

Направление подготовки 09.03.03 «Прикладная информатика»
Профиль «Прикладная информатика в топливно-энергетическом комплексе»
Методическое обеспечение РПД Б1.О.15 «Алгоритмизация и программирование»



**Филиал федерального государственного бюджетного образовательного учреждения
высшего образования
«Национальный исследовательский университет «МЭИ»
в г. Смоленске**

**МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ДЛЯ ОБЕСПЕЧЕНИЯ
ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА**

Направление подготовки: **09.03.03 «Прикладная информатика»**

Профиль: **«Прикладная информатика в топливно-энергетическом комплексе»**

Уровень высшего образования: **бакалавриат**

Нормативный срок обучения: **4 года**

Форма обучения: **очная**

Год набора: **2023**


Смоленск

Методические материалы составили:

канд. техн. наук, доцент кафедры
информационных технологий в экономике и управлении _____ А.Ю. Пучков

«20» _____ января _____ 2023 г.

Заведующий кафедрой информационных технологий в экономике и управлении:


_____ д-р техн. наук, профессор М.И. Дли
подпись _____ ФИО

«08» февраля 2023 г.

МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ

ЛАБОРАТОРНАЯ РАБОТА 1. РАЗРАБОТКА СХЕМ КОМБИНИРОВАННЫХ АЛГОРИТМОВ

Цель работы: изучить на практике разработку схемы комбинированного алгоритма.
Для подготовки к работе изучить:

1. Понятие комбинированного алгоритма
2. Правила построения схемы алгоритма

Теоретические сведения.

Комбинированный алгоритм - это алгоритм, в котором встречаются два вида алгоритма - циклический и разветвленный.

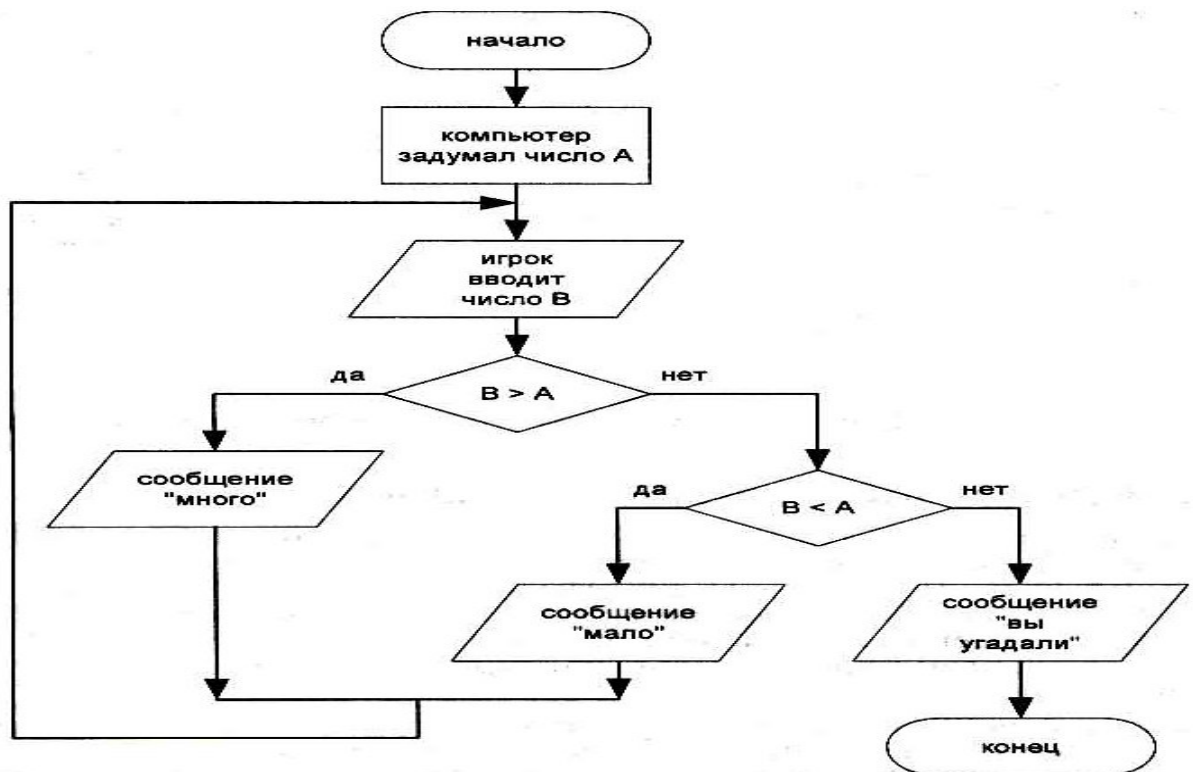


Рисунок 1 – Пример комбинированного алгоритма

Задание: составить схему комбинированного алгоритма и создать программу, основанную на данном алгоритме.

Контрольные вопросы:

1. Что такое комбинированный алгоритм?
2. Какие существуют правила построения схемы алгоритма?
3. Что такое циклический алгоритм?
4. Что такое разветвленный алгоритм?
5. Что такое схема алгоритма?

ЛАБОРАТОРНАЯ РАБОТА 2. ПРОСТАЯ ОБРАБОТКА МНОГОМЕРНЫХ МАССИВОВ

Цель работы: на практике изучить обработку многомерных массивов.

Для подготовки к работе изучить:

1. Понятие многомерного массива
2. Способы ввода многомерного массива (ручной, автоматический)
3. Способы вывода многомерного массива на экран

Теоретические сведения.

Массивы могут иметь несколько измерений. Например, следующее объявление создаст двухмерный массив из четырех строк и двух столбцов.

```
int[,] array = newint[4, 2];
```

Следующее объявление создаст массив из трех измерений: 4, 2 и 3.

```
int[,,] array1 = newint[4, 2, 3];
```

Инициализация массива

Массив можно инициализировать при объявлении, как показано в следующем примере.

```
// Two-dimensional array.
```

```
int[,] array2D = newint[,] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

```
// The same array with dimensions specified.
```

```
int[,] array2Da = newint[4, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

```
// A similar array with string elements.
```

```
string[,] array2Db = newstring[3, 2] { { "one", "two" }, { "three", "four" },  
{ "five", "six" } };
```

```
// Three-dimensional array.
```

```
int[,,] array3D = newint[,,] { { { 1, 2, 3 }, { 4, 5, 6 } },
```

```
{ { 7, 8, 9 }, { 10, 11, 12 } } };
```

```
// The same array with dimensions specified.
```

```
int[,,] array3Da = newint[2, 2, 3] { { { 1, 2, 3 }, { 4, 5, 6 } },
```

```
{ { 7, 8, 9 }, { 10, 11, 12 } } };
```

```
// Accessing array elements.
```

```
System.Console.WriteLine(array2D[0, 0]);
```

```
System.Console.WriteLine(array2D[0, 1]);
```

```
System.Console.WriteLine(array2D[1, 0]);
```

```
System.Console.WriteLine(array2D[1, 1]);
```

```
System.Console.WriteLine(array2D[3, 0]);
```

```
System.Console.WriteLine(array2Db[1, 0]);
```

```
System.Console.WriteLine(array3Da[1, 0, 1]);
```

```
System.Console.WriteLine(array3D[1, 1, 2]);
```

```
// Getting the total count of elements or the length of a given dimension.
```

```
var allLength = array3D.Length;
```

```
var total = 1;
```

```
for (inti = 0; i < array3D.Rank; i++)
```

```
{
```

```
total *= array3D.GetLength(i);
```

```
}
```

```
System.Console.WriteLine("{0} equals {1}", allLength, total);
```

```
// Output:
```

```
// 1
```

```
// 2
```

```
// 3
// 4
// 7
// three
// 8
// 12
// 12 equals 12
```

Можно также инициализировать массив без указания ранга.

```
int[,] array4 = { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

Чтобы объявить переменную массива без инициализации, используйте оператор new для присвоения массива переменной. Использование оператора new показано в следующем примере.

```
int[,] array5;
array5 = newint[,] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } }; // OK
//array5 = {{1,2}, {3,4}, {5,6}, {7,8}}; // Error
```

В следующем примере присваивается значение конкретному элементу массива.

```
array5[2, 1] = 25;
```

Аналогичным образом, в следующем примере получается значение конкретного элемента массива, которое присваивается переменной elementValue.

```
intelementValue = array5[2, 1];
```

В следующем примере кода элементы массива инициализируются с использованием значений по умолчанию (кроме массивов массивов).

```
int[,] array6 = newint[10, 10];
```

Задание представлено в таблице:

№	Задание
1	В массиве $X(n,m)$ изменить значения всех положительных элементов, умножив их значения на 5, а отрицательные элементы уменьшить вдвое.
2	Найти произведения четных и нечетных элементов массива $X(n,m)$.
3	Найти среднее арифметическое не равных нулю элементов массива $X(n,m)$ и подсчитать количество элементов с неположительными значениями.
4	В массиве $Y(n,m)$ найти среднее арифметическое положительных элементов, имеющих нечетные номера.
5	Найти произведения средних арифметических элементов массивов $X(n,m)$ и $Y(n,m)$.
6	В массиве $Y(n,m)$ найти по отдельности суммы и количества элементов, значения которых соответственно больше 5 и меньше -9.
7	Вычислить куб суммы чисел тех элементов массива $X(n,m)$, значения которых меньше заданной величины A или находятся в пределах от B до C (включая указанные границы).

8	В массиве $X(n,m)$ найти среднее геометрическое тех элементов, квадраты которых не превышают заданную величину A .
9	Найти сумму и общее количество тех элементов массива $X(n,m)$, абсолютная величина которых отличается от 5 не более, чем на 1.2.
10	В массиве $X(n,m)$ найти среднее арифметическое тех элементов, значения которых не превышают заданную величину A .
11	В массивах $X(n,m)$, $Y(n,m)$ заменить значение каждого неположительного элемента массива $X(n,m)$ абсолютной величиной соответствующего элемента массива $Y(n,m)$ и подсчитать количество замен.
12	Найти произведение минимальных элементов массивов $X(n,m)$ и $Y(n,m)$.
13	В массиве $X(n,m)$ заменить значения отрицательных элементов их абсолютными величинами, при этом подсчитать число элементов, равных нулю.
14	В массиве $X(n,m)$ найти количество тех элементов, значения которых превышают заданную величину A .
15	Изменить значения всех положительных элементов массива $X(n,m)$ делением каждого из них на номер его строки.
16	Подсчитать по отдельности суммы $C1$ и $C2$ и количества $M1$ и $M2$ отрицательных и положительных элементов заданного массива $X(n,m)$.
17	Найти сумму и количество тех элементов массива $X(n,m)$, которые больше величины P , но меньше другой величины $T (P < T)$.
18	В заданном массиве $X(n,m)$ найти отношение A/B , где A - произведение всех элементов массива, а B - их сумма.
19	В массиве $Y(n,m)$ найти среднее арифметическое нечетных элементов, имеющих четные номера.
20	Подсчитать количество отрицательных значений четных элементов и количество положительных значений нечетных элементов массива $X(n,m)$.
21	В массиве действительных чисел $X(n,m)$ определить количество целых чисел (не имеющих дробной части).
22	В массиве $X(n,m)$ найти среднее арифметическое тех элементов, которые находятся в строках с четными номерами.

23	В массиве $X(n,m)$ заменить значения отрицательных элементов их абсолютными величинами, при этом подсчитать число элементов, равных нулю.
24	Найти сумму и общее количество тех элементов массива $X(n,m)$, абсолютная величина которых отличается от 7 не более, чем на 1.
25	В массиве $Y(n,m)$ найти по отдельности суммы и количества элементов, стоящих на четных и нечетных позициях.
26	В массиве $Y(n,m)$ найти среднее арифметическое четных элементов, имеющих четные номера.
27	В заданном массиве $X(n,m)$ найти отношение A/B , где A – сумма элементов нечетных строк массива, а B – их сумма.
28	Найти произведение минимальных элементов массивов $X(n,m)$ и $Y(n,m)$.
29	В массивах $X(n,m)$, $Y(n,m)$ заменить значение каждого элемента массива $X(n,m)$ соответствующим элементом массива $Y(n,m)$, если $X[i,j] > Y[i,j]$, и подсчитать количество замен.
30	В массиве действительных чисел $X(n,m)$ определить количество чисел, которые являются полным квадратом.

Контрольные вопросы:

1. Что такое двумерный массив?
2. В чем заключается ручной способ ввода двумерного массива?
3. В чем заключается автоматический способ ввода двумерного массива?
4. Какие есть способы вывода двумерного массива на экран?
5. Как можно эффективно изменить размерность массива?

ЛАБОРАТОРНАЯ РАБОТА 3. СЛОЖНАЯ ОБРАБОТКА МНОГОМЕРНЫХ МАССИВОВ

Цель работы: на практике изучить сложную обработку многомерных массивов.

Для подготовки к работе изучить:

1. Понятие многомерного массива
2. Понятие сложной обработки массивов

Теоретические сведения.

Массивы могут иметь несколько измерений. Например, следующее объявление создает двухмерный массив из четырех строк и двух столбцов.

```
int[,] array = newint[4, 2];
```

Следующее объявление создает массив из трех измерений: 4, 2 и 3.

```
int[,,] array1 = newint[4, 2, 3];
```

Инициализация массива

Массив можно инициализировать при объявлении, как показано в следующем примере.

```
// Two-dimensional array.
```

```
int[,] array2D = newint[,] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

```
// The same array with dimensions specified.
```

```
int[,] array2Da = newint[4, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

```
// A similar array with string elements.
```

```
string[,] array2Db = newstring[3, 2] { { "one", "two" }, { "three", "four" },  
{ "five", "six" } };
```

```
// Three-dimensional array.
```

```
int[,,] array3D = newint[,,] { { { 1, 2, 3 }, { 4, 5, 6 } },  
{ { 7, 8, 9 }, { 10, 11, 12 } } };
```

```
// The same array with dimensions specified.
```

```
int[,,] array3Da = newint[2, 2, 3] { { { 1, 2, 3 }, { 4, 5, 6 } },  
{ { 7, 8, 9 }, { 10, 11, 12 } } };
```

```
// Accessing array elements.
```

```
System.Console.WriteLine(array2D[0, 0]);  
System.Console.WriteLine(array2D[0, 1]);  
System.Console.WriteLine(array2D[1, 0]);  
System.Console.WriteLine(array2D[1, 1]);  
System.Console.WriteLine(array2D[3, 0]);  
System.Console.WriteLine(array2Db[1, 0]);  
System.Console.WriteLine(array3Da[1, 0, 1]);  
System.Console.WriteLine(array3D[1, 1, 2]);
```

```
// Getting the total count of elements or the length of a given dimension.
```

```
var allLength = array3D.Length;  
var total = 1;  
for (inti = 0; i < array3D.Rank; i++)  
{  
total *= array3D.GetLength(i);  
}  
System.Console.WriteLine("{0} equals {1}", allLength, total);
```

```
// Output:
```

```
// 1  
// 2  
// 3  
// 4
```

```
// 7
// three
// 8
// 12
// 12 equals 12
```

Можно также инициализировать массив без указания ранга.

```
int[,] array4 = { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

Чтобы объявить переменную массива без инициализации, используйте оператор new для присвоения массива переменной. Использование оператора new показано в следующем примере.

```
int[,] array5;
array5 = newint[,] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } }; // OK
//array5 = {{1,2}, {3,4}, {5,6}, {7,8}}; // Error
```

В следующем примере присваивается значение конкретному элементу массива.

```
array5[2, 1] = 25;
```

Аналогичным образом, в следующем примере получается значение конкретного элемента массива, которое присваивается переменной elementValue.

```
intelementValue = array5[2, 1];
```

В следующем примере кода элементы массива инициализируются с использованием значений по умолчанию (кроме массивов массивов).

```
int[,] array6 = newint[10, 10];
```

Задание представлено в таблице:

№	Задание
1	Если максимальный элемент массива $A(n,m)$ положителен, то найти количество нулевых элементов каждого столбца, иначе – удвоить все отрицательные элементы всего массива.
2	Если количество положительных элементов массива $A(n,m)$ больше количества отрицательных элементов, то заменить все отрицательные элементы на -1, иначе – найти количество ненулевых элементов каждого столбца.
3	Если сумма элементов массива $A(n,m)$ больше 100, то найти количество отрицательных элементов каждой строки, иначе – увеличить все четные элементы массива на 1.
4	Если минимальный элемент массива $A(n, m)$ четный, то удвоить все отрицательные элементы, иначе – найти сумму положительных элементов каждой строки.
5	Если сумма элементов главной диагонали массива $A(n,n)$ больше суммы элементов побочной диагонали, то найти количество четных элементов каждого столбца, иначе – заменить все элементы диагоналей массива на 1.

6	Если количество четных элементов массива $A(n, m)$ больше количества нечетных элементов, то уменьшить все положительные элементы вдвое, иначе – найти сумму отрицательных элементов каждого столбца.
7	Если количество нечетных элементов массива $A(n, m)$ больше 5, то найти среднее арифметическое отрицательных элементов каждой строки, иначе – удвоить все элементы массива.
8	Если среднее арифметическое элементов массива $A(n, m)$ больше 100, но меньше 50, то увеличить все положительные элементы на 5, иначе – найти количество нечетных элементов каждой строки.
9	Если сумма элементов побочной диагонали массива $A(n, n)$ больше положительна, то найти количество нечетных элементов каждого столбца, иначе – найти среднее арифметическое отрицательных элементов всего массива.
10	Если разность максимального и минимального элементов массива $A(n, m)$ больше 20, то уменьшить все положительные элементы вдвое, иначе – найти произведение отрицательных элементов каждого столбца.
11	Если количество нечетных элементов массива $A(n, m)$ больше количества четных элементов, то найти сумму положительных элементов каждой строки, иначе – уменьшить все элементы массива вдва раза.
12	Если среднее арифметическое положительных элементов массива $A(n, m)$ больше 100, то удвоить все четные элементы, иначе - найти количество нечетных элементов каждой строки.
13	Дана матрица $A(n, m)$. Вывести номера тех столбцов, сумма элементов которых меньше нуля, и число таких столбцов.
14	Найти общую сумму элементов тех столбцов матрицы $A(n, m)$, сумма элементов в каждом из которых положительна.
15	Дана матрица $A(n, m)$. Отпечатать номера только тех строк, сумма элементов которых превышает заданную величину T , и число таких строк.
16	Дана матрица $A(n, m)$. Отпечатать номера тех строк, элементы которых имеют совпадающие значения, и число таких строк.
17	Дана матрица $A(n, m)$. Отпечатать номера тех столбцов, в которых не менее 2 элементов имеют нулевое значение, и число таких столбцов.

18	Дана матрица $A(n, m)$. В каждом столбце удвоить те элементы, которые следуют за минимальным элементом этого столбца.
19	Дана матрица $A(n, m)$. Напечатать номер каждой строки, в которой второй элемент меньше среднего арифметического элементов этой строки, и число таких строк.
20	Среди столбцов целочисленной матрицы $A(n, m)$ найти столбец с минимальным произведением элементов.
21	Дан массив $A(n, m)$. В каждой строке находится минимальный элемент, затем среди этих чисел выбирается максимальное. Напечатать номер строки массива A , в которой расположено выбранное число.
22	Дана целочисленная матрица $A(n, m)$. Найти номера столбцов, элементы каждого из которых образуют возрастающую последовательность.
23	Дана матрица $A(n, m)$. Отпечатать номера тех строк, которые имеют нулевые элементы, и число таких строк.
24	Дана матрица $A(n, m)$. Найти номера строк, элементы в каждой из которых одинаковы.
25	Дана матрица $A(n, m)$. Найти номера строк, элементы каждой из которых образуют монотонную последовательность (монотонно убывающую или монотонно возрастающую).
26	Дана матрица $A(n, m)$. Найти номера строк, элементы которых образуют симметричные последовательности (палиндромы).
27	Дана матрица $A(n, m)$. В каждой строке удвоить те элементы, которые следуют за минимальным элементом этой строки.
28	Найти общую сумму элементов тех строк матрицы $A(n, m)$, сумма элементов в каждой из которых положительна.
29	Дана целочисленная матрица $A(n, m)$. Найти номера строк, элементы каждой из которых образуют возрастающую последовательность.
30	Если сумма элементов массива $A(n, m)$ больше 20, то найти количество четных элементов каждого столбца, иначе — увеличить все отрицательные элементы массива на 1.

Контрольные вопросы:

1. Что такое двумерный массив?
2. В чем заключается ручной способ ввода двумерного массива?
3. В чем заключается автоматический способ ввода двумерного массива?
4. Какие есть способы вывода двумерного массива на экран?
5. Как можно эффективно изменить размерность массива?

ЛАБОРАТОРНАЯ РАБОТА 4. РАЗРАБОТКА АЛГОРИТМОВ СОРТИРОВКИ И СРАВНЕНИЕ ИХ ЭФФЕКТИВНОСТИ

Цель работы: на практике разработать различные алгоритмы сортировки и сравнить их эффективность

Для подготовки к работе изучить:

1. Метод случайной сортировки
2. Метод сортировки пузырьком
3. Метод сортировки по частям
4. Метод сортировки вставками

Теоретические сведения.

Реализация случайно сортировки:

```
//метод для проверки упорядоченности массива
static bool IsSorted(int[] a)
{
    for (int i = 0; i < a.Length - 1; i++)
    {
        if (a[i] > a[i + 1])
            return false;
    }

    return true;
}

//перемешивание элементов массива
static int[] RandomPermutation(int[] a)
{
    Random random = new Random();
    var n = a.Length;
    while (n > 1)
    {
        n--;
        var i = random.Next(n + 1);
        var temp = a[i];
        a[i] = a[n];
        a[n] = temp;
    }

    return a;
}

//случайная сортировка
static int[] BogoSort(int[] a)
{
    while(!IsSorted(a))
    {
        a = RandomPermutation(a);
    }

    return a;
}

static void Main(string[] args)
{
    Console.WriteLine("Случайная сортировка");
    Console.Write("Введите элементы массива: ");
    var parts = Console.ReadLine().Split(new[] { " ", ",", ";" }, StringSplitOptions.RemoveEmptyEntries);
    var array = new int[parts.Length];
    for (int i = 0; i < parts.Length;i++)
    {
        array[i] = Convert.ToInt32(parts[i]);
    }

    Console.WriteLine("Отсортированный массив: {0}", string.Join(", ", BogoSort(array)));

    Console.ReadLine();
}
```

Реализация сортировки пузырьком:

```
using System;

class Program
{
    //метод обмена элементов
    static void Swap(ref int e1, ref int e2)
    {
        var temp = e1;
        e1 = e2;
        e2 = temp;
    }

    //сортировка пузырьком
    static int[] BubbleSort(int[] array)
    {
        var len = array.Length;
        for (var i = 1; i < len; i++)
        {
            for (var j = 0; j < len - i; j++)
            {
                if (array[j] > array[j + 1])
                {
                    Swap(ref array[j], ref array[j + 1]);
                }
            }
        }

        return array;
    }

    static void Main(string[] args)
    {
        Console.WriteLine("Сортировка пузырьком");
        Console.Write("Введите элементы массива: ");
        var parts = Console.ReadLine().Split(new[] { " ", ",", ";" }, StringSplitOptions.RemoveEmptyEntries);
        var array = new int[parts.Length];
        for (int i = 0; i < parts.Length; i++)
        {
            array[i] = Convert.ToInt32(parts[i]);
        }

        Console.WriteLine("Отсортированный массив: {0}", string.Join(" ", BubbleSort(array)));

        Console.ReadLine();
    }
}
```

Метод сортировки по частям:

```
using System;

class Program
{
    //метод обмена элементов
    static void Swap(ref int a, ref int b)
    {
        var t = a;
        a = b;
        b = t;
    }

    //сортировка по частям
    static int[] StoogeSort(int[] array, int startIndex, int endIndex)
    {
        if (array[startIndex] > array[endIndex])
        {
            Swap(ref array[startIndex], ref array[endIndex]);
        }

        if (endIndex - startIndex > 1)
        {
            var len = (endIndex - startIndex + 1) / 3;
            StoogeSort(array, startIndex, endIndex - len);
            StoogeSort(array, startIndex + len, endIndex);
            StoogeSort(array, startIndex, endIndex - len);
        }

        return array;
    }

    static int[] StoogeSort(int[] array)
    {
        return StoogeSort(array, 0, array.Length - 1);
    }

    static void Main(string[] args)
    {
        Console.WriteLine("Сортировка по частям");
        Console.Write("Введите элементы массива: ");
        var parts = Console.ReadLine().Split(new[] { " ", ",", ";" }, StringSplitOptions.RemoveEmptyEntries);
        var array = new int[parts.Length];
        for (int i = 0; i < parts.Length; i++)
        {
            array[i] = Convert.ToInt32(parts[i]);
        }

        Console.WriteLine("Упорядоченный массив: {0}", string.Join(" ", StoogeSort(array)));

        Console.ReadLine();
    }
}
```

Метод сортировки вставками:

```
using System;

class Program
{
    //метод обмена элементов
    static void Swap(ref int e1, ref int e2)
    {
        var temp = e1;
        e1 = e2;
        e2 = temp;
    }

    //сортировка вставками
    static int[] InsertionSort(int[] array)
    {
        for (var i = 1; i < array.Length; i++)
        {
            var key = array[i];
            var j = i;
            while ((j > 1) && (array[j - 1] > key))
            {
                Swap(ref array[j - 1], ref array[j]);
                j--;
            }

            array[j] = key;
        }

        return array;
    }

    static void Main(string[] args)
    {
        Console.WriteLine("Сортировка вставками");
        Console.Write("Введите элементы массива: ");
        var parts = Console.ReadLine().Split(new[] { " ", ",", ";" }, StringSplitOptions.RemoveEmptyEntries);
        var array = new int[parts.Length];
        for (int i = 0; i < parts.Length; i++)
        {
            array[i] = Convert.ToInt32(parts[i]);
        }

        Console.WriteLine("Упорядоченный массив: {0}", string.Join(", ", InsertionSort(array)));

        Console.ReadLine();
    }
}
```

Задание: на практике реализовать всех приведенные методы сортировки, после чего сделать выводы об их эффективности.

Контрольные вопросы:

1. В чем заключается метод случайной сортировки?
2. В чем заключается метод пузырьковой сортировки?
3. В чем заключается метод сортировки по частям?
4. В чем заключается метод сортировки вставками?
5. Какой из перечисленных методов наиболее эффективен?

ЛАБОРАТОРНАЯ РАБОТА 5. РЕКУРСИВНЫЕ АЛГОРИТМЫ СОРТИРОВКИ ДАННЫХ

Цель работы: на практике изучить рекурсивные алгоритмы сортировки данных.

Для подготовки к работе изучить:

1. Изучить метод быстрой сортировки
2. Изучить метод сортировки слиянием

Теоретические сведения.

Метод быстрой сортировки:

```
using System;

class Program
{
    //метод для обмена элементов массива
    static void Swap(ref int x, ref int y)
    {
        var t = x;
        x = y;
        y = t;
    }

    //метод возвращающий индекс опорного элемента
    static int Partition(int[] array, int minIndex, int maxIndex)
    {
        var pivot = minIndex - 1;
        for (var i = minIndex; i < maxIndex; i++)
        {
            if (array[i] < array[maxIndex])
            {
                pivot++;
                Swap(ref array[pivot], ref array[i]);
            }
        }

        pivot++;
        Swap(ref array[pivot], ref array[maxIndex]);
        return pivot;
    }

    //быстрая сортировка
    static int[] QuickSort(int[] array, int minIndex, int maxIndex)
    {
        if (minIndex >= maxIndex)
        {
            return array;
        }

        var pivotIndex = Partition(array, minIndex, maxIndex);
        QuickSort(array, minIndex, pivotIndex - 1);
        QuickSort(array, pivotIndex + 1, maxIndex);

        return array;
    }

    static int[] QuickSort(int[] array)
    {
        return QuickSort(array, 0, array.Length - 1);
    }

    static void Main(string[] args)
    {
        Console.WriteLine("N = ");
        var len = Convert.ToInt32(Console.ReadLine());
```

```
var a = new int[len];  
for (var i = 0; i < a.Length; ++i)  
{  
    Console.WriteLine("a[{0}] = ", i);  
    a[i] = Convert.ToInt32(Console.ReadLine());  
}  
  
Console.WriteLine("Упорядоченный массив: {0}", string.Join(", ", QuickSort(a)));  
Console.ReadLine();  
}  
}
```

Метод сортировки слиянием:

```
using System;  
  
class Program  
{  
    //метод для слияния массивов  
    static void Merge(int[] array, int lowIndex, int middleIndex, int highIndex)  
    {  
        var left = lowIndex;  
        var right = middleIndex + 1;  
        var tempArray = new int[highIndex - lowIndex + 1];  
        var index = 0;  
  
        while ((left <= middleIndex) && (right <= highIndex))  
        {  
            if (array[left] < array[right])  
            {  
                tempArray[index] = array[left];  
                left++;  
            }  
            else  
            {  
                tempArray[index] = array[right];  
                right++;  
            }  
  
            index++;  
        }  
  
        for (var i = left; i <= middleIndex; i++)  
        {  
            tempArray[index] = array[i];  
            index++;  
        }  
  
        for (var i = right; i <= highIndex; i++)  
        {  
            tempArray[index] = array[i];  
            index++;  
        }  
  
        for (var i = 0; i < tempArray.Length; i++)  
        {  
            array[lowIndex + i] = tempArray[i];  
        }  
    }  
  
    //сортировка слиянием  
    static int[] MergeSort(int[] array, int lowIndex, int highIndex)  
    {  
        if (lowIndex < highIndex)  
        {
```

```
        var middleIndex = (lowIndex + highIndex) / 2;  
        MergeSort(array, lowIndex, middleIndex);  
        MergeSort(array, middleIndex + 1, highIndex);  
        Merge(array, lowIndex, middleIndex, highIndex);  
    }  
  
    return array;  
}  
  
static int[] MergeSort(int[] array)  
{  
    return MergeSort(array, 0, array.Length - 1);  
}  
  
static void Main(string[] args)  
{  
    Console.WriteLine("Сортировка слиянием");  
    Console.Write("Введите элементы массива: ");  
    var s = Console.ReadLine().Split(new[] { " ", ",", ";" }, StringSplitOptions.RemoveEmptyEntries);  
    var array = new int[s.Length];  
    for (int i = 0; i < s.Length; i++)  
    {  
        array[i] = Convert.ToInt32(s[i]);  
    }  
  
    Console.WriteLine("Упорядоченный массив: {0}", string.Join(" ", MergeSort(array)));  
    Console.ReadLine();  
}
```

Задание: реализовать два метода сортировки, описанных выше

Контрольные вопросы:

1. Что такое рекурсия?
2. В чем заключается метод сортировки слиянием?
3. В чем заключается метод быстрой сортировки?
4. Обоснуйте эффективность каждого из методов
5. Какой из этих двух методов более эффективен?

ЛАБОРАТОРНАЯ РАБОТА 6. РАЗРАБОТКА ПРОГРАММ С ИСПОЛЬЗОВАНИЕМ УКАЗАТЕЛЕЙ

Цель работы: на практике изучить разработку программы с использованием указателей.

Для подготовки к работе изучить:

1. Понятие указателя
2. Способы применения указателей

Теоретические сведения.

В языке C# указатели очень редко используются, однако в некоторых случаях можно прибегать к ним для оптимизации приложений. Код, применяющий указатели, еще называют небезопасным кодом. Однако это не значит, что он представляет какую-то опасность. Просто при работе с ним все действия по использованию памяти, в том числе по ее очистке, ложится целиком на нас, а не на среду CLR. И с точки зрения CLR такой код не безопасен, так как среда не может проверить данный код, поэтому повышается вероятность различного рода ошибок.

Чтобы использовать небезопасный код в C#, надо первым делом указать проекту, что он будет работать с небезопасным кодом. Для этого надо установить в настройках проекта соответствующий флаг - в меню Project (Проект) найти Свойства проекта. Затем в меню Build установить флажок **Allowunsafe code (Разрешить небезопасный код)**

Теперь мы можем приступать к работе с небезопасным кодом и указателями.

Ключевое слово unsafe

Блок кода или метод, в котором используются указатели, помечается ключевым словом **unsafe**:

```
static void Main(string[] args)
{
    // блок кода, использующий указатели
    unsafe
    {

    }
}
```

Метод, использующий указатели:

```
unsafe private static void PointerMethod()
{

}
```

Также с помощью unsafe можно объявлять структуры:

```
unsafe struct State
{

}
```

Операции * и &

Ключевой при работе с указателями является операция *, которую еще называют операцией разыменовывания. Операция разыменовывания позволяет получить или установить значение по адресу, на который указывает указатель. Для получения адреса переменной применяется операция &:

```
static void Main(string[] args)
{
    unsafe
    {
        int* x; // определение указателя
0         int y = 10; // определяем переменную

1         x = &y; // указатель x теперь указывает на адрес переменной y
        Console.WriteLine(*x); // 10

2         y = y + 20;
        Console.WriteLine(*x); // 30

3         *x = 50;
4         Console.WriteLine(y); // переменная y=50
    }
5     Console.ReadLine();
}

6
7
8
```

При объявлении указателя указываем тип `int* x`; - в данном случае объявляется указатель на целое число. Но кроме типа `int` можно использовать и другие: `sbyte`, `byte`, `short`, `ushort`, `int`, `uint`, `long`, `ulong`, `char`, `float`, `double`, `decimal` или `bool`. Также можно объявлять указатели на типы `enum`, структуры и другие указатели.

Выражение `x = &y`; позволяет нам получить адрес переменной `y` и установить на него указатель `x`. До этого указатель `x` не на что не указывал.

После этого все операции с `y` будут влиять на значение, получаемое через указатель `x` и наоборот, так как они указывают на одну и ту же область в памяти.

Для получения значения, которое хранится в области памяти, на которую указывает указатель `x`, используется выражение `*x`.

Получение адреса

Используя преобразование указателя к целочисленному типу, можно получить адрес памяти, на который указывает указатель:

```
int* x; // определение указателя
int y = 10; // определяем переменную

x = &y; // указатель x теперь указывает на адрес переменной y

// получим адрес переменной y
uint addr = (uint)x;
Console.WriteLine("Адрес переменной y: {0}", addr);
```

Так как значение адреса - это целое число, а на 32-разрядных системах диапазон адресов 0 до 4 000 000 000, то для получения адреса используется преобразование в тип `uint`,

long или ulong. Соответственно на 64-разрядных системах диапазон доступных адресов гораздо больше, поэтому в данном случае лучше использовать ulong, чтобы избежать ошибки переполнения.

Указатель на другой указатель

Объявление и использование указателя на указатель:

```
static void Main(string[] args)
{
    unsafe
    {
        int* x; // определение указателя
        int y = 10; // определяем переменную

0         x = &y; // указатель x теперь указывает на адрес переменной y
1         int** z = &x; // указатель z теперь указывает на адрес, который указывает на y
2         **z = **z + 40; // изменение указателя z повлечет изменение переменной y
3         Console.WriteLine(y); // переменная y=50
4         Console.WriteLine(**z); // переменная **z=50
5     }
    Console.ReadLine();
}
```

Задание: реализовать две программы, приведенные ниже.

Программа 1:

```
using System;

namespace ConsoleApplication0
{
    class Program
    {
        unsafe static void Main (string[] args) {
            Console.WriteLine("Enter array size:");
            int arrSz = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter int A:");
            int A = Convert.ToInt32(Console.ReadLine());

            int val = 0;
            int i = 0, j = 0;

            int* arrSource = stackalloc int[arrSz];

            Console.WriteLine("Enter source array values:");
            while (i < arrSz && int.TryParse(Console.ReadLine(),
                out val)) {
                if (val > A) j++;
            }
        }
    }
}
```

```
        arrSource[i++] = val;
    }

    int* arrDest = stackalloc int[j];
    j = 0;

    Console.WriteLine("\r\nSource array:");
    for (i = 0; i < arrSz; i++) {
        if (arrSource[i] > A)
            arrDest[j++] = arrSource[i];
        Console.Write(arrSource[i] + " ");
    }

    Console.WriteLine("\r\n\r\nResult array:");
    for (i = 0; i < j; i++)
        Console.Write(arrDest[i] + " ");

    Console.ReadLine();
}
}
```

Программа 2:

```
using System;
using System.Runtime.InteropServices;

namespace ConsoleApplication0
{
    class Program
    {
        unsafe static void Main (string[] args) {
            Console.WriteLine("Enter array size:");
            int arrSz = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter int A:");
            int A = Convert.ToInt32(Console.ReadLine());

            int val = 0;
            int i = 0, j = 0;

            int* arrSource = (int*) Marshal.AllocHGlobal (arrSz * sizeof (int));

            Console.WriteLine("Enter source array values:");
            while (i < arrSz && int.TryParse (Console.ReadLine (),
            out val)) {
                if (val > A) j++;
                arrSource[i++] = val;
            }

            int* arrDest = (int*) Marshal.AllocHGlobal (j
            * sizeof (int));
```

```
        j =0;

        Console.WriteLine("\r\nSource array:");
        for (i=0;i<arrSz;i++){
            if (arrSource[i]> A)
                arrDest[j++]=arrSource[i];
            Console.Write(arrSource[i]+" ");
        }

        Console.WriteLine("\r\n\r\nResult array:");
        for (i=0;i< j;i++)
            Console.Write(arrDest[i]+" ");

        Console.ReadLine();

        Marshal.FreeHGlobal ((IntPtr) arrSource);
        Marshal.FreeHGlobal ((IntPtr) arrDest);
    }
}
```

Контрольные вопросы:

1. Что такое указатели?
2. Как можно применять указатели?
3. Для чего используются указатели?
4. Как применить указатель на другой указатель?
5. Какая операция является ключевой при работе с указателями?

ЛАБОРАТОРНАЯ РАБОТА 7. ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ СТЕКА И ОЧЕРЕДИ (ИНФИКСНЫЕ, ПОСТФИКСНЫЕ И ПРЕФИКСНЫЕ ФОРМЫ ЗАПИСИ ВЫРАЖЕНИЙ)

Цель работы: на практике изучить применение стека и очереди.

Для подготовки к работе изучить:

1. Понятие стека;
2. Понятие очереди;
3. Инфиксные, постфиксные и префиксные формы записи выражений.

Теоретические сведения.

Очередь (queue) -это структура данных, которая работает по принципу FIFO (FirstInFirstOut - Первый пришел, первый вышел). При добавлении новый элемент помещается в конец очереди или ее хвост, а удаление идет с начала очереди или головы очереди.

Банальным примером очереди может служить обычная очередь в больнице, где новые пациенты встают в конец очереди, а прием пациентов осуществляется с начала очереди.

Для реализации очереди мы опять же будем использовать класс элемента, который содержит сами данные и ссылку на следующий элемент:

```
1
2 publicclassNode<T>
3 {
4     publicNode(T data)
5     {
6         Data = data;
7     }
8     publicT Data { get; set; }
9     publicNode<T> Next { get; set; }
10 }
```

Каждый элемент в стеке, как и в односвязном списке, будет представлен классом

Node:

```
1
2 publicclassNode<T>
3 {
4     publicNode(T data)
5     {
6         Data = data;
7     }
8     publicT Data { get; set; }
9     publicNode<T> Next { get; set; }
10 }
```

На основе класса Node мы можем реализовать структуру стек:

```
1 usingSystem;
2 usingSystem.Collections.Generic;
3 usingSystem.Collections;
4 namespaceSimpleAlgorithmsApp
5 {
6     publicclassNodeStack<T> :IEnumerable<T>
7     {
8         Node<T> head;
9         intcount;
```

```
10     public bool IsEmpty
11     {
12         get { return count == 0; }
13     }
14     public int Count
15     {
16         get { return count; }
17     }
18     public void Push(T item)
19     {
20         // увеличиваем стек
21         Node<T> node = newNode<T>(item);
22         node.Next = head; // переустанавливаем верхушку стека на новый элемент
23         head = node;
24         count++;
25     }
26     public T Pop()
27     {
28         // если стек пуст, выбрасываем исключение
29         if (IsEmpty)
30             throw new InvalidOperationException("Стек пуст");
31         Node<T> temp = head;
32         head = head.Next; // переустанавливаем верхушку стека на следующий элемент
33         count--;
34         return temp.Data;
35     }
36     public T Peek()
37     {
38         if (IsEmpty)
39             throw new InvalidOperationException("Стек пуст");
40         return head.Data;
41     }
42     IEnumerator IEnumerable.GetEnumerator()
43     {
44         return ((IEnumerable) this).GetEnumerator();
45     }
46     IEnumerator<T> IEnumerable<T>.GetEnumerator()
47     {
48         Node<T> current = head;
49         while (current != null)
50         {
51             yield return current.Data;
52             current = current.Next;
53         }
54     }
55 }
56
57
58
59
```

60

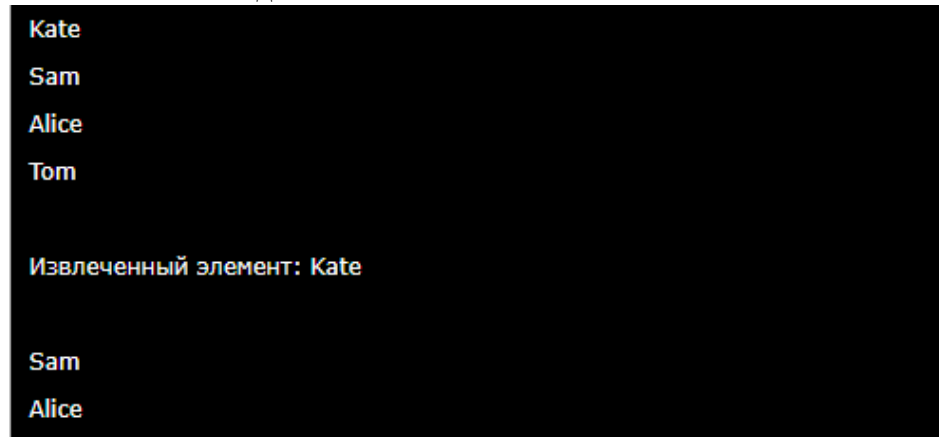
61

Задание: на практике реализовать две программы, приведенные ниже

Программа 1 - применение очереди:

```
Queue<string> queue = new Queue<string>();  
queue.Enqueue("Kate");  
queue.Enqueue("Sam");  
queue.Enqueue("Alice");  
queue.Enqueue("Tom");  
  
foreach(string item in queue)  
    Console.WriteLine(item);  
Console.WriteLine();  
  
Console.WriteLine();  
string firstItem = queue.Dequeue();  
Console.WriteLine($"Извлеченный элемент: {firstItem}");  
Console.WriteLine();  
  
foreach(string item in queue)  
    Console.WriteLine(item);
```

Консольный вывод:

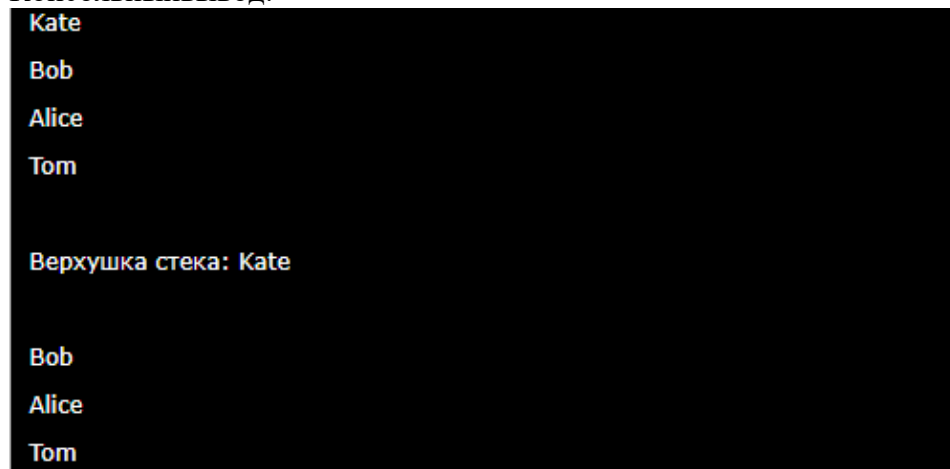


```
Kate  
Sam  
Alice  
Tom  
  
Извлеченный элемент: Kate  
  
Sam  
Alice
```

Программа 2 - применение стека:

```
NodeStack<string> stack = newNodeStack<string>();  
stack.Push("Tom");  
stack.Push("Alice");  
stack.Push("Bob");  
stack.Push("Kate");  
  
foreach(var item instack)  
{  
    Console.WriteLine(item);  
}  
Console.WriteLine();  
stringheader = stack.Peek();  
Console.WriteLine($"Верхушкастека: {header}");  
Console.WriteLine();  
  
header = stack.Pop();  
foreach(var item instack)  
{  
    Console.WriteLine(item);  
}
```

Консольный вывод:



```
Kate  
Bob  
Alice  
Tom  
  
Верхушка стека: Kate  
  
Bob  
Alice  
Tom
```

Контрольные вопросы:

1. Что такое очередь?
2. Что такое стек?
3. Для чего применяется очередь?
4. Для чего применяется стек?
5. Какие бывают формы записи выражений?

ЛАБОРАТОРНАЯ РАБОТА 8. РАЗРАБОТКА ПРОГРАММЫ ДЛЯ АЛГОРИТМА ДЕЙКСТРЫ

Цель работы: на практике разработать программу для алгоритма Дейкстры.

Для подготовки к работе изучить:

1. Алгоритм Дейкстры;
2. Способ реализации алгоритма Дейкстры.

Теоретические сведения.

Алгоритм Дейкстры – алгоритм для поиска кратчайшего пути между двумя заданными вершинами графа.

Алгоритм можно использовать для графов с положительными значениями весов ребер.

Описание алгоритма Дейкстры

Для каждой вершины графа вычисляется сумма весов ребер(расстояние) от начальной вершины.

Инициализация

Сумма весов для стартовой вершины устанавливается в ноль, а для всех остальных – бесконечность. Это показывает что расстояние до других вершин неизвестны. Все кроме первой вершины помечаются как непосещенные.

Итерации алгоритма

Алгоритм завершается как только все вершины посещены. В ином случае, из непосещенных вершин, выбирается та, у которой меньшая сумма весов. Затем рассматриваются все маршруты, в которых выбранная вершина стоит на предпоследнем месте.

Задание: на примере приведенной ниже программы реализовать алгоритм Дейкстры.

Программа для поиска кратчайшего пути между заданными вершинами графа с использованием алгоритма Дейкстры

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

class Program
{
    static void Main(string[] args)
    {
        var g = new Graph();

        //добавление вершин
        g.AddVertex("A");
        g.AddVertex("B");
        g.AddVertex("C");
        g.AddVertex("D");
        g.AddVertex("E");
        g.AddVertex("F");
        g.AddVertex("G");

        //добавление ребер
        g.AddEdge("A", "B", 22);
        g.AddEdge("A", "C", 33);
        g.AddEdge("A", "D", 61);
        g.AddEdge("B", "C", 47);
        g.AddEdge("B", "E", 93);
        g.AddEdge("C", "D", 11);
        g.AddEdge("C", "E", 79);
        g.AddEdge("C", "F", 63);
        g.AddEdge("D", "F", 41);
        g.AddEdge("E", "F", 17);
        g.AddEdge("E", "G", 58);
        g.AddEdge("F", "G", 84);

        var dijkstra = new Dijkstra(g);
        var path = dijkstra.FindShortestPath("A", "G");
        Console.WriteLine(path);
        Console.ReadLine();
    }
}
```

Контрольные вопросы:

1. В чем суть алгоритма Дейкстры?
2. Для чего применяется алгоритм Дейкстры?
3. Какие задачи можно решать с помощью алгоритма Дейкстры?
4. Как происходит инициализация в алгоритме Дейкстры?
5. Как происходит итерация в алгоритме Дейкстры?

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1 Залогова, Л.А. Основы объектно-ориентированного программирования на базе языка C#: учебное пособие / Л. А. Залогова. – 2-е изд., стер. – Санкт-Петербург : Лань, 2020. – 192 с. – ISBN 978-5-8114-4757-2. – Текст : электронный // Лань : электронно-библиотечная система. – URL: <https://e.lanbook.com/book/126160> – Режим доступа: для авториз. пользователей.

2 Исакова, А.И. Информационные технологии : учебное пособие / А.И. Исакова ; Томский Государственный университет систем управления и радиоэлектроники (ТУСУР), Кафедра автоматизированных систем управления (АСУ). – Томск : ТУСУР, 2013. – 207 с. : ил. – Режим доступа: по подписке. – URL: <https://biblioclub.ru/index.php?page=book&id=480610> (дата обращения: 13.01.2021). – Библиогр.: с. 197-198. – Текст : электронный.

3 Городня, Л. В. Парадигма программирования : учебное пособие для вузов / Л. В. Городня. – 2-е изд., стер. – Санкт-Петербург : Лань, 2021. — 232 с. — ISBN 978-5-8114-6680-1. – Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/151660> (дата обращения: 24.01.2021). — Режим доступа: для авториз. пользователей.

МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ РАСЧЕТНО-ГРАФИЧЕСКОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ

ОБЩИЕ ПОЛОЖЕНИЯ

Расчетно-графическая работа (РГР) содержит задания, которые следует выполнить в интегрированной среде разработки программного обеспечения Visual Studio на языке C# (или другом, по согласованию с преподавателем). Работающая программа должна иметь вид windows-приложения с оконным интерфейсом или использовать платформу WPF (Windows Presentation Foundation).

Номер N варианта каждого задания выбирается по номеру студента в списке журнала группы. **Номер варианта совпадает с номером по списку в журнале. Первые номера из таблицы 1 берет группа ПИЭ-1, за ними, по порядку, ПИЭ-2.**

На оценку «отлично» надо выполнить 5 заданий; на оценку «хорошо» надо выполнить 4 задания; на оценку «удовлетворительно» надо выполнить 3 задания; на оценку «неудовлетворительно» задания надо выполнять неудовлетворительно.

В расчетно-пояснительной записке (РПЗ) к РГР для каждого задания необходимо представить:

- текст задания;
- схему алгоритма (блок-схему), таблицу идентификаторов, а если программа содержит несколько модулей, то и схему взаимосвязи модулей;
- листинг программы, формы;
- результаты тестирования.

В конце РПЗ должен быть представлен список литературы, по тексту на него должны быть ссылки.

Требования к оформлению РГР: шрифт TimesNewRoman, одинарный межстрочный интервал, поля: правое – 10 мм, верхнее, левое и нижнее – 20 мм, абзац – 1.25. Более детальные требования представлены в «Требования_к_оформлениюРПЗ_2016.docx». Для более простого соблюдения требований оформления рекомендуется воспользоваться образцом оформления РГР.

ЗАДАНИЕ 1. ЦИКЛИЧЕСКИЕ АЛГОРИТМЫ. ТАБУЛИРОВАНИЕ ФУНКЦИЙ

Составить алгоритм программы вычисления значений функции $y(x)$ в M равноотстоящих друг от друга точках при изменении x на отрезке $[x1; x2]$. Вид функции $y(x)$ взять из таблицы 1.1 в соответствии с вариантом. Значения $M, x1, x2, A$ вводятся при работе программы.

Таблица 1.1

№	Функция $y = f(x)$
1	$y = \begin{cases} \frac{A}{2}(e^{x/A} + e^{-x/A}), & \text{при } x \leq 0 \\ 4A^3/(x^2 + 4A^2), & \text{при } x > 0 \end{cases}$
2	$y = \begin{cases} -\sqrt{\sqrt[3]{A^2} - \sqrt[3]{x^2}}, & \text{при } x < A \\ \ln x, & \text{при } x \geq A \end{cases}$
3	$y = \begin{cases} A \cos(x^2/2), & \text{при } x < 0 \\ A/2(e^{x/A} + e^{-x/A}), & \text{при } x \geq 0 \end{cases}$
4	$y = \begin{cases} A - 8A^3/(x^2 + 4A^2), & \text{при } x \leq 0 \\ -A\sqrt{1 - x^2/(4A^2)}, & \text{при } x > 0 \end{cases}$
5	$y = \begin{cases} -\sqrt{2A(-3A - x)}, & \text{при } x \leq -3A \\ A \sin(x + 3A), & \text{при } x > -3A \end{cases}$
6	$y = \begin{cases} -A\sqrt{1 - x^2/(4A^2)}, & \text{при } x < 0 \\ A/2(e^{x/A} + e^{-x/A}), & \text{при } x \geq 0 \end{cases}$
7	$y = \begin{cases} \sqrt{\sqrt[3]{A^2} - \sqrt[3]{x^2}}, & \text{при } x < 0 \\ A + \sqrt{x^3/(2A - x)}, & \text{при } x \geq 0 \end{cases}$
8	$y = \begin{cases} -\sqrt{A^2 - (x - A)^2}, & \text{при } x < 2A \\ A(1 - e^{A-x}), & \text{при } x \geq 2A \end{cases}$
9	$y = \begin{cases} -(x + 3A)^2 - 2A, & \text{при } x \leq -6A \\ A \cos(x + 3A) - 3A, & \text{при } x > -6A \end{cases}$
10	$y = \begin{cases} 2A\sqrt{1 - x^2/A^2}, & \text{при } x < 0 \\ A/2 \cos(x) + 3A/2, & \text{при } x \geq 0 \end{cases}$
11	$y = \begin{cases} A \cos(x + 1) + A, & \text{при } x \leq -1 \\ A(x + 2)^{3/2}, & \text{при } x > -1 \end{cases}$
12	$y = \begin{cases} 2A - A/2(e^{x/A} + e^{-x/A}), & \text{при } x \leq 0 \\ A + \sqrt{x^3/(2A - x)}, & \text{при } x > 0 \end{cases}$
13	$y = \begin{cases} A - \sqrt{A^2 - (x - A)^2}, & \text{при } x < A \\ A(x - A)^{3/2}, & \text{при } x \geq A \end{cases}$
14	$y = \begin{cases} -A \ln(-x - 2A), & \text{при } x < -2A \\ \sqrt{A^2 - (x + A)^2}, & \text{при } x \geq -2A \end{cases}$
15	$y = \begin{cases} A - Ae^{x-8A}, & \text{при } x < 8A \\ \sqrt{2Ax - 16A^2}, & \text{при } x \geq 8A \end{cases}$

16	$y = \begin{cases} \sqrt{\left(\sqrt[3]{4A^2} - \sqrt[3]{(x+A)^2}\right)^3}, & \text{при } x < 0 \\ -\sqrt{x^3/(2A-x)}, & \text{при } x \geq 0 \end{cases}$
17	$y = \begin{cases} \sqrt[3]{3A(1-\sqrt{-x^3})}, & \text{при } x < 0 \\ \sqrt{\sqrt{16A^4 + 4A^2x^2} - x^2 - A^2}, & \text{при } x \geq 0 \end{cases}$

18	$y = \begin{cases} -\sqrt{\sqrt{16A^4 + 4A^2(x+A)^2} - A^2}, & \text{при } x < 0 \\ \sqrt{A^2 - (x+A)^2} - 2A, & \text{при } x \geq 0 \end{cases}$
19	$y = \begin{cases} \sqrt{16A^2 - (x-4A)^2}, & \text{при } x < 4A \\ 8A^3 / ((x-4A)^2 + 4A^2), & \text{при } x \geq 4A \end{cases}$
20	$y = \begin{cases} \sqrt{A^2 - (x-A)^2} - A, & \text{при } x < 2A \\ -2A - Ae^{-(x-2A)}, & \text{при } x \geq 2A \end{cases}$
21	$y = \begin{cases} -Ae^{x-3A}, & \text{при } x < 3A \\ -A - A \ln(x-3A), & \text{при } x \geq 3A \end{cases}$
22	$y = \begin{cases} \sqrt{\left(\sqrt[3]{4A^2} - \sqrt[3]{(x+2A)^2}\right)^3}, & \text{при } x < 0 \\ -\sqrt{\left(\sqrt[3]{4A^2} - \sqrt[3]{(x+2A)^2}\right)^3}, & \text{при } x \geq 0 \end{cases}$
23	$y = \begin{cases} \sqrt{\sqrt{16A^4 + 4A^2x^2} - x^2 - A^2}, & \text{при } x < 0 \\ A\sqrt{3} + \sqrt{\left(\sqrt[3]{A^2} - \sqrt[3]{(x-A)^2}\right)^3}, & \text{при } x \geq 0 \end{cases}$
24	$y = \begin{cases} \sqrt{A^2 - (x+A)^2} - A, & \text{при } x \leq 0 \\ 2A\sqrt{1-x^2}/(4A)^2, & \text{при } x > 0 \end{cases}$
25	$y = \begin{cases} \frac{8A^3}{x^2 + 4A^2}, & \text{при } x \leq 0 \\ A(e^{x/A} + e^{-x/A}), & \text{при } x > 0 \end{cases}$
26	$y = \begin{cases} Ae^{(x+3A)/A} + e^{-(x+3A)/A}, & \text{при } x < -3A \\ A + A \cos(x+3A), & \text{при } x \geq -3A \end{cases}$
27	$y = \begin{cases} A - 8A/(x^2 + 4A^2), & \text{при } x < 0 \\ \sqrt{\left(\sqrt[3]{A^2} - \sqrt[3]{(x-A)^2}\right)^3}, & \text{при } x \geq 0 \end{cases}$

28	$y = \begin{cases} 2A - A/2(e^{(x-2A)/A} + e^{-(x-2A)/A}), & \text{при } x < 2A \\ A + \sqrt{Ax^3}, & \text{при } x \geq 2A \end{cases}$
29	$y = \begin{cases} \sqrt{\left(\sqrt[3]{4A^2} - \sqrt[3]{x^2}\right)^3}, & \text{при } x \leq 0 \\ A + \sqrt{\left(\sqrt[3]{A^2} - \sqrt[3]{(x-A)^2}\right)^3}, & \text{при } x > 0 \end{cases}$
30	$y = \begin{cases} -8A^3/(x^2 + 4A^2), & \text{при } x \leq 0 \\ 2A\sqrt{1-x^2/A^2} - 4A, & \text{при } x > 0 \end{cases}$
31	$y = \begin{cases} -\sqrt[3]{\sqrt{16A^4 + 4A^2(x+A)^2} - A^2}, & \text{при } x < 0 \\ \sqrt{A^2 - (x+A)^2} - \sin(x), & \text{при } x \geq 0 \end{cases}$
32	$y = \begin{cases} \sqrt[4]{16A^2 - (x-4A)^2}, & \text{при } x < 4A \\ 8A^3/((x-4A)^2 + 4A^2), & \text{при } x \geq 4A \end{cases}$
33	$y = \begin{cases} \sqrt{A^2 - (x-A)^2} - A, & \text{при } x < 2A \\ -2A - Ae^{-\cos(x)}, & \text{при } x \geq 2A \end{cases}$
34	$y = \begin{cases} -Ae^{x-3A}, & \text{при } x < 3A \\ -A - x \ln(x-3A), & \text{при } x \geq 3A \end{cases}$
35	$y = \begin{cases} \sqrt{\left(\sqrt[3]{4A^2} - \sqrt[3]{(x+2A)^2}\right)^3}, & \text{при } x < 0 \\ -\sqrt{\left(\sqrt[3]{4A^2} - \sqrt[3]{(x+2A)^2}\right)^{1/3}}, & \text{при } x \geq 0 \end{cases}$
36	$y = \begin{cases} \sqrt{\sqrt{16A^4 + 4A^2x^2} - x^2 - A^2}, & \text{при } x < 0 \\ A\sqrt{3} + \sqrt{\left(\sqrt[3]{A^2} - e^{(-x)} \sin(Ax)\right)^3}, & \text{при } x \geq 0 \end{cases}$

37	$y = \begin{cases} -\sqrt{\sqrt{16A^4 + 4A^2(x+A)^2} - A^2}, & \text{при } x < 0 \\ \sqrt{A^2 - (x+A)^2} - 2A, & \text{при } x \geq 0 \end{cases}$
38	$y = \begin{cases} \sqrt{16A^2 - (x-4A)^2}, & \text{при } x < 16A \\ 8A^3 / ((x-4A)^2 + 4A^2), & \text{при } x \geq 16A \end{cases}$
39	$y = \begin{cases} \sqrt{A^2 - (x-A)^2} - A, & \text{при } x < 2A \\ -2A - Ae^{-(x-2A)}, & \text{при } x \geq 2A \end{cases}$
40	$y = \begin{cases} -Ae^{x-3A}, & \text{при } x < 3A \\ -A - A \ln(x-3A), & \text{при } x \geq 3A \end{cases}$
41	$y = \begin{cases} \sqrt{\left(\sqrt[3]{4A^2} - \sqrt[3]{(x+2A)^2}\right)^3}, & \text{при } x < 0 \\ -\sqrt{\left(\sqrt[3]{4A^2} - \sqrt[3]{(x+2A)^2}\right)^3}, & \text{при } x \geq 0 \end{cases}$
42	$y = \begin{cases} \sqrt{\sqrt{16A^4 + 4A^2x^2} - x^2 - A^2}, & \text{при } x < 0 \\ A\sqrt{3} + \sqrt{\left(\sqrt[3]{A^2} - \sqrt[3]{(x-A)^2}\right)^3}, & \text{при } x \geq 0 \end{cases}$
43	$y = \begin{cases} \sqrt{A^2 - (x+A)^2} - A, & \text{при } x \leq 0 \\ 2A\sqrt{1-x^2} / (4A)^2, & \text{при } x > 0 \end{cases}$
44	$y = \begin{cases} \frac{8A^3}{x^2 + 4A^2}, & \text{при } x \leq 0 \\ A(e^{x/A} + e^{-x/A}), & \text{при } x > 0 \end{cases}$

ЗАДАНИЕ 2. ИТЕРАЦИОННЫЕ ЦИКЛЫ

Составить алгоритм и написать программу вывода таблицы из M значений функции $f(x)$ на отрезке $[a, b]$ с шагом h . Вид функции $f(x)$ взять из таблицы 2. Функция представлена и в виде формулы, и в виде бесконечного ряда (суммируя члены ряда можно получить примерно такое же значение $f(x)$, что и рассчитанное по формуле). Чем больше членов ряда суммируется, тем точнее вычисляется $f(x)$, то есть, ее значение получается ближе к рассчитанному по формуле.

Для пояснения задачи обозначим результат вычисления $f(x)$ с помощью ряда как $f_{\text{поряду}}(x)$, а с помощью формулы как $f_{\text{по формуле}}(x)$. Сложение членов ряда надо выполнять до тех пор, пока не будет достигнута заданная точность ε , то есть, не выполнится условие:

$$\varepsilon > |f_{\text{поряду}}(x)_i|,$$

где $f_{\text{поряду}}(x)_i$ – значение i -го члена ряда при расчете значения функции $f(x)$ в точке x .

Входные данные программы: M, a, b, h, ε .

Выходные данные программы: таблица значений функции, структура которой показана на рисунке 2.1. Рассчитанная величина погрешности для каждой точки x вычисляется по формуле:

$$\varepsilon_{\text{расч}} = |f_{\text{поряду}}(x) - f_{\text{по формуле}}(x)|.$$

№	Значения x	Значения функции, вычисленные по формуле	Значения функции, вычисленные с помощью ряда	$\varepsilon_{\text{расч}}$
1	a	$f_{\text{по формуле}}(a)$	$f_{\text{поряду}}(a)$	0.01
2	$a+h$	$f_{\text{по формуле}}(a+h)$	$f_{\text{поряду}}(a+h)$	0.00
3	$a+2h$	$f_{\text{по формуле}}(a+2h)$	$f_{\text{поряду}}(a+2h)$	0.02
4	$a+3h$	$f_{\text{по формуле}}(a+3h)$	$f_{\text{поряду}}(a+3h)$...
...
...	$b-h$	$f_{\text{по формуле}}(b-h)$	$f_{\text{поряду}}(b-h)$...
	b	$f_{\text{по формуле}}(b)$	$f_{\text{поряду}}(b)$...

Рисунок 2.1 – Структура таблицы расчета функции $f(x)$

Таблица 2

№	f(x)	
	Суммируемый ряд	Формула
1	$\frac{2x^2(3+x)}{3!} + \frac{2x^6(7+x)}{7!} + \dots + \frac{2x^{4k-2} \cdot (4k-1+x)}{(4k-1)!} + \dots$	$e^x - \sin x - \cos x$
2	$\frac{3x^3}{1!} + \frac{5x^5}{2!} + \frac{7x^7}{3!} + \dots + \frac{2k+1}{k!} x^{2k+1} + \dots$	$(x+2x^3) \cdot e^{x^2} - x$
3	$\frac{x}{3!} - \frac{x^3}{5!} + \frac{x^5}{7!} - \dots + (-1)^{k-1} \cdot \frac{x^{2k-1}}{(2k+1)!} + \dots$	$\frac{x - \sin x}{x^2}$
4	$\frac{x-1}{x+1} + \frac{1}{3} \left(\frac{x-1}{x+1}\right)^3 + \dots + \frac{1}{2k-1} \left(\frac{x-1}{x+1}\right)^{2k-1} + \dots$	$\frac{1}{2} \ln x$
5	$\frac{x^4}{4!} + \frac{x^8}{8!} + \frac{x^{12}}{12!} + \dots + \frac{x^{4k}}{(4k)!} + \dots$	$\frac{(e^x + e^{-x} + 2 \cos x)}{4} - 1$
6	$\frac{3x^2}{2!} - \frac{9x^4}{4!} + \frac{19x^6}{6!} - \dots + (-1)^{k-1} \cdot \frac{(2k^2+1)x^{2k}}{(2k)!} + \dots$	$1 + \frac{x}{2} \sin x + \left(\frac{x^2}{2} - 1\right) \cos x$
7	$\frac{(x-1)^2}{1} - \frac{(x-1)^4}{2} + \frac{(x-1)^6}{3} - \dots + (-1)^{k-1} \cdot \frac{(x-1)^{2k}}{k} + \dots$	$\ln(2 - 2x + x^2)$
8	$\frac{4x}{1!} + \frac{4x^5}{5!} + \frac{4x^9}{9!} + \dots + \frac{4x^{4k-3}}{(4k-3)!} + \dots$	$e^x + 2 \sin x - e^{-x}$
9	$\frac{1^2+1}{1!} \left(\frac{x}{2}\right) + \frac{2^2+1}{2!} \left(\frac{x}{2}\right)^2 + \dots + \frac{k^2+1}{k!} \left(\frac{x}{2}\right)^k + \dots$	$\left(\frac{x^2}{4} + \frac{x}{2} + 1\right) e^{\frac{x}{2}} - 1$
10	$\frac{x^2}{2!} - \frac{3x^4}{4!} + \frac{5x^6}{6!} - \dots + (-1)^{k-1} \cdot \frac{(2k-1)x^{2k}}{(2k)!} + \dots$	$\cos x + x \sin x - 1$
11	$\frac{4x}{1!} + \frac{8x^3}{3!} + \frac{12x^5}{5!} + \dots + \frac{4k \cdot x^{2k-1}}{(2k-1)!} + \dots$	$(x+1)e^x + (x-1)e^{-x}$
12	$\frac{3x^2}{4!} - \frac{5x^4}{6!} + \frac{7x^6}{8!} - \dots + (-1)^{k-1} \frac{(2k+1)x^{2k}}{(2k+2)!} + \dots$	$\frac{1 - \cos x}{x^2} - \frac{\sin x}{x} + \frac{1}{2}$
13	$\frac{2x-1}{2 \cdot 3} x^3 + \frac{4x-1}{4 \cdot 5} x^5 + \dots + \frac{2k \cdot x - 1}{2k \cdot (2k+1)} x^{2k+1} + \dots$	$\ln \frac{(1+x)^{x+1/2}}{\sqrt{1-x}} - x^2 - x$
14	$\frac{8\sqrt{x} \cdot x}{3!} + \frac{32\sqrt{x} \cdot x^2}{5!} + \frac{72\sqrt{x} \cdot x^3}{7!} + \dots + \frac{8k^2 \sqrt{x} \cdot x^k}{(2k+1)!} + \dots$	$(x+1-\sqrt{x})e^{\sqrt{x}} - (x+1+\sqrt{x})e^{-\sqrt{x}}$
15	$\frac{x}{1} + \frac{x^3}{2} + \frac{x^5}{3} + \dots + \frac{x^{2k-1}}{k} + \dots$	$-\frac{1}{x} \ln(1-x^2)$
16	$\frac{2x^4(5+x)}{5!} + \frac{2x^8(9+x)}{9!} + \dots + \frac{2x^{4k}(4k+1+x)}{(4k+1)!} + \dots$	$\sin x - 2x - 2 + \cos x + e^x$
17	$\frac{x(4-2x+x)}{1 \cdot 2} + \frac{x^3(8-4x+x)}{3 \cdot 4} + \dots + \frac{x^{2k-1}(4k-2kx+x)}{(2k-1) \cdot 2k} + \dots$	$\ln \sqrt{(1+x)^3 / (1-x)}$

18	$\frac{(2x)^2}{2!} - \frac{(2x)^4}{4!} + \frac{(2x)^6}{6!} - \dots + (-1)^{k-1} \frac{(2x)^{2k}}{(2k)!} + \dots$	$2 \sin^2 x$
19	$\frac{x^4}{3!} + \frac{x^8}{7!} + \frac{x^{12}}{11!} + \dots + \frac{x^{4k}}{(4k-1)!} - \dots$	$\frac{x}{4}(e^x - 2 \sin x - e^{-x})$
20	$\frac{\ln 3}{1!} x + \frac{\ln^2 3}{2!} x^2 + \frac{\ln^3 3}{3!} x^3 + \dots + \frac{\ln^k 3}{k!} x^k + \dots$	$3^x - 1$
21	$\frac{x(3+x)}{3!} - \frac{x^3(5+x)}{5!} + \dots + (-1)^{k-1} \frac{x^{2k-1}(2k+1+x)}{(2k+1)!} + \dots$	$\frac{1 - \cos x - \sin x}{x} + 1$
22	$\frac{x^2}{4!} - \frac{x^4}{6!} + \frac{x^6}{8!} - \dots + (-1)^{k-1} \frac{x^{2k}}{(2k+2)!} + \dots$	$\frac{\cos x - 1}{x^2} + \frac{1}{2}$
23	$\frac{x}{2!} + \frac{x^5}{6!} + \frac{x^9}{10!} - \dots + \frac{x^{4k-3}}{(4k-2)!} + \dots$	$\frac{e^x - 2 \cos x + e^{-x}}{4x}$
24	$\frac{3}{4} \frac{5}{6} \frac{x^7}{7} + \frac{3}{4} \frac{5}{6} \frac{7}{8} \frac{9}{10} \frac{x^{11}}{11} + \dots + \frac{3}{4} \frac{5}{6} \dots \frac{(4k-1)(4k+1)}{4k(4k+2)} \frac{x^{4k+3}}{(4k+3)}$	$\frac{-x^3}{3} + \arcsin x - \ln(x + \sqrt{x^2 + 1})$
25	$x + \frac{1}{2} \frac{3}{4} \frac{x^5}{5} + \frac{1}{2} \frac{3}{4} \frac{5}{6} \frac{7}{8} \frac{x^9}{9} + \dots + \frac{1}{2} \frac{3}{4} \dots \frac{(4k-7)(4k-5)}{(4k-6)(4k-4)} \frac{x^{4k-5}}{(4k-5)} + \dots$	$\frac{\arcsin x - \ln(x + \sqrt{x^2 + 1})}{2}$
26	$\frac{x^2}{2} - \frac{x^4}{12} + \frac{x^6}{30} - \dots + (-1)^{k-1} \frac{x^{2k}}{2k(2k-1)} + \dots$	$x \cdot \operatorname{arctg}(x) - \ln \sqrt{1+x^2}$
27	$\frac{4x^5}{5} + \frac{4x^9}{9} + \frac{4x^{13}}{13} + \dots + \frac{4x^{4k+1}}{4k+1} + \dots$	$2 \operatorname{arctg}(x) - 4x + \ln \left(\frac{1+x}{1-x} \right)$
28	$\frac{2}{3} \frac{x^3}{1+x^2} + \frac{2 \cdot 4}{3 \cdot 5} \frac{x^5}{(1+x^2)^2} + \dots + \frac{2 \cdot 4 \cdot \dots \cdot 2k}{3 \cdot 5 \cdot \dots \cdot (2k+1)} \frac{x^{2k+1}}{(1+x^2)^k} + \dots$	$(1+x^2) \operatorname{arctg}(x) - x$
29	$\frac{x}{3} - \frac{x^3}{5} + \frac{x^5}{7} - \dots + (-1)^{k-1} \frac{x^{2k-1}}{2k+1} + \dots$	$\frac{x - \operatorname{arctg}(x)}{x^2}$
30	$\frac{2x^3}{3} - \frac{2x^5}{15} + \frac{2x^7}{35} - \dots + (-1)^{k-1} \frac{2x^{2k+1}}{4k^2-1} + \dots$	$(1+x^2) \operatorname{arctg}(x) - x$

31	$\sin(x) + \frac{(2x)^2}{2!} - \frac{(2x)^4}{4!} + \frac{(2x)^6}{6!} - \dots + (-1)^{k-1} \frac{(2x)^{2k}}{(2k)!} + \dots$	$\sin(x) + 2\sin^2 x$
32	$x + \frac{x^4}{3!} + \frac{x^8}{7!} + \frac{x^{12}}{11!} + \dots + \frac{x^{4k}}{(4k-1)!} - \dots$	$x + \frac{x}{4}(e^x - 2\sin x - e^{-x})$
33	$\cos(2x) + \frac{\ln 3}{1!}x + \frac{\ln^2 3}{2!}x^2 + \frac{\ln^3 3}{3!}x^3 + \dots + \frac{\ln^k 3}{k!}x^k + \dots$	$3^x - 1 + \cos(2x)$
34	$3 + \frac{x(3+x)}{3!} - \frac{x^3(5+x)}{5!} + \dots + (-1)^{k-1} \frac{x^{2k-1}(2k+1+x)}{(2k+1)!} + \dots$	$\frac{1 - \cos x - \sin x}{x} + 4$
35	$x^2 + \frac{x^2}{4!} - \frac{x^4}{6!} + \frac{x^6}{8!} - \dots + (-1)^{k-1} \frac{x^{2k}}{(2k+2)!} + \dots$	$\frac{\cos x - 1}{x^2} + \frac{2x^2 + 1}{2}$
36	$\sin(x) + \frac{x}{2!} + \frac{x^5}{6!} + \frac{x^9}{10!} - \dots + \frac{x^{4k-3}}{(4k-2)!} + \dots$	$\frac{e^x - 2\cos x + e^{-x} + 4x\sin x}{4x}$
37	$\frac{2x^2(3+x)}{3!} + \frac{2x^6(7+x)}{7!} + \dots + \frac{2x^{4k-2} \cdot (4k-1+x)}{(4k-1)!} + \dots$	$e^x - \sin x - \cos x$
38	$\frac{3x^3}{1!} + \frac{5x^5}{2!} + \frac{7x^7}{3!} + \dots + \frac{2k+1}{k!}x^{2k+1} + \dots$	$(x + 2x^3) \cdot e^{x^2} - x$
39	$\frac{x}{3!} - \frac{x^3}{5!} + \frac{x^5}{7!} - \dots + (-1)^{k-1} \frac{x^{2k-1}}{(2k+1)!} + \dots$	$\frac{x - \sin x}{x^2}$
40	$\frac{x-1}{x+1} + \frac{1}{3} \left(\frac{x-1}{x+1}\right)^3 + \dots + \frac{1}{2k-1} \left(\frac{x-1}{x+1}\right)^{2k-1} + \dots$	$\frac{1}{2} \ln x$
41	$\frac{x^4}{4!} + \frac{x^8}{8!} + \frac{x^{12}}{12!} + \dots + \frac{x^{4k}}{(4k)!} + \dots$	$\frac{(e^x + e^{-x} + 2\cos x)}{4} - 1$
42	$\frac{3x^2}{2!} - \frac{9x^4}{4!} + \frac{19x^6}{6!} - \dots + (-1)^{k-1} \frac{(2k^2+1)x^{2k}}{(2k)!} + \dots$	$1 + \frac{x}{2}\sin x + \left(\frac{x^2}{2} - 1\right)\cos x$
43	$\frac{(x-1)^2}{1} - \frac{(x-1)^4}{2} + \frac{(x-1)^6}{3} - \dots + (-1)^{k-1} \frac{(x-1)^{2k}}{k} + \dots$	$\ln(2 - 2x + x^2)$
44	$\frac{x^4}{3!} + \frac{x^8}{7!} + \frac{x^{12}}{11!} + \dots + \frac{x^{4k}}{(4k-1)!} - \dots$	$\frac{x}{4}(e^x - 2\sin x - e^{-x})$

ЗАДАНИЕ 3. ПОДПРОГРАММЫ. ОБРАБОТКА МАССИВОВ

Разработать функцию (метод класса), входными данными которой являются два массива $X(n)$ и $Y(n)$ (или один из них) или матрица $A(n, n)$. Преобразование, выполняемое функцией, задано в таблице 3.1. В головной программе осуществить ввод размеров массивов, ручной ввод их элементов, вызов функции, отображение исходного массива и результатов преобразований.

Таблица 3.1 – Преобразование, выполняемое функцией

№	Функция должна определять:
1	Число перемен знака в массиве $X(n)$.
2	Количество положительных элементов в двух заданных строках матрицы $A(n, n)$.
3	Наименьший элемент в совокупности элементов двух массивов $X(n)$, $Y(n)$.
4	Абсолютная величина разности максимальных элементов двух заданных столбцов матрицы $A(n, n)$.
5	Общее количество нулей в i -ой и последней строке, i -ом и последнем столбце матрицы $A(n, n)$.
6	Количество локальных минимумов матрицы $A(n, n)$.
7	Среднее арифметическое элементов над главной диагональю матрицы $A(n, n)$.
8	Количество строк матрицы $A(n, n)$, сумма элементов каждой из которых меньше нуля.
9	Максимальный элемент в заданной группе соседних строк матрицы $A(n, n)$.
10	Наибольшее число подряд идущих положительных элементов среди $X(n)$.

11	Разность сумм элементов над и под главной диагональю матрицы $A(n, n)$.
12	Общее количество отрицательных элементов на главной диагонали и на двух соседних с ней (сверху и снизу) диагоналях матрицы $A(n, n)$.
13	Наименьшая сумма строки в матрице $A(n, n)$.
14	Наибольший из минимальных элементов строк матрицы $A(n, n)$.
15	Количество элементов среди $X(n)$, значения которых совпадают со значениями элементов массива $Y(n)$.
16	Сумма отрицательных элементов массива $Y(n)$.
17	Произведение положительных элементов среди элементов $X(n)$.
18	Полусумма минимального и максимального элементов массива $X(n)$.
19	Значение многочлена $Y_1Z^{n-1} + Y_2Z^{n-2} + \dots + Y_{n-1}Z + Y_n$. Z -число, вводится с клавиатуры
20	Количество нулей в массиве $X(n)$.

21	Наибольшая абсолютная величина элемента среди $Y(n)$.
22	Число элементов массива $Y(n)$, значения которых совпадают со значениями $X(n)$.
23	Скалярное произведение, равное $\sum_{i=1}^n X_i Y_i$.
24	Произведение максимальных элементов исходных массивов.
25	Число элементов массива $X(n)$, которые больше максимального элемента в массиве $Y(n)$.
26	Число элементов среди $X(n)$, которые не превосходят максимального элемента $Y(n)$ и то же время не меньше его минимального элемента.
27	Число элементов массива $X(n)$, которые делятся на 7 без остатка.
28	Значение наибольшего элемента главной диагонали матрицы $A(n, n)$.
29	Общее количество локальных максимумов в строках матрицы $A(n, n)$.
30	Расстояние от начала координат до точки n -мерного пространства с координатами X_1, X_2, \dots, X_n .

31	Сформировать матрицу $V=AA^T$. Поменять местами минимальные элементы матриц A и B . Также, поменять местами максимальные элементы матриц A и B .
32	Отобразить элементы матрицы A симметрично относительно k -го столбца или строки (по выбору пользователя)
33	Даны две одинаковые матрицы A и B . Создать новую матрицу C , каждый элемент которой C_{ij} равен произведение суммы элементов i -ой строки матрицы A и j -го столбца матрицы B .
34	Дана матрица A . Найти номер строки, элементы которой образуют наиболее длинную возрастающую последовательность и вывести длину этой последовательности.
35	Дана матрицы A и B , причем количество строк и столбцов в матрице A не больше чем в B . Определить, является ли матрица A фрагментом матрицы B . Определить количество фрагментов, если их несколько.
36	Даны две одинаковые квадратные матрицы A и B . Создать новую матрицу C , элементы главной диагонали которой C_{ii} содержит суммы элементов соответствующих i -х строк матрицы A . Все остальные элементы C содержат квадраты элементов матрицы B .
37	Дана матрица A . Составить новую матрицу-столбец C , каждый элемент которой C_i содержит номер элемента i -й строки A , с которого начинается возрастающая последовательность наибольшей длины.
38	Разность сумм элементов над и под главной диагональю матрицы $A(n, n)$.
39	Общее количество отрицательных элементов на главной диагонали и на двух соседних с ней (сверху и снизу) диагоналях матрицы $A(n, n)$.
40	Наименьшая сумма строки в матрице $A(n, n)$.
41	Наибольший из минимальных элементов строк матрицы $A(n, n)$.
42	Количество элементов среди $X(n)$, значения которых совпадают со значениями элементов массива $Y(n)$.
43	Сумма отрицательных элементов массива $Y(n)$.
44	Произведение положительных элементов среди элементов $X(n)$.
45	Полусумма минимального и максимального элементов массива $X(n)$.
46	Значение многочлена $Y_1Z^{n-1} + Y_2Z^{n-2} + \dots + Y_{n-1}Z + Y_n$. Z - число, вводится с клавиатуры

ЗАДАНИЕ 4. МНОГОМОДУЛЬНЫЕ ПРИЛОЖЕНИЯ

Разработать модуль, содержащий описание методов (процедуры и функции) из таблицы 4.1. Подключить разработанный модуль к приложению и продемонстрировать его работу. В головной программе приложения должны вводиться исходные данные для работы методов, потом эти данные должны передаваться в модуль для обработки и результаты обработки должны **возвращаться** в головную программу.

Таблица 4.1

№	Алгоритмы функции и процедуры
1	Функция: Вычисление площади трапеции. Процедура: Перевод целого числа из 10-ой с.с. в двоичную с.с.
2	Функция: Определение простоты числа. Процедура: Перевод целого числа из 10-ой с.с. в восьмеричную с.с.
3	Функция: Вычисление площади сегмента окружности. Процедура: Перевод целого числа из 10-ой с.с. в шестнадцатеричную с.с.
4	Функция: Вычисление поверхности прямого конуса по его радиусу основания и высоте. Процедура: Перевод целого числа из 2-ой с.с. в восьмеричную с.с.
5	Функция: Вычисление площади шарового сегмента. Процедура: Перевод целого числа из 2-ой с.с. в 16-тиричную с.с.
6	Функция: Вычисление интеграла по формуле трапеций. Процедура: Перевод целого числа из 2-ой с.с. в десятичную с.с.
7	Функция: Вычисление площади правильного б-тиугольника по известной его диагонали. Процедура: Перевод целого числа из 8-ой с.с. в двоичную с.с.
8	Функция: Вычисление интеграла по формуле прямоугольника. Процедура: Перевод целого числа из 10-ой с.с. в двоичную с.с.

9	Функция: Вычисление площади шарового сектора. Процедура: Сложение двух двоичных чисел.
10	Функция: Вычисление площади правильного N -угольника. Процедура: Вычитание двух двоичных чисел.
11	Функция: Вычисление дискриминанта квадратного уравнения. Процедура: Сложение двух восьмеричных чисел.
12	Функция: Вычисление площади треугольника по известным его сторонам. Процедура: Сложение двух 16-ричных чисел.
13	Функция: Вычисление суммы углов правильного N -угольника. Процедура: Получение дополнительного кода двоичного числа.
14	Функция: Вычисление площади части кольца. Процедура: Получение дополнительного кода 16-тиричного числа.
15	Функция: Вычисление площади шарового сегмента. Процедура: Сложение двух 16-ричных чисел.
16	Функция: Вычисление интеграла по формуле трапеций. Процедура: Перевод целого числа из 10-ой с.с. в двоичную с.с.
17	Функция: Вычисление площади правильного 6-тиугольника по известной его диагонали. Процедура: Сложение двух восьмеричных чисел.
18	Функция: Вычисление площади части кольца. Процедура: Перевод целого числа из 10-ой с.с. в двоичную с.с.
19	Функция: Вычисление площади трапеции. Процедура: Сложение двух двоичных чисел.
20	Функция: Вычисление площади треугольника по известным его сторонам. Процедура: Перевод целого числа из 10-ой с.с. в восьмеричную с.с.
21	Функция: Определение простоты числа. Процедура: Сложение двух двоичных чисел.
22	Функция: Вычисление суммы углов правильного N -угольника. Процедура: Перевод целого числа из 10-ой с.с. в шестнадцатеричную с.с.
23	Функция: Вычисление интеграла по формуле прямоугольника. Процедура: Получение дополнительного кода двоичного числа.
24	Функция: Вычисление поверхности прямого конуса по его радиусу основания и высоте. Процедура: Получение дополнительного кода 16-тиричного числа.
25	Функция: Вычисление площади правильного N -угольника. Процедура: Вычитание двух двоичных чисел.

26	Функция: Вычисление площади правильного 7-угольника Процедура: Перевод целого числа из 13-ой с.с. в двоичную с.с.
27	Функция: Определение факториала числа. Процедура: Перевод целого числа из 9-ой с.с. в восьмеричную с.с.
28	Функция: Вычисление объема сегмента шара Процедура: Перевод целого числа из 11-ой с.с. в шестнадцатеричную с.с.
29	Функция: Вычисление поверхности косого конуса по его радиусу основания и высоте. Процедура: Перевод целого числа из 5-ой с.с. в восьмеричную с.с.
30	Функция: Вычисление объема фрагмента пересечения двух кубов Процедура: Перевод целого числа из 7-ой с.с. в 16-тиричную с.с.
31	Функция: Вычисление интеграла по формуле левых прямоугольников Процедура: Перевод целого числа из 8-ой с.с. в десятичную с.с.
32	Функция: Вычисление площади правильного 8-угольника по известной его диагонали. Процедура: Перевод целого числа из 14-ой с.с. в двоичную с.с.
33	Функция: Вычисление интеграла по формуле средних прямоугольников Процедура: Перевод целого числа из 10-ой с.с. в двоичную с.с.
34	Функция: Вычисление площади шарового сектора. Процедура: Сложение двух двоичных чисел.
35	Функция: Вычисление площади правильного N -угольника. Процедура: Вычитание двух двоичных чисел.
36	Функция: Вычисление дискриминанта квадратного уравнения. Процедура: Сложение двух восьмеричных чисел.
37	Функция: Вычисление площади треугольника по известным его сторонам. Процедура: Сложение двух 16-ричных чисел.
38	Функция: Вычисление суммы углов правильного N -угольника. Процедура: Получение дополнительного кода двоичного числа.
39	Функция: Вычисление площади части кольца. Процедура: Получение дополнительного кода 16-тиричного числа.
40	Функция: Вычисление площади шарового сегмента. Процедура: Сложение двух 16-ричных чисел.

41	Функция: Вычисление интеграла по формуле трапеций. Процедура: Перевод целого числа из 10-ой с.с. в двоичную с.с.
42	Функция: Вычисление площади правильного 6-тиугольника по известной его диагонали. Процедура: Сложение двух восьмеричных чисел.
43	Функция: Вычисление площади части кольца. Процедура: Перевод целого числа из 10-ой с.с. в двоичную с.с.
44	Функция: Вычисление площади трапеции. Процедура: Сложение двух двоичных чисел.
45	Функция: Вычисление площади треугольника по известным его сторонам. Процедура: Перевод целого числа из 10-ой с.с. в восьмеричную с.с.

















































ЗАДАНИЕ 5. ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Описать иерархию классов для работы с изображениями. Приложение должно выполнять следующие действия:

- изменение размера объекта;
- изменение цвета объекта;
- движение объекта – вверх, вниз, влево, вправо и по диагонали.

Приложение должно демонстрировать основные принципы ООП. Классы должны содержать поля, методы, свойства, конструкторы. Вид (значок) графического объекта представлен в таблице 5.1.

Таблица 5.1 – Значки графических объектов

№	Рисунок	№	Рисунок	№	Рисунок	№	Рисунок
1		9		17		25	
2		10		18		26	
3		11		19		27	
4		12		20		28	
5		13		21		29	
6		14		22		30	
7		15		23		31	
8		16		24		32	
33		34		35		36	
37		38		39		40	
41		42		43		44	
45		46		47		48	

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Ишкова Э. А. Самоучитель C#. Начала программирования. Серия: Самоучитель. Издательство: Наука и Техника, 2013 г. 496 с.
2. Васильев А. Программирование на C# для начинающих. Основные сведения. Серия: Российский компьютерный бестселлер. – М.: Эксмо, 2018. – 592 с.
3. Справочник по C#. URL: <https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/>
4. Пахомов Б. И. C# для начинающих. — СПб.: БХВ-Петербург, 2014. — 432 с. URL: https://lesmatveev.narod.ru/knigi_project/ci_sharp.pdf