1].Y < SegmentList[i].Y)
                    OutSegmentList[OutSegmentList.Count - 1] = SegmentList[i];

                // Основная проверка
                if (OutSegmentList[OutSegmentList.Count - 1].Y < SegmentList[i].X)
                    OutSegmentList.Add(SegmentList[i]);
            }
        }
    }
}
```

```

    }

    //Вывод итогового списка непересекающихся сегментов
    Console.WriteLine($"[Program] Максимальное количество не пересекающихся точек -
{OutSegmentList.Count}");
    foreach (Segment segment in OutSegmentList) Console.WriteLine(segment.WriteSegment());
    Console.ReadKey();
    }
}

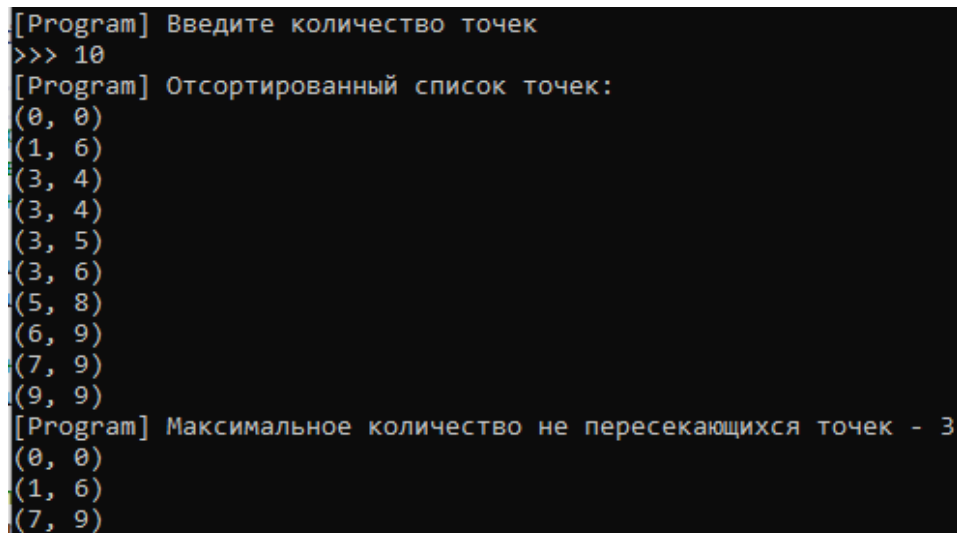
/*
Для удобства работы с сегментами я создал класс,
в котором объект имеет координаты X, Y, а также
функцию для вывода сегмента в виде (X, Y)
*/

class Segment
{
    public int X;
    public int Y;

    public Segment(int X, int Y)
    {
        this.X = X;
        this.Y = Y;
    }

    public string WriteSegment() { return $"({X}, {Y})"; }
}

```



```

[Program] Введите количество точек
>>> 10
[Program] Отсортированный список точек:
(0, 0)
(1, 6)
(3, 4)
(3, 4)
(3, 5)
(3, 6)
(5, 8)
(6, 9)
(7, 9)
(9, 9)
[Program] Максимальное количество не пересекающихся точек - 3
(0, 0)
(1, 6)
(7, 9)

```

Рисунок 2 – Результат работы программы

Вывод: В процессе выполнения практической работы были написаны две программы. Первая программа создаёт двумерный массив, имитирующий отрезки с координатами (X, Y), и выводит минимальное количество отрезков

среди вводных, нужное для покрывания всех других отрезков. Вторая программа создаёт отрезки (класс Segment) в количестве, котором ввёл пользователь, и отбирает среди них максимальное количество отрезков, нужное для покрытия всех вводных отрезков и при этом не пересекающиеся между собой.