Пермский филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский университет

«Высшая школа экономики»

Факультет экономики, менеджмента и бизнес-информатики

Образовательная программа бакалавриата «Программная инженерия»

**ОТЧЕТ  
по учебной практике**

Выполнил студент группы ПИ-19-1 Соломатин Роман Игоревич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Проверил:

Руководитель практики   
преподаватель кафедры информационных

технологий в бизнесе

Крючков Михаил Викторович

\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

Пермь, 2019 год

# Аннотация

В данном отчете описаны этапы проектирования, разработки и тестирования 12 задач. Программы реализованы на языке высокого уровня C#.

Работа выполнена студентом НИУ ВШЭ Пермь Соломатиным Романом Игоревичем. Кафедра информационных технологий в бизнесе.

Работа содержит **NNNN** страниц формата А4 основного текста, включая 12 глав соответствующей каждой учебной задаче.

Библиографический список включает в себя 4 источника.

В каждой главе содержится анализ задачи (установка функциональных и не функциональных требований к задаче), проектирование алгоритмов и тестирования ПО. Приложения к задачам содержат реализацию задач.

Оглавление

[Аннотация 1](#_Toc42978393)

[Введение 5](#_Toc42978394)

[1. Задача 1 6](#_Toc42978395)

[1.1. Анализ 7](#_Toc42978396)

[1.2. Проектирование 8](#_Toc42978397)

[1.3. Тестирование 9](#_Toc42978398)

[2. Задача 2 10](#_Toc42978399)

[2.1. Анализ 10](#_Toc42978400)

[2.2. Проектирование 11](#_Toc42978401)

[2.3. Тестирование 12](#_Toc42978402)

[3. Задача 3 13](#_Toc42978403)

[3.1. Анализ 13](#_Toc42978404)

[3.2. Проектирование 14](#_Toc42978405)

[3.3. Тестирование 14](#_Toc42978406)

[4. Задача 4 15](#_Toc42978407)

[4.1. Анализ 15](#_Toc42978408)

[4.2. Проектирование 15](#_Toc42978409)

[4.3. Тестирование 16](#_Toc42978410)

[5. Задача 5 17](#_Toc42978411)

[5.1. Анализ 17](#_Toc42978412)

[5.2. Проектирование 18](#_Toc42978413)

[5.3. Тестирование 19](#_Toc42978414)

[6. Задача 6 20](#_Toc42978415)

[6.1. Анализ 20](#_Toc42978416)

[6.2. Проектирование 20](#_Toc42978417)

[6.3. Тестирование 21](#_Toc42978418)

[7. Задача 7 23](#_Toc42978419)

[7.1. Анализ 23](#_Toc42978420)

[7.2. Проектирование 23](#_Toc42978421)

[7.3. Тестирование 24](#_Toc42978422)

[8. Задача 8 26](#_Toc42978423)

[8.1. Анализ 26](#_Toc42978424)

[8.2. Проектирование 26](#_Toc42978425)

[8.3. Тестирование 27](#_Toc42978426)

[9. Задача 9 28](#_Toc42978427)

[9.1. Анализ 28](#_Toc42978428)

[9.2. Проектирование 28](#_Toc42978429)

[9.3. Тестирование 29](#_Toc42978430)

[10. Задача 10 30](#_Toc42978431)

[10.1. Анализ 30](#_Toc42978432)

[10.2. Проектирование 30](#_Toc42978433)

[10.3. Тестирование 31](#_Toc42978434)

[11. Задача 11 33](#_Toc42978435)

[11.1. Анализ 33](#_Toc42978436)

[11.2. Проектирование 34](#_Toc42978437)

[11.3. Тестирование 35](#_Toc42978438)

[12. Задача 12 37](#_Toc42978439)

[12.1. Анализ 37](#_Toc42978440)

[12.2. Проектирование 38](#_Toc42978441)

[12.3. Тестирование 39](#_Toc42978442)

[12.4. Сравнение алгоритмов сортировки 40](#_Toc42978443)

[Заключение 41](#_Toc42978444)

[Библиографический список 42](#_Toc42978445)

[Приложение A. К задаче 1 43](#_Toc42978446)

[Приложение B. К задаче 2 45](#_Toc42978447)

[Приложение C. К задаче 3 47](#_Toc42978448)

[Приложение D. К задаче 4 48](#_Toc42978449)

[Приложение E. К задаче 5 50](#_Toc42978450)

[Приложение F. К задаче 6 54](#_Toc42978451)

[Приложение G. К задаче 7 57](#_Toc42978452)

[Приложение H. К задаче 8 62](#_Toc42978453)

[Приложение I. К задаче 9 67](#_Toc42978454)

[Приложение J. К задаче 10 75](#_Toc42978455)

[Приложение K. К задаче 11 80](#_Toc42978456)

[Приложение L. К задаче 12 84](#_Toc42978457)

# Введение

Основная задача данной работы состоит в развитии практических навыков создания программных систем при помощи современных средств разработки, таких как MS Visual Studio, а также получения необходимого опыта проектирования, реализации и тестирования различных программных систем.

Цель данной работы заключается в закреплении знаний и навыков, полученных по дисциплинам «Дискретная математика», «Компьютерный практикум по основам алгоритмизации и методам программирования», «Линейная алгебра», «Программирование», «Введение в программную инженерию», а также приобретение опыта алгоритмизации задач.

Для достижения поставленной цели необходимо:

1. Наличие практических навыков работы с языком программирования высокого уровня C# и современных сред разработки для реализации предложенных программ.
2. Умение работать с системой контроля версий Git посредством использования веб-сервиса GitHub.
3. Провести тестирование всех разработанных систем по критериям черного ящика.
4. Развитие практических навыков объектно-ориентированного программирования.

Основная часть данной работы содержит результаты выполнения 12 заданий учебной практики и включает в себя: постановку задачи, формат входных и выходных данных, описание решения задачи, разработка функциональных и нефункциональных требований, разработку алгоритмов, процесса реализации и результатов тестирования программы.

# Задача 1

Вы являетесь одним из разработчиков нового архитектурного пакета прикладных программ «CadArch». Одной из его функций является проектирование укладки половых плиток. В настоящее время вы занимаетесь программной реализацией модуля, который отвечает за укладку плиток в прямоугольных помещениях. Для простоты будем считать, что пол помещения представляет собой прямоугольник размером n на m метров, разбитый на m∙n квадратиков со стороной по 1 метру. Кроме этого, будем считать, что имеется четыре типа плиток, показанные в таблице. Каждая из плиток представляет собой квадрат размером 2 на 2 метра, из которого вырезан один квадратик размером 1 на 1 метр. Проектируемый модуль должен работать следующим образом. На вход модуля подается набор команд, каждая из которых обозначает, в какое место и какого типа плитку необходимо положить. Команда обрабатывается следующим образом: если ни один из квадратиков, который должна занимать текущая плитка, незанят и плитка полностью помещается внутри прямоугольника, то плитка размещается в указанном месте, в противном случае – нет. Требуется написать программу, которая определяет, какая площадь в соответствии с заданным набором команд будет покрыта плитками. [**ДОБАВИТЬ КАРТИНКУ ЗАДАЧИ**](https://acmp.ru/asp/article/image.asp?id=255)

Входной файл: input.txt

Выходной файл: output.txt

Программа должна выполняться не более чем за 1 секунду и требовать не более чем 16 МБ памяти.

Первая строка входного файла INPUT.TXT содержит два числа n и m — длина и ширина пола помещения (1 ≤ m, n ≤ 50). Вторая строка содержит число k — количество команд, которые необходимо обработать. Каждая из последующих k строк содержит описание одной команды из набора команд. Описание команды состоит из трех чисел. Первое число определяет тип плитки (число от 1 до 4), а два других - координаты левого верхнего угла квадрата (y, x) размером 2 на 2, в который вписана соответствующая плитка (0 ≤ x, y, k ≤ 1000).

В выходной файл OUTPUT.TXT необходимо вывести одно число, определяющее площадь, покрытую плитками после выполнения заданной во входном файле последовательности команд

## Анализ

Для решения задачи надо:

1) Считать данные

2) Определить тип плитки и ее позицию

3) Проверить ее не пересекает ли она другие клетки

4) Вывести ответ

Сначала определяется размер поля и количество плиток, потом создается матрица заданного размера MxN. Потом заданное количество раз считывается тип плитки и ее координаты. Потом проверяется помещается ли плитка в поле, а затем свободны ли клетки плитки, на которые она кладется. Если да, то ячейки матрицы, где лежит плитка заполняются 1, иначе ничего не происходит. Потом по массиву проверяется количество заполненных клеток.

## Проектирование

Согласно проведенному анализу и приведенному там алгоритму была составлена блок-схема для решения задачи (см. рис. 1.1).

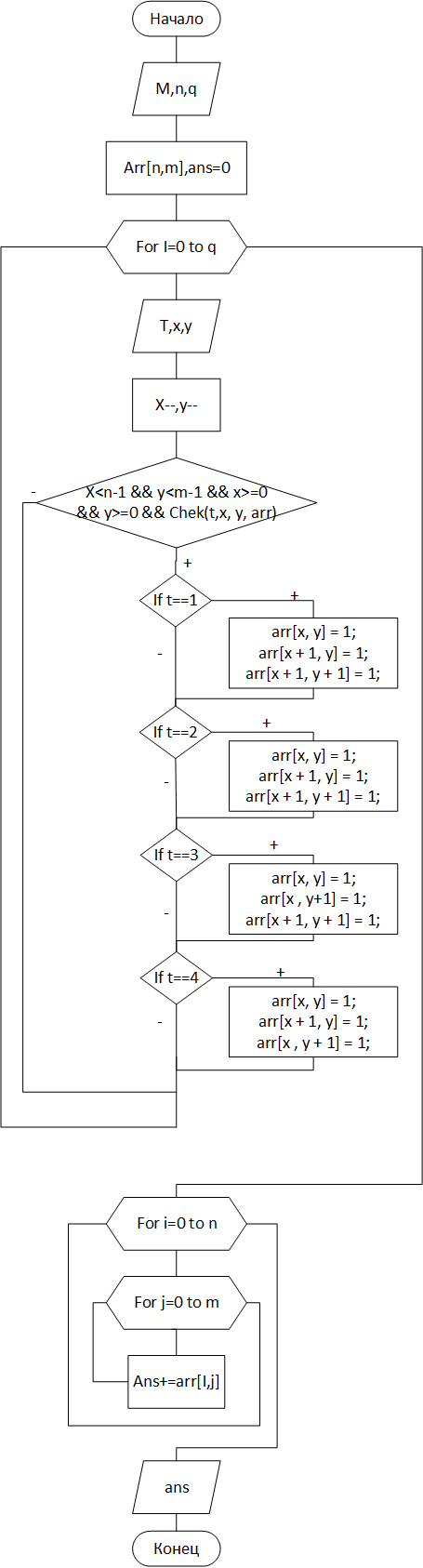


Рисунок 1.1. Блок-схема для задачи 1

## Тестирование

Результаты тестирования задачи на сайте acmp.ru.

Требования к скорости работы и занимаемой памяти также были выполнены.

# Задача 2

Петя написал программу движения робота К-79. Программа состоит из следующих команд:

* S — сделать шаг вперед
* L — повернуться на 90 градусов влево
* R — повернуться на 90 градусов вправо

Напишите программу, которая по заданной программе для робота определит, сколько шагов он сделает прежде, чем впервые вернется на то место, на котором уже побывал до этого, либо установит, что этого не произойдет.

Входные данные: во входном файле INPUT.TXT записана одна строка из заглавных английских букв S, L, R, описывающая программу для робота. Общее число команд в программе от 1 до 200, при этом команд S — не более 50.

Выходные данные: в выходной файл OUTPUT.TXT выведите, сколько шагов будет сделано (то есть выполнено команд S) прежде, чем робот впервые окажется в том месте, через которое он уже проходил. Если такого не произойдет, выведите в выходной файл число –1.

Программа должна выполняться не более чем за 1 секунду и требовать не более чем 16 МБ памяти.

## Анализ

Для решения задачи надо:

1) Создать массив с клетками

2) Считать очередность команд

3) Выполнить очередность команд

4) Определить момент, когда робот наступает на клетку, которую уже посетил

Считывается очередность команд. Потом создается массив и дополнитльные переменные, которые означают текущие координаты. Потом выполняются все в команды и если после выполенения команды, была посещена новая клетка, то она заполняется 1, а если робот наступил клетку, которая уже 1, то выводится результат.

## Проектирование

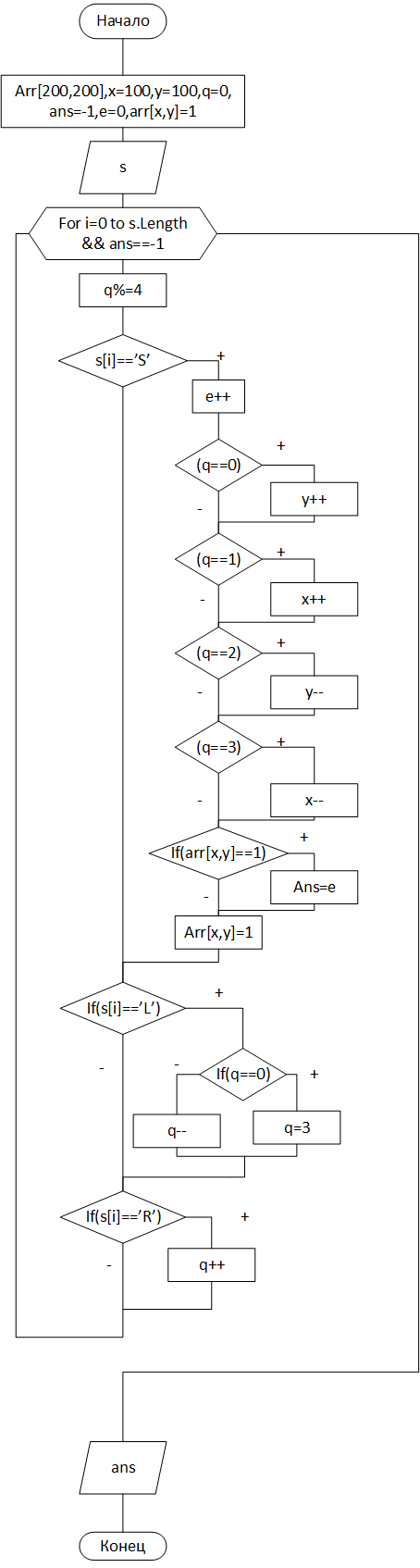
Согласно проведенному анализу и приведенному там алгоритму была составлена блок-схема для решения задачи (см. рис. 2.1).

Рисунок 2.1. Блок-схема для задачи 2

## Тестирование

Результаты тестирования задачи на сайте acmp.ru.



Требования к скорости работы и занимаемой памяти также были выполнены.

# Задача 3

Пусть D-заштрихованная часть плоскости и пусть u определяется по x и y следующим образом (запись означает, что точка с координатами x, y принадлежит D):

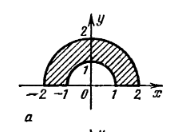


Рисунок 3.1. График к задаче 3

## Анализ

Для решения задачи надо:

1) Считать два числа

2) Проверить попадают ли они в заданную область

Надо считать число, потом определить попадает ли оно в область, если да, то вывести 0, иначе координату по оси x.

Также необходимо производить проверку ввода на правильность ввода действительного числа.

## Проектирование

Согласно проведенному анализу, была составлена блок-схема для решения задачи (см. рис. 3.2).

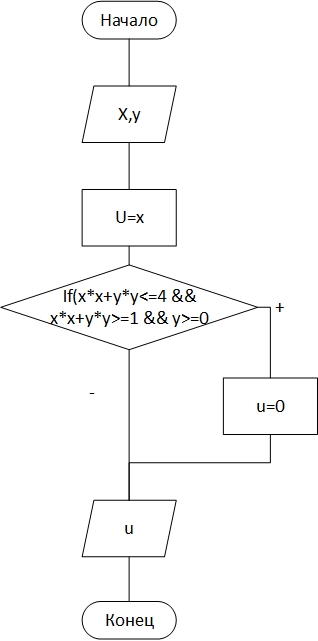


Рисунок 3.2. Блок-схема для задачи 3

## Тестирование

.

# Задача 4

Дано действительное положительное число ε. Методом деления отрезка пополам найти приближенное значение корня уравнения Абсолютная погрешность найденного значения не должна превосходить ε.

## Анализ

Для решения данной задачи считывается действительное число, затем методом деления отрезка пополам в интервале от 0 до 2, ищется X, при котором f(x) будет = 0 с заданной точностью. Также необходимо проверить корректность ввода.

## Проектирование

Согласно проведенному анализу, была составлена блок-схема для решения задачи (см. рис. 4.1).

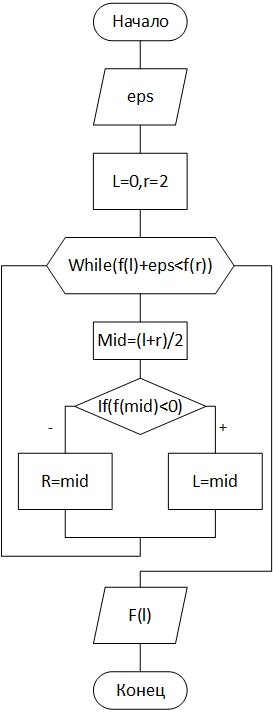


Рисунок 4.1. Блок-схема для задачи 4

## Тестирование

Для тестирования была составлена следующая таблица тестов (табл. 4.1). Она содержит входные данные, ожидаемые и реальные результаты работы программы с такими данными:

Таблица 4.1. Таблица тестов для задачи 4

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | j | Ошибка ввода | Ошибка ввода |
| Т2 | 7 о | Ошибка ввода | Ошибка ввода |
| Т3 | -4 | Ошибка ввода | Ошибка ввода |
| Т4 | 9 - 4 | Ошибка ввода | Ошибка ввода |
| Т5 | 3 5,4 | Ошибка ввода | Ошибка ввода |
| Т6 | 3 1 | Ошибка ввода | Ошибка ввода |
| Т7 | 34 2 | 0 1 0 0 0 1 | 0 1 0 0 0 1 |

Данные тесты покрывают критерии черного ящика, представленные в таблице D.1 в приложении D.

# Задача 5

Даны натуральное число , действительная квадратная матрица порядка n. Построить последовательность из нулей и единиц, в которой тогда и только тогда, когда элементы i-й строки матрицы образуют возрастающую или убывающую последовательность.

## Анализ

Считывается число N и массив целых чисел размера NxN. Потом для каждой строки определяется это последовательность убывающая, возрастающая или другая. Потом выводится ответ.

## Проектирование

В процессе проектирования было принято решение проектировать метод определения координат следующей ячейки как отдельную функцию (см. рис. E.1 в приложении E). Блок-схема остальной программы представлена на рисунке 5.2:

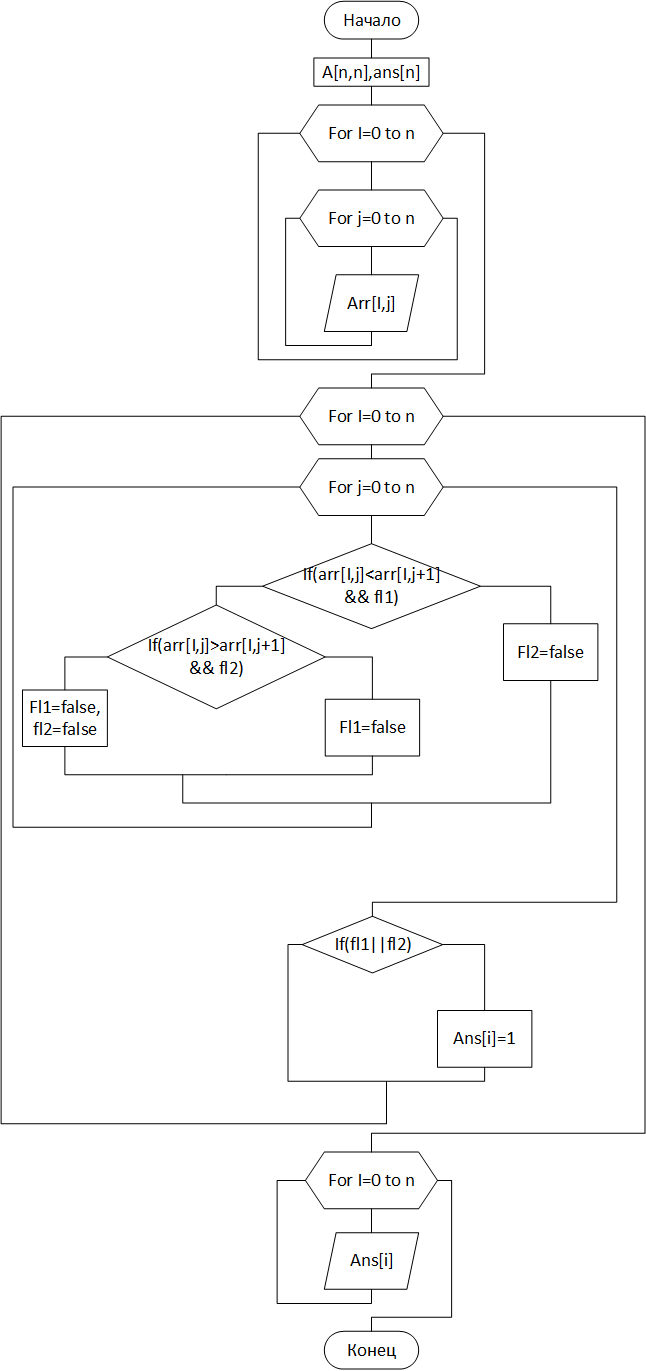


Рисунок 5.1. Блок-схема алгоритма для задачи 5

## Тестирование

Для тестирования была составлена следующая таблица тестов (табл. 5.1). Она содержит входные данные, ожидаемые и реальные результаты работы программы с такими данными:

Таблица 5.1. Таблица тестов для задачи 5

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | 0 0 0 0 0 0 0 0 | Ошибка ввода | Ошибка ввода |
| Т2 | 0 0 0 0 | Ошибка ввода | Ошибка ввода |
| Т3 | 9 5 7 р 9 4 9 а 6 7 8 3 2 4 9 5 7 р 9 4 9 а 6 7 8 3 2 4 9 5 7 р 9 4 9 а 6 7 8 3 2 4 0 0 0 0 0 0 0 | Ошибка ввода | Ошибка ввода |
| Т4 | -1 0,5 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 | -1 0,5 1 2 3 4 5 12 19 26 33 40 47 46 45 44 43 42 41 34 27 20 13 6 7 8 9 10 11 18 25 32 39 38 37 36 35 28 21 14 15 16 17 24 31 30 29 22 23 | -1 0,5 1 2 3 4 5 12 19 26 33 40 47 46 45 44 43 42 41 34 27 20 13 6 7 8 9 10 11 18 25 32 39 38 37 36 35 28 21 14 15 16 17 24 31 30 29 22 23 |

Данные тесты покрывают критерии черного ящика, представленные в таблице E.1 в приложении E.

# Задача 6

Ввести . Построить последовательность чисел . Довести ее до . Вывести последовательность. Сообщить, выполняется ли равенство . Сравнить J и N.

## Анализ

Так как последовательность задана рекуррентной формулой, то вычислять члены последовательности следует с помощью рекурсивной функции. Данная функция должна принимать три предыдущих члена последовательности число M и переменную для подсчета длины последовательности N. На основе введенных членов последовательности вычисляется следующий член ак. Если он больше M, то следует вернуть N, а также вывести сообщение о том, что аN не равно М. Если же ак меньше или равен M, то следует вывести данный член последовательности в строку вывода последовательности и увеличить N на единицу. Если при этом ак равен M, то следует вывести сообщение о том, что аN равно М и вернуть N, иначе следует вернуть значение данной функции, в которую подали в качестве аргументов ak, ak­1, ak‑2, M и N в соответственном порядке.

## Проектирование

Согласно проведенному анализу метод для вычисления членов последовательности был спроектирован как отдельная функция (см. рис. 6.1)

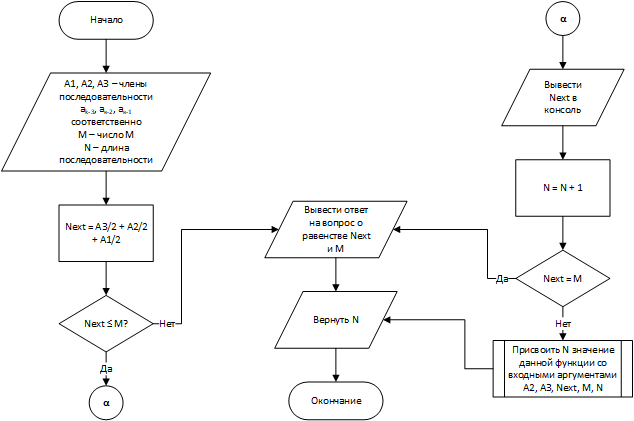


Рисунок 6.1. Блок-схема алгоритма расчета последовательности по рекуррентной формуле

Блок-схема алгоритма программы представляет значительно меньший интерес, поэтому была вынесена в приложение F (см. рис. F.1)

## Тестирование

Для тестирования была составлена следующая таблица тестов (см. табл. 6.1 и 6.2). Она содержит входные данные, ожидаемые и реальные результаты работы программы с такими данными:

Таблица 6.1. Таблица тестов для задачи 6

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | y | Ошибка ввода | Ошибка ввода |
| Т2 | 6 y | Ошибка ввода | Ошибка ввода |
| Т3 | 6 -8 h | Ошибка ввода | Ошибка ввода |
| Т4 | 4 5,6 3 t | Ошибка ввода | Ошибка ввода |
| Т5 | 0 0 0 8 | Последовательность бесконечна, состоит из нулей | Последовательность бесконечна, состоит из нулей |
| Т6 | 9 6 3 7 | Последовательность пуста | Последовательность пуста |
| Т7 | 3 8 5 7 | Последовательность: 3 Длина: 1 Равен M: False | Последовательность: 3 Длина: 1 Равен M: False |

Таблица 6.2. Продолжение таблицы тестов для задачи 6

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т8 | 6 4 9 6 | Последовательность: 6 4 Длина: 2 Равен M: True | Последовательность: 6 4 Длина: 2 Равен M: True |
| Т9 | 5 6 7 9 | Последовательность: 5 6 7 9 Длина: 4 Равен M: True | Последовательность: 5 6 7 9 Длина: 4 Равен M: True |

Данные тесты покрывают критерии черного ящика, представленные в таблице F.1 в приложении F.

# Задача 7

Заданы частоты символов входного алфавита. Построить двоичный суффиксный код Хаффмана. Кодовые слова выписать в лексикографическом порядке.

## Анализ

Сначала пользователь вводит количество символов, а затем длины этих символов.

Суффиксный двоичный код – это кодирующий алфавит, слова которого состоят из 0 и 1, и никакое слово не является суффиксом (окончанием) никакого другого слова в данном алфавите. Из этого следует, что составлять каждое слово следует с конца.

РАСПИСАТЬ КОД ХАФФМАНА + БЛОК-СХЕМА

## Проектирование

Согласно проведенному анализу метод для создания кодирующего суффиксного алфавита был спроектирован как отдельная функция (см. Приложение G, рис. G.1). Для определения добавляемой в начало слова цифры здесь используется остаток от деления количества слов единичной на данном этапе длины. Если же таких слов не было, то для одной половины используется 0, а для другой 1.

Метод сортировки алфавита в лексикографическом порядке также был спроектирован отдельной функцией (см. Приложение G, рис. G.2). Он основан на сортировке пузырьком, но с другими условиями сортировки.

Блок-схема основной функции программы представлена на рисунке 7.1:

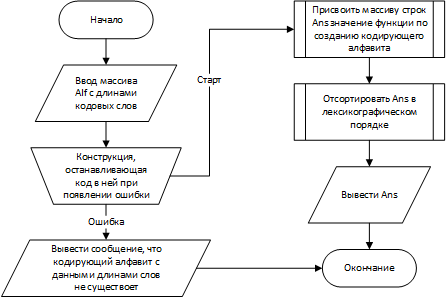


Рисунок 7.1. Блок-схема алгоритма для задачи 7

## Тестирование

Для тестирования была составлена следующая таблица тестов (см. табл. 7.1).:

Таблица 7.1. Таблица тестов для задачи 7

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | y | Ошибка ввода | Ошибка ввода |
| Т2 | 9,8 | Ошибка ввода | Ошибка ввода |
| Т3 | 6 h | Ошибка ввода | Ошибка ввода |
| Т4 | 4 5,6 | Ошибка ввода | Ошибка ввода |
| Т5 | 3 1 1 1 | Ошибка ввода | Ошибка ввода |

Таблица 7.2. Продолжение таблицы тестов для задачи 7

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т6 | 4 2 2 2 2 | 00 01 10 11 | 00 01 10 11 |
| Т7 | 5 3 3 4 6 8 | 000 00000100 000011 001 0010 | 000 00000100 000011 001 0010 |

В процессе тестирования была обнаруженная и исправлена ошибка с неправильной проверкой ввода чисел: проверку проходили все целые числа, а не только натуральные.

Данные тесты покрывают критерии черного ящика, представленные в таблице G.1 в приложении G.

# Задача 8

Граф задан матрицей смежности. Найти в нем какую-либо простую цепь из K вершин.

## Анализ

Сначала пользователь задает размер массива и длину цепи, затем матрицу смежности.

После этого необходимо создать рекурсивную функцию, которая будет проверять возможность пройти по какому-либо ребру графа, если это возможно, проходить и вызываться для следующей точки. Получается, что вызывается обход в глубину для каждой вершины. Затем восстанавливается путь, который получился в результате обхода графа и проверяется его цикличность.

## Проектирование

## Тестирование

Для тестирования была составлена следующая таблица тестов (см. табл. 8.2). Она содержит входные данные, ожидаемые и реальные результаты работы программы с такими данными:

Таблица 8.2. Таблица тестов для задачи 8

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | 5 6 п 8 7 5 | Ошибка ввода | Ошибка ввода |
| Т2 | 4 9 6  0 1 9 | Ошибка ввода | Ошибка ввода |
| Т3 | 0 1 0 0 1 0 0 1 0 0 0 0 1 1 0 0 0 0 | Ошибка ввода | Ошибка ввода |
| Т4 | 0 1 0 1 1 0 0 1 0 1 1 1 1 0 0 1 0 0 1 0 | Ошибка ввода | Ошибка ввода |
| Т5 | 1 1 1 1 0 0 0 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0 1 1 | 1 => 2 => 3 => 4 => 1 => 3 => 1 | 1 => 2 => 3 => 4 => 1 => 3 => 1 |

Данные тесты покрывают критерии черного ящика, представленные в таблице H.1 в приложении H.

# Задача 9

Напишите метод создания линейного списка, в информационные поля элементов которого последовательно заносятся номера с 1 до N (N водится с клавиатуры). Первый включенный в список элемент, имеющий номер 1, оказывается в голове списка (первым). Разработайте методы поиска и удаления элементов списка

## Анализ

Надо создать линейный однонаправленный список, для этого создан класс, в котором будем заполняться последовательность элементов со ссылкой на следующий. В этом классе будут создан конструктор, который буде принимать целое число, а также будут реализованы методы для добавления, удаления и поиска элементов.

## Проектирование

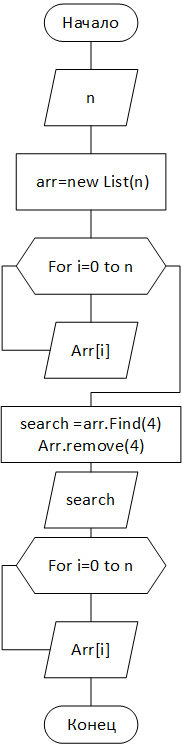
Блок-схема для основного тела программы представлена на рисунке 9.1:

Рисунок 9.1. Блок-схема алгоритма для задачи 9

## Тестирование

Для тестирования была составлена следующая таблица тестов (см. табл. 9.1). Она содержит входные данные, ожидаемые и реальные результаты работы программы с такими данными:

Таблица 9.1. Таблица тестов для задачи 9

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | y | Ошибка ввода | Ошибка ввода |
| Т2 | 9,8 | Ошибка ввода | Ошибка ввода |
| Т3 | 9 | Полученный список: 1 2 3 4 5 6 7 8 9 Полученный список: 1 2 3 4 5 6 7 8 Поиск 6 элемента: 6 | Полученный список: 1 2 3 4 5 6 7 8 9 Полученный список: 1 2 3 4 5 6 7 8 Поиск 6 элемента: 6 |

Данные тесты покрывают критерии черного ящика, представленные в таблице I.5 в приложении I.

# Задача 10

Даны натуральное число , действительные числа Получить последовательность . Для решения этой задачи полезен однонаправленный список.

## Анализ

Надо создать линейный однонаправленный список, для этого создан класс, в котором будем заполняться последовательность элементов со ссылкой на следующий. В этом классе будут создан конструктор, который буде принимать массив действительных чисел. Потом из каждого элемента будет вычитаться последний.

## Проектирование

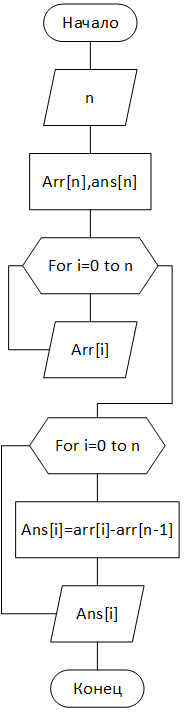
Блок-схема для основного тела программы представлена на рисунке 10.1:

Рисунок 10.1. Блок-схема алгоритма для задачи 10

## Тестирование

Для тестирования была составлена следующая таблица тестов (см. табл. 10.1). Она содержит входные данные, ожидаемые и реальные результаты работы программы с такими данными:

Таблица 10.1. Таблица тестов для задачи 10

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | input1.txt: пуст input2.txt: пуст | output.txt: пуст | output.txt: пуст |
| Т2 | input1.txt:  6 7 input2.txt: пуст | output.txt: пуст | output.txt: пуст |
| Т3 | input1.txt:  6 7 2 7 input2.txt:  1 8 | output.txt:  7 56 3 56 | output.txt:  7 56 3 56 |
| Т4 | input1.txt:  6 y input2.txt:  1 9 | Ошибка ввода | Ошибка ввода |
| Т5 | input1.txt:  6 0 2 0 66 0 input2.txt:  1 10 3 2 | output.txt: пуст | output.txt: пуст |

Данные тесты покрывают критерии черного ящика, представленные в таблице J.3 в приложении J.

# Задача 11

Следующий способ предназначен для шифровки последовательностей нулей и единиц (или же, например, точек и тире). Пусть —такая последовательность. То, что предлагается в качестве ее шифра, — это последо­вательность , образованная по следующему закону:

Пользуясь изложенным способом:

а) зашифровать данную последовательность;

б) расшифровать данную последовательность

## Анализ

Сначала считывается строка, состоящая из 0 и 1. Затем она шифруется в соответствии с заданной системой. Потом она дешифруется обратным способом.

## Проектирование

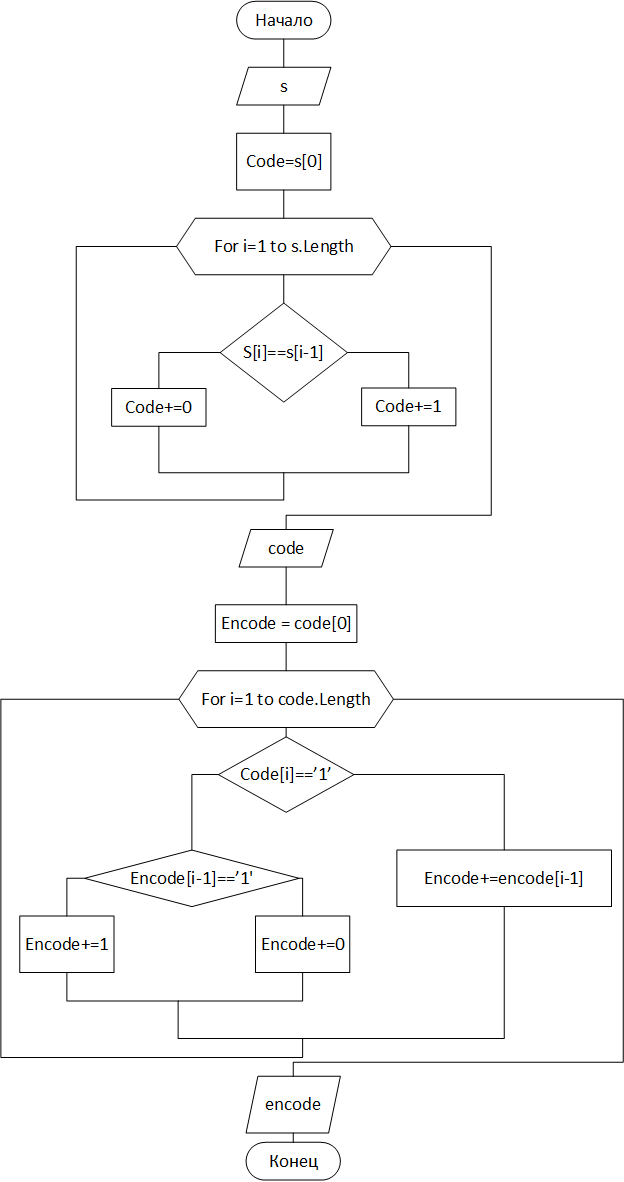
Согласно проведенному анализу, была составлена блок-схема для решения задачи (см. рис. 3.2).

Рисунок 11.1. Блок-схема алгоритма

## Тестирование

Для тестирования была составлена следующая таблица тестов (см. табл. 11.1, 11.2). Она содержит входные данные, ожидаемые и реальные результаты работы программы с такими данными:

Таблица 11.1. Таблица тестов для задачи 11

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | аывп | Ошибка ввода | Ошибка ввода |
| Т2 | 01234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789 | Ошибка ввода | Ошибка ввода |
| Т3 | 0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789 1 0 0 1 | Ошибка ввода | Ошибка ввода |
| Т4 | 0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789 0 1 1 1 0 1 1 0 1 1 1 1 0 1 1 2 1 1 0 1 1 0 1 0 1 1 1 0 1 1 0 1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 0 1 1 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 | Ошибка ввода | Ошибка ввода |
| Т5 | 0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789 h 1 1 1 0 1 1 0 1 1 1 1 0 1 1 0 1 1 0 1 1 0 1 0 1 1 1 0 1 1 0 1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 0 1 1 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 | Ошибка ввода | Ошибка ввода |

Таблица 11.2. Продолжение таблицы тестов для задачи 11

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т6 | 0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789 0 1 1 1 0 1 1 0 1 1 1 1 0 1 1 0 1 1 0 1 1 0 1 0 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 0 1 1 0 1 0 1 1 0 1 1 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 | Ошибка ввода | Ошибка ввода |
| Т7 | 0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789 0 1 1 1 0 1 1 0 1 1 1 1 0 1 1 0 1 1 0 1 1 0 1 0 1 1 1 0 1 1 0 1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 0 1 1 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 | Зашифрованная строка: 0756124295403304415276379718809968103281792538645665468352971823018699088179736725144033045924216570  Расшифрованная строка: 0472581370592471396803069999888777666555444332211195274186294075286031969300001112223334445556677888 | Зашифрованная строка: 0756124295403304415276379718809968103281792538645665468352971823018699088179736725144033045924216570  Расшифрованная строка: 0472581370592471396803069999888777666555444332211195274186294075286031969300001112223334445556677888 |

Данные тесты покрывают критерии черного ящика, представленные в таблице K.1 в приложении K.

# Задача 12

Выполнить сравнение двух предложенных методов сортировки одномерных массивов, содержащих n элементов, по количеству пересылок и сравнений.

Для этого необходимо выполнить программную реализацию двух методов сортировки, включив в нее подсчет количества пересылок (т.е. перемещений элементов с одного места на другое) и сравнений.

Провести анализ методов сортировки для трех массивов: упорядоченного по возрастанию, упорядоченного по убыванию и неупорядоченного.

Все три массива следует отсортировать обоими методами сортировки.

Найти в литературе теоретические оценки сложности каждого из методов и сравнить их с оценками, полученными на практике.

Сделать выводы о том, насколько отличаются теоретические и практические оценки количества операций, объяснить почему это происходит. Сравнить оценки сложности двух алгоритмов. Предложенные методы сортировки:

1. Сортировка перемешиванием.
2. Пирамидальная сортировка.

## Анализ

Сортировка пузырьком предполагает проход по массиву со сравниванием рядом стоящих элементов. Если они стоят в неправильном порядке – их следует поменять местами. После такого прохода отсортируется один элемент – самый легкий или самый тяжелый, в зависимости от направления прохода. Далее повторять данный процесс, но каждый раз уменьшать длину прохода на отсортированный элемент, пока массив не будет полностью состоять из отсортированных элементов.

Эта версия сортировки без улучшений и упрощений, но так как необходимо оценить сложность алгоритма на практике, то следует оставить такую версию.

Теоретическое количество пересылок в лучшем случае равно 0, в худшем случае – ((N - 1) \* N / 2). Количество сравнений всегда равно (N – 1)\*N/2

Сортировка слиянием предполагает разделение массива на более мелкие части и их сортировка, а после – слияние обратно в единый массив. В реализации это выглядит как деление массива пополам и вызов данной функции для половинок массива, если они длиннее одного элемента, и дальнейшее слияние данных массивов в правильном порядке.

Теоретическое количество сравнений равно количеству перестановок, не зависит от случая и вычисляется по рекуррентной формуле C(1)=0, C(n) = C(n/2) + n.

## Проектирование

Блок-схемы алгоритмов сортировки пузырьком и сортировки слиянием представлены на рисунках 12.1 и 12.2 соответственно.

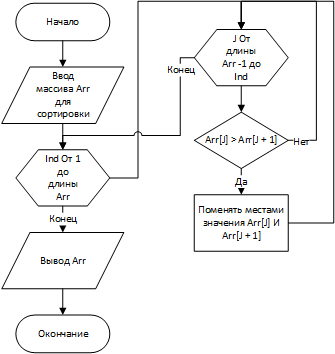


Рисунок 12.1. Блок-схема алгоритма сортировки пузырьком

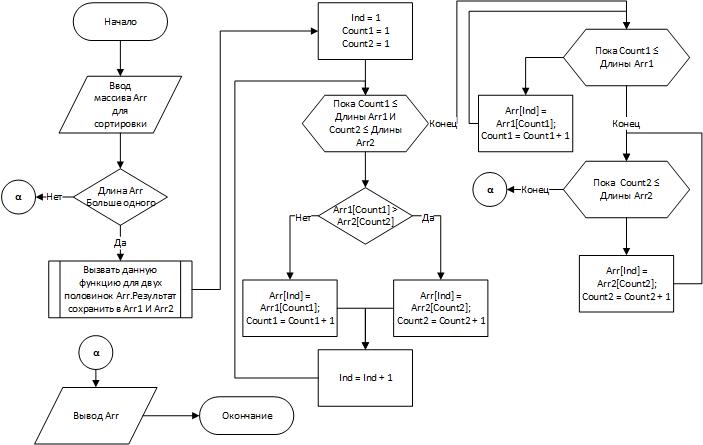


Рисунок 12.2. Блок-схема алгоритма сортировки слиянием

Для подсчета количества сравнений и перестановок необходимо добавить переменные-счетчики и увеличивать из на один после каждого сравнения или перестановки соответственно.

## Тестирование

Для тестирования была составлена следующая таблица тестов (см. табл. 11.1, 11.2). Она содержит входные данные, ожидаемые и реальные результаты работы программы с такими данными:

Таблица 12.1. Таблица тестов для задачи 12

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемые результаты** | **Реальные результаты** |
|
| Т1 | 3 5 а | Ошибка ввода | Ошибка ввода |
| Т2 | 1 5 6 8 9 34 56 78 | 1 5 6 8 9 34 56 78 1 5 6 8 9 34 56 78 | 1 5 6 8 9 34 56 78 1 5 6 8 9 34 56 78 |
| Т3 | 99 65 43 21 7 5 3 2 1 | 1 2 3 5 7 21 43 65 99 1 2 3 5 7 21 43 65 99 | 1 2 3 5 7 21 43 65 99 1 2 3 5 7 21 43 65 99 |
| Т4 | 8 3 99 5 73 5 79 1 31 | 1 3 5 5 8 31 73 79 99 1 3 5 5 8 31 73 79 99 | 1 3 5 5 8 31 73 79 99 1 3 5 5 8 31 73 79 99 |
| Т5 |  | Массив пуст | Массив пуст |
| Т6 | 6 | 6 6 | 6 6 |

Данные тесты покрывают критерии черного ящика, представленные в таблице L.1 в приложении L.

## Сравнение алгоритмов сортировки

Практические данные по количеству пересылок и количеству сравнений при сортировке представлены в таблице 12.2:

Таблица 12.2. Сравнение двух сортировок

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Число сравнений** | | **Число пересылок** | |
| **Пузырьком** | **Слиянием** | **Пузырьком** | **Слиянием** |
| Упорядоченный по возрастанию массив | 45 | 34 | 0 | 34 |
| Упорядоченный по убыванию массив | 45 | 34 | 45 | 34 |
| Неупорядоченный массив | 45 | 34 | 16 | 34 |

Теоретические данные возможно найти для худших и лучших результатов, и для обеих сортировок они совпадают с реальными результатами.

Неупорядоченный массив при сортировке пузырьком имеет 45 сравнений, как и любой иной случай сортировки массива из 10 элементов, так как при данной сортировке количество сравнений является константой при определенном N. Число пересылок отличается от худшего и лучшего случая, но находится между ними, так как лучший случай указывает на минимальное количество пересылок, а худший – на максимальное.

При сортировке слиянием количество пересылок и сравнений является константой при определенном N, поэтому в случае неупорядоченного массива их число не изменилось.

При сравнении двух сортировок заметно, что даже на таком маленьком массиве число сравнений при сортировке слиянием заметно меньше, чем при сортировке пузырьком. Число пересылок зависит от случая, так как сортировка слиянием не имеет худшего случая, в отличии от сортировки пузырьком. В худшем случае пузырьковой сортировки производится больше пересылок, чем при сортировке слиянием.

Теоретические данные показывают, что чем более крупный набор данных необходимо отсортировать, тем заметнее будет превосходство сортировки слиянием над сортировкой пузырьком.

# Заключение

Подводя итоги, были спроектированы, разработаны и протестированы двенадцать задач; развиты и закреплены практические навыки построения и описания алгоритмов для решения задач из разных предметных областей (численные методы, дискретная математика, структуры данных и др). Были развиты и закреплены практические навыки использования языков высокого уровня и современных сред разработки для реализации построенных алгоритмов. Были развиты и закреплены практические навыки объектно-ориентированного программирования. Были развиты практические навыки оформления отчетов о проделанной работе и публичного выступления с защитой проекта.

По окончании работы получены приложения выполняющие поставленные задачи. Таким образом, все задачи были выполнены, и как следствие – достигнута цель.

# Библиографический список

1. Абрамов С.А. Задачи по программированию // С.А. Абрамов, Г.Г. Гнездилова, Е.Н. Капустина, М.И. Селюн — М.: Наука, 1988. — 578 c.
2. Плаксин М. А. Тестирование и отладка программ для профессионалов будущих и настоящих / М. Плаксин - М.: Бином, 2013. – 20-45 с.
3. Терехов А.Н. Технология программирования: учебное пособие. / А.Н. Терехов: – 2-е изд. – М.: БИНОМ. Лаборатория знаний, 2012. — 130 с.
4. Троелсен Э. Язык программирования C# 2010 и платформа .NET 4 // пер. с англ. – М.: Издательский дом "Вильямс", 2012. — 1412 c.

# К задаче 1

Листинг программы:

using System;

namespace Task1

{

internal class Program

{

static bool Chek(int q,int x, int y, int[,] arr)

{

bool fl;

if (q == 1)

{

return arr[x + 1, y] == 0 && arr[x, y + 1] == 0 && arr[x+1, y+1] == 0;

}

if (q == 2)

{

return arr[x, y] == 0 && arr[x + 1, y] == 0 && arr[x+1, y+1] == 0;

}

if (q == 3)

{

return arr[x, y] == 0 && arr[x, y + 1] == 0 && arr[x+1, y+1] == 0;

}

return arr[x, y] == 0 && arr[x + 1, y] == 0 && arr[x, y + 1] == 0;

}

public static void Main(string[] args)

{

string[] str=Console.ReadLine().Split(' ');

int n=int.Parse(str[0]);

int m=int.Parse(str[1]);

int[,] arr = new int[n,m];

int q = int.Parse(Console.ReadLine());

for (int i = 0; i < q; ++i)

{

str=(Console.ReadLine().Split(' '));

int t = int.Parse(str[0]);

int x = int.Parse(str[1]);

int y = int.Parse(str[2]);

x--;y--;

if (x < n-1 && y < m-1 && x>=0 && y>=0)

switch (t)

{

case 1:

if (Chek(t,x,y,arr)){

arr[x+1, y] = 1;

arr[x, y+1] = 1;

arr[x+1, y+1] = 1;

}

break;

case 2:

if (Chek(t,x, y, arr))

{

arr[x, y] = 1;

arr[x + 1, y] = 1;

arr[x + 1, y + 1] = 1;

}

break;

case 3:

if (Chek(t,x, y, arr))

{

arr[x, y] = 1;

arr[x, y + 1] = 1;

arr[x + 1, y + 1] = 1;

}

break;

case 4:

if (Chek(t,x, y, arr))

{

arr[x, y] = 1;

arr[x + 1, y] = 1;

arr[x, y + 1] = 1;

}

break;

}

}

int ans = 0;

for (int i = 0; i < n; ++i)

{

for (int j = 0; j < m; ++j)

{

ans += arr[i, j];

}

}

Console.WriteLine(ans);

}

}

}

# К задаче 2

Листинг программы:

using System;

namespace Task1

{

internal class Program

{

static bool Chek(int q,int x, int y, int[,] arr)

{

bool fl;

if (q == 1)

{

return arr[x + 1, y] == 0 && arr[x, y + 1] == 0 && arr[x+1, y+1] == 0;

}

if (q == 2)

{

return arr[x, y] == 0 && arr[x + 1, y] == 0 && arr[x+1, y+1] == 0;

}

if (q == 3)

{

return arr[x, y] == 0 && arr[x, y + 1] == 0 && arr[x+1, y+1] == 0;

}

return arr[x, y] == 0 && arr[x + 1, y] == 0 && arr[x, y + 1] == 0;

}

public static void Main(string[] args)

{

string[] str=Console.ReadLine().Split(' ');

int n=int.Parse(str[0]);

int m=int.Parse(str[1]);

int[,] arr = new int[n,m];

int q = int.Parse(Console.ReadLine());

for (int i = 0; i < q; ++i)

{

str=(Console.ReadLine().Split(' '));

int t = int.Parse(str[0]);

int x = int.Parse(str[1]);

int y = int.Parse(str[2]);

x--;y--;

if (x < n-1 && y < m-1 && x>=0 && y>=0)

switch (t)

{

case 1:

if (Chek(t,x,y,arr)){

arr[x+1, y] = 1;

arr[x, y+1] = 1;

arr[x+1, y+1] = 1;

}

break;

case 2:

if (Chek(t,x, y, arr))

{

arr[x, y] = 1;

arr[x + 1, y] = 1;

arr[x + 1, y + 1] = 1;

}

break;

case 3:

if (Chek(t,x, y, arr))

{

arr[x, y] = 1;

arr[x, y + 1] = 1;

arr[x + 1, y + 1] = 1;

}

break;

case 4:

if (Chek(t,x, y, arr))

{

arr[x, y] = 1;

arr[x + 1, y] = 1;

arr[x, y + 1] = 1;

}

break;

}

}

int ans = 0;

for (int i = 0; i < n; ++i)

{

for (int j = 0; j < m; ++j)

{

ans += arr[i, j];

}

}

Console.WriteLine(ans);

}

}

}

# К задаче 3

Листинг программы:

using System;

namespace Task3

{

public class Program

{

public static int ReadInt(int left = -100, int right = 100)

{

bool ok = false;

int number = 0;

do

{

try

{

number = int.Parse(Console.ReadLine());

if (number >= left && number <= right) ok = true;

else

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введено не целое число. Введите целое число.");

ok = false;

}

catch (OverflowException)

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

} while (!ok);

return number;

}

public static int Solve(int x, int y)

{

int u = x;

if (x \* x + y \* y <= 4 && x \* x + y \* y >= 1 && y >= 0)

u = 0;

return u;

}

public static void Main(string[] args)

{

Console.WriteLine("Введите число x");

int x = ReadInt();

Console.WriteLine("Введите число y");

int y = ReadInt(), u = x;

if (x \* x + y \* y <= 4 && x \* x + y \* y >= 1 && y >= 0)

u = 0;

Console.WriteLine(u);

}

}

}

Листинг приложения WinForm Designed

namespace Task3

{

partial class Form1

{

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

/// <param name="disposing">истинно, если управляемый ресурс должен быть удален; иначе ложно.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Код, автоматически созданный конструктором форм Windows

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

private void InitializeComponent()

{

this.button = new System.Windows.Forms.Button();

this.x\_label = new System.Windows.Forms.Label();

this.xText = new System.Windows.Forms.TextBox();

this.yText = new System.Windows.Forms.TextBox();

this.y\_label = new System.Windows.Forms.Label();

this.label1 = new System.Windows.Forms.Label();

this.answer = new System.Windows.Forms.TextBox();

this.SuspendLayout();

//

// button

//

this.button.Location = new System.Drawing.Point(237, 226);

this.button.Name = "button";

this.button.Size = new System.Drawing.Size(75, 23);

this.button.TabIndex = 0;

this.button.Text = "Готово";

this.button.UseVisualStyleBackColor = true;

this.button.Click += new System.EventHandler(this.button\_Click);

//

// x\_label

//

this.x\_label.AutoSize = true;

this.x\_label.Location = new System.Drawing.Point(77, 131);

this.x\_label.Name = "x\_label";

this.x\_label.Size = new System.Drawing.Size(12, 13);

this.x\_label.TabIndex = 1;

this.x\_label.Text = "x";

//

// xText

//

this.xText.Location = new System.Drawing.Point(95, 131);

this.xText.Name = "xText";

this.xText.Size = new System.Drawing.Size(100, 20);

this.xText.TabIndex = 2;

this.xText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// yText

//

this.yText.Location = new System.Drawing.Point(95, 181);

this.yText.Name = "yText";

this.yText.Size = new System.Drawing.Size(100, 20);

this.yText.TabIndex = 4;

this.yText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// y\_label

//

this.y\_label.AutoSize = true;

this.y\_label.Location = new System.Drawing.Point(77, 184);

this.y\_label.Name = "y\_label";

this.y\_label.Size = new System.Drawing.Size(12, 13);

this.y\_label.TabIndex = 3;

this.y\_label.Text = "y";

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(272, 138);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(40, 13);

this.label1.TabIndex = 5;

this.label1.Text = "Вывод";

//

// answer

//

this.answer.Location = new System.Drawing.Point(318, 135);

this.answer.Name = "answer";

this.answer.ReadOnly = true;

this.answer.Size = new System.Drawing.Size(100, 20);

this.answer.TabIndex = 6;

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.answer);

this.Controls.Add(this.label1);

this.Controls.Add(this.yText);

this.Controls.Add(this.y\_label);

this.Controls.Add(this.xText);

this.Controls.Add(this.x\_label);

this.Controls.Add(this.button);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Button button;

private System.Windows.Forms.Label x\_label;

private System.Windows.Forms.TextBox xText;

private System.Windows.Forms.TextBox yText;

private System.Windows.Forms.Label y\_label;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox answer;

}

}

Листинг WinForm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task3

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void KeyPress(object sender, KeyPressEventArgs e)

{

if ((e.KeyChar <= 47 || e.KeyChar >= 58) && e.KeyChar != 8 && e.KeyChar != 46)

{

e.Handled = true;

}

}

private void button\_Click(object sender, EventArgs e)

{

double x, y;

if (double.TryParse(xText.Text,out x) && double.TryParse(xText.Text, out y) && xText.Text!="" && yText.Text!="")

{

answer.Text = Solve(x, y).ToString();

}

else

{

MessageBox.Show($"x={x\_label.Text} y={y\_label.Text}");

}

}

public static double Solve(double x, double y)

{

double u = x;

if (x \* x + y \* y <= 4 && x \* x + y \* y >= 1 && y >= 0)

u = 0;

return u;

}

}

}

# К задаче 4

Листинг программы:

using System;

namespace Task\_4

{

public class Program

{

public static double ReadD(int left = -100, int right = 100)

{

bool ok = false;

double number = 0;

do

{

try

{

number = double.Parse(Console.ReadLine());

if (number >= left && number <= right) ok = true;

else

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введено не целое число. Введите целое число.");

ok = false;

}

catch (OverflowException)

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

} while (!ok);

return number;

}

static double F(double x)

{

return x + Math.Log(x + 0.5) - 0.5;

}

public static double Solve(double eps)

{

double l = 0, r = 2;

while (F(l) + eps < F(r))

{

double mid = (l + r) / 2;

if (F(mid) < 0)

l = mid;

else r = mid;

}

return r;

}

public static void Main(string[] args)

{

double eps = ReadD();

var ans = Solve(eps);

Console.WriteLine(ans);

}

}

}

Листинг WinForm Designer

namespace Task4Form

{

partial class Form1

{

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

/// <param name="disposing">истинно, если управляемый ресурс должен быть удален; иначе ложно.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Код, автоматически созданный конструктором форм Windows

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

private void InitializeComponent()

{

this.button = new System.Windows.Forms.Button();

this.epsText = new System.Windows.Forms.TextBox();

this.label1 = new System.Windows.Forms.Label();

this.label2 = new System.Windows.Forms.Label();

this.answer = new System.Windows.Forms.TextBox();

this.SuspendLayout();

//

// button

//

this.button.Location = new System.Drawing.Point(214, 242);

this.button.Name = "button";

this.button.Size = new System.Drawing.Size(75, 23);

this.button.TabIndex = 0;

this.button.Text = "Готово";

this.button.UseVisualStyleBackColor = true;

this.button.Click += new System.EventHandler(this.button\_Click);

//

// epsText

//

this.epsText.Location = new System.Drawing.Point(135, 156);

this.epsText.Name = "epsText";

this.epsText.Size = new System.Drawing.Size(100, 20);

this.epsText.TabIndex = 1;

this.epsText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(89, 156);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(41, 13);

this.label1.TabIndex = 2;

this.label1.Text = "Epsilon";

//

// label2

//

this.label2.AutoSize = true;

this.label2.Location = new System.Drawing.Point(275, 156);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(42, 13);

this.label2.TabIndex = 4;

this.label2.Text = "Answer";

//

// answer

//

this.answer.Location = new System.Drawing.Point(323, 156);

this.answer.Name = "answer";

this.answer.ReadOnly = true;

this.answer.Size = new System.Drawing.Size(100, 20);

this.answer.TabIndex = 3;

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.label2);

this.Controls.Add(this.answer);

this.Controls.Add(this.label1);

this.Controls.Add(this.epsText);

this.Controls.Add(this.button);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Button button;

private System.Windows.Forms.TextBox epsText;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox answer;

}

}

Листинг WinForm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task4Form

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void KeyPress(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\-,\d]") && e.KeyChar != 8 && e.KeyChar != 46)

{

e.Handled = true;

}

}

private void button\_Click(object sender, EventArgs e)

{

double eps;

if (double.TryParse(epsText.Text, out eps) && epsText.Text != "")

{

answer.Text = Solve(eps).ToString();

}

else

{

MessageBox.Show($"Неправильные данные");

}

}

static double F(double x)

{

return x + Math.Log(x + 0.5) - 0.5;

}

public static double Solve(double eps)

{

double l = 0, r = 2;

while (F(l) + eps < F(r))

{

double mid = (l + r) / 2;

if (F(mid) < 0)

l = mid;

else r = mid;

}

return r;

}

}

}

# К задаче 5

Листинг программы:

using System;

using System.Collections.Concurrent;

namespace Task\_5

{

public class Program

{

public static int ReadInt(int left = -100, int right = 100)

{

bool ok = false;

int number = 0;

do

{

try

{

number = int.Parse(Console.ReadLine());

if (number >= left && number <= right) ok = true;

else

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введено не целое число. Введите целое число.");

ok = false;

}

catch (OverflowException)

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

} while (!ok);

return number;

}

public static int[] Solve(int[,] a,int n)

{

int[] ans = new int[n];

for (int i = 0; i < n; ++i)

{

bool fl1 = true, fl2 = true;

for (int j = 0; j < n - 1; ++j)

{

if (a[i, j] < a[i, j + 1] && fl1)

{

fl2 = false;

}

else if (a[i, j] > a[i, j + 1] && fl2)

{

fl1 = false;

}

else

{

fl1 = false;

fl2 = false;

}

}

if (fl1 || fl2) ans[i] = 1;

}

return ans;

}

public static void Main(string[] args)

{

Console.WriteLine("Введите размер матрицы");

int n = ReadInt(2);

int[,] a= new int[n,n];

for (int i = 0; i < n; ++i)

{

for (int j = 0; j < n; ++j)

{

Console.WriteLine($"Введите {i\*n+j+1} элемент матрицы");

a[i, j] = ReadInt();

}

}

var ans = Solve(a, n);

foreach (var tmp in ans)

{

Console.Write(tmp+" ");

}

}

}

}

Листинг WinForm Designer

namespace Task5

{

partial class Form1

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

this.label2 = new System.Windows.Forms.Label();

this.goButton = new System.Windows.Forms.Button();

this.label3 = new System.Windows.Forms.Label();

this.rndButton = new System.Windows.Forms.Button();

this.table = new System.Windows.Forms.TableLayoutPanel();

this.resultPanel = new System.Windows.Forms.TableLayoutPanel();

this.SuspendLayout();

//

// label2

//

this.label2.AutoSize = true;

this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label2.Location = new System.Drawing.Point(12, 54);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(72, 18);

this.label2.TabIndex = 2;

this.label2.Text = "Матрица:";

//

// goButton

//

this.goButton.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.goButton.Location = new System.Drawing.Point(592, 316);

this.goButton.Name = "goButton";

this.goButton.Size = new System.Drawing.Size(165, 27);

this.goButton.TabIndex = 4;

this.goButton.Text = "Готово";

this.goButton.UseVisualStyleBackColor = true;

this.goButton.Click += new System.EventHandler(this.goButton\_Click);

//

// label3

//

this.label3.AutoSize = true;

this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label3.Location = new System.Drawing.Point(488, 54);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(80, 18);

this.label3.TabIndex = 6;

this.label3.Text = "Результат";

//

// rndButton

//

this.rndButton.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.rndButton.Location = new System.Drawing.Point(592, 259);

this.rndButton.Name = "rndButton";

this.rndButton.Size = new System.Drawing.Size(165, 49);

this.rndButton.TabIndex = 7;

this.rndButton.Text = "Заполнить случайными числами";

this.rndButton.UseVisualStyleBackColor = true;

this.rndButton.Click += new System.EventHandler(this.rndButton\_Click);

//

// table

//

this.table.ColumnCount = 10;

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.Location = new System.Drawing.Point(15, 75);

this.table.Name = "table";

this.table.RowCount = 10;

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.table.Size = new System.Drawing.Size(470, 268);

this.table.TabIndex = 8;

//

// resultPanel

//

this.resultPanel.ColumnCount = 1;

this.resultPanel.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 100F));

this.resultPanel.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 20F));

this.resultPanel.Location = new System.Drawing.Point(492, 76);

this.resultPanel.Name = "resultPanel";

this.resultPanel.RowCount = 10;

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.resultPanel.Size = new System.Drawing.Size(76, 267);

this.resultPanel.TabIndex = 9;

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.BackColor = System.Drawing.SystemColors.Control;

this.ClientSize = new System.Drawing.Size(769, 349);

this.Controls.Add(this.resultPanel);

this.Controls.Add(this.table);

this.Controls.Add(this.rndButton);

this.Controls.Add(this.label3);

this.Controls.Add(this.goButton);

this.Controls.Add(this.label2);

this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;

this.Name = "Form1";

this.Load += new System.EventHandler(this.Task5Form\_Load);

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Label label2;

private System.Windows.Forms.Button goButton;

private System.Windows.Forms.Label label3;

private System.Windows.Forms.Button rndButton;

private System.Windows.Forms.TableLayoutPanel table;

private System.Windows.Forms.TableLayoutPanel resultPanel;

}

}

Листинг WinForm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text.RegularExpressions;

using System.Windows.Forms;

namespace Task5

{

public partial class Form1 : Form

{

private Random rand = new Random();

private static double[,] matr = new double[10, 10];

public Form1()

{

InitializeComponent();

}

private void Task5Form\_Load(object sender, EventArgs e)

{

for (int i = 0; i < 10; ++i)

{

for (int j = 0; j < 10; ++j)

{

TextBox text = new TextBox();

text.KeyPress += new KeyPressEventHandler(KeyPress);

table.Controls.Add(text, i, j);

}

}

for (int i = 0; i < 10; i++)

{

TextBox text = new TextBox();

text.ReadOnly = true;

resultPanel.Controls.Add(text, 0, i);

}

}

private void KeyPress(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\d,\-]") && e.KeyChar != 8 && e.KeyChar!=46)

e.Handled = true;

}

private void rndButton\_Click(object sender, EventArgs e)

{

for (int i = 0; i < table.Controls.Count; i++)

table.Controls[i].Text = rand.Next(100).ToString();

}

private bool HandleInputs()

{

bool ok = true;

string error = "";

for (int i = 0; i < table.Controls.Count && ok; i++)

{

ok = double.TryParse(table.Controls[i].Text, out matr[i % 10, i / 10]);

if (!ok)

error = $"Данные в строке {i / 10 + 1}, столбце {i % 10 + 1} введены некорректно";

}

if (!ok)

MessageBox.Show(error, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

return ok;

}

public static int[] Solve(double[,] matr)

{

int[] ans = new int[matr.GetLength(0)];

for (int i = 0; i < matr.GetLength(0); ++i)

{

bool fl1 = true, fl2 = true;

for (int j = 0; j < matr.GetLength(1)-1; ++j)

{

if (matr[i, j] < matr[i, j + 1] && fl1)

{

fl2 = false;

}

else if (matr[i, j] > matr[i, j + 1] && fl2)

{

fl1 = false;

}

else

{

fl1 = false;

fl2 = false;

}

}

if (fl1 || fl2) ans[i] = 1;

}

return ans;

}

private void goButton\_Click(object sender, EventArgs e)

{

if (!HandleInputs())

return;

var ans = Solve(matr);

for (int i = 0; i < matr.GetLength(0); i++)

{

resultPanel.Controls[i].Text = ans[i].ToString();

}

}

}

}

# К задаче 6

Листинг программы:

using System;

using System.Collections.Generic;

using System.Security.Cryptography;

namespace Task\_6

{

public class Program

{

public static int ReadInt(int left = -10000, int right = 10000)

{

bool ok = false;

int number = 0;

do

{

try

{

number = int.Parse(Console.ReadLine());

if (number >= left && number <= right) ok = true;

else

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введено не целое число. Введите целое число.");

ok = false;

}

catch (OverflowException)

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

} while (!ok);

return number;

}

public static List<double> arr;

private static void Rec(int index, double m)

{

double a = arr[index - 1] \* 3 / 2 - arr[index - 2] \* 2 / 3 - arr[index - 3] / 3;

if (a < m) return;

arr.Add(a);

Rec(index + 1, m);

}

public static double[] Solve(double a1, double a2, double a3, double m, double n, out int j)

{

arr = new List<double>();

double a;

j = 0;

arr.Add(a1);

arr.Add(a2);

arr.Add(a3);

Rec(3, m);

j = arr.Count - 3;

return arr.ToArray();

}

public static void Main(string[] args)

{

arr = new List<double>();

double a1, a2, a3, a;

int j = 0;

Console.WriteLine("Введите а1");

a1 = ReadInt();

arr.Add(a1);

Console.WriteLine("Введите а2");

a2 = ReadInt();

arr.Add(a2);

Console.WriteLine("Введите а3");

a3 = ReadInt();

arr.Add(a3);

Console.WriteLine("Введите M");

int m = ReadInt();

Console.WriteLine("Введите N");

int n = ReadInt();

Solve(a1, a2, a3, m, n,out j);

Console.WriteLine($"a={arr[arr.Count - 1]} J={j}");

Console.WriteLine(arr[arr.Count - 1] == m ? "А и М равны" : "А и М не равны");

if (j > n)

{

Console.WriteLine("J больше N");

}

else if (j < n)

{

Console.WriteLine("J меньше N");

}

else

{

Console.WriteLine("J равно N");

}

}

}

}

Листинг WinForm designer

namespace Task6Form

{

partial class Form1

{

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

/// <param name="disposing">истинно, если управляемый ресурс должен быть удален; иначе ложно.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Код, автоматически созданный конструктором форм Windows

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

private void InitializeComponent()

{

this.button = new System.Windows.Forms.Button();

this.label1 = new System.Windows.Forms.Label();

this.a1Text = new System.Windows.Forms.TextBox();

this.a2Text = new System.Windows.Forms.TextBox();

this.label2 = new System.Windows.Forms.Label();

this.a3Text = new System.Windows.Forms.TextBox();

this.label3 = new System.Windows.Forms.Label();

this.mText = new System.Windows.Forms.TextBox();

this.label4 = new System.Windows.Forms.Label();

this.nText = new System.Windows.Forms.TextBox();

this.label5 = new System.Windows.Forms.Label();

this.ansCompare = new System.Windows.Forms.Label();

this.jAns = new System.Windows.Forms.Label();

this.label6 = new System.Windows.Forms.Label();

this.order = new System.Windows.Forms.TextBox();

this.SuspendLayout();

//

// button

//

this.button.Location = new System.Drawing.Point(278, 362);

this.button.Name = "button";

this.button.Size = new System.Drawing.Size(75, 23);

this.button.TabIndex = 0;

this.button.Text = "Готово";

this.button.UseVisualStyleBackColor = true;

this.button.Click += new System.EventHandler(this.button\_Click);

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(49, 77);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(19, 13);

this.label1.TabIndex = 1;

this.label1.Text = "a1";

//

// a1Text

//

this.a1Text.Location = new System.Drawing.Point(91, 77);

this.a1Text.Name = "a1Text";

this.a1Text.Size = new System.Drawing.Size(100, 20);

this.a1Text.TabIndex = 2;

this.a1Text.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// a2Text

//

this.a2Text.Location = new System.Drawing.Point(91, 103);

this.a2Text.Name = "a2Text";

this.a2Text.Size = new System.Drawing.Size(100, 20);

this.a2Text.TabIndex = 4;

this.a2Text.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// label2

//

this.label2.AutoSize = true;

this.label2.Location = new System.Drawing.Point(49, 103);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(19, 13);

this.label2.TabIndex = 3;

this.label2.Text = "a2";

//

// a3Text

//

this.a3Text.Location = new System.Drawing.Point(91, 129);

this.a3Text.Name = "a3Text";

this.a3Text.Size = new System.Drawing.Size(100, 20);

this.a3Text.TabIndex = 6;

this.a3Text.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// label3

//

this.label3.AutoSize = true;

this.label3.Location = new System.Drawing.Point(49, 129);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(19, 13);

this.label3.TabIndex = 5;

this.label3.Text = "a3";

//

// mText

//

this.mText.Location = new System.Drawing.Point(91, 155);

this.mText.Name = "mText";

this.mText.Size = new System.Drawing.Size(100, 20);

this.mText.TabIndex = 8;

this.mText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// label4

//

this.label4.AutoSize = true;

this.label4.Location = new System.Drawing.Point(49, 155);

this.label4.Name = "label4";

this.label4.Size = new System.Drawing.Size(16, 13);

this.label4.TabIndex = 7;

this.label4.Text = "M";

//

// nText

//

this.nText.Location = new System.Drawing.Point(91, 181);

this.nText.Name = "nText";

this.nText.Size = new System.Drawing.Size(100, 20);

this.nText.TabIndex = 10;

this.nText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// label5

//

this.label5.AutoSize = true;

this.label5.Location = new System.Drawing.Point(49, 181);

this.label5.Name = "label5";

this.label5.Size = new System.Drawing.Size(15, 13);

this.label5.TabIndex = 9;

this.label5.Text = "N";

//

// ansCompare

//

this.ansCompare.AutoSize = true;

this.ansCompare.Location = new System.Drawing.Point(254, 110);

this.ansCompare.Name = "ansCompare";

this.ansCompare.Size = new System.Drawing.Size(0, 13);

this.ansCompare.TabIndex = 13;

//

// jAns

//

this.jAns.AutoSize = true;

this.jAns.Location = new System.Drawing.Point(257, 76);

this.jAns.Name = "jAns";

this.jAns.Size = new System.Drawing.Size(0, 13);

this.jAns.TabIndex = 14;

//

// label6

//

this.label6.AutoSize = true;

this.label6.Location = new System.Drawing.Point(254, 135);

this.label6.Name = "label6";

this.label6.Size = new System.Drawing.Size(115, 13);

this.label6.TabIndex = 15;

this.label6.Text = "Последовательность";

//

// order

//

this.order.Location = new System.Drawing.Point(382, 135);

this.order.Multiline = true;

this.order.Name = "order";

this.order.ReadOnly = true;

this.order.ScrollBars = System.Windows.Forms.ScrollBars.Vertical;

this.order.Size = new System.Drawing.Size(181, 78);

this.order.TabIndex = 16;

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.order);

this.Controls.Add(this.label6);

this.Controls.Add(this.jAns);

this.Controls.Add(this.ansCompare);

this.Controls.Add(this.nText);

this.Controls.Add(this.label5);

this.Controls.Add(this.mText);

this.Controls.Add(this.label4);

this.Controls.Add(this.a3Text);

this.Controls.Add(this.label3);

this.Controls.Add(this.a2Text);

this.Controls.Add(this.label2);

this.Controls.Add(this.a1Text);

this.Controls.Add(this.label1);

this.Controls.Add(this.button);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Button button;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox a1Text;

private System.Windows.Forms.TextBox a2Text;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox a3Text;

private System.Windows.Forms.Label label3;

private System.Windows.Forms.TextBox mText;

private System.Windows.Forms.Label label4;

private System.Windows.Forms.TextBox nText;

private System.Windows.Forms.Label label5;

private System.Windows.Forms.Label ansCompare;

private System.Windows.Forms.Label jAns;

private System.Windows.Forms.Label label6;

private System.Windows.Forms.TextBox order;

}

}

Листинг WinForm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task6Form

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void KeyPress(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\d,\-]") && e.KeyChar != 8)

e.Handled = true;

}

private static List<double> arr;

private static void Rec(int index, double m)

{

double a = arr[index - 1] \* 3 / 2 - arr[index - 2] \* 2 / 3 - arr[index - 3] / 3;

if (a < m) return;

arr.Add(a);

Rec(index + 1, m);

}

public static double[] Solve(double a1,double a2,ref double a3, double m, double n,out int j)

{

arr = new List<double>();

double a;

j = 0;

arr.Add(a1);

arr.Add(a2);

arr.Add(a3);

Rec(3, m);

j = arr.Count-3;

return arr.ToArray();

}

private void button\_Click(object sender, EventArgs e)

{

double a1, a2, a3, m, n;

int j;

if (a1Text.Text != "" && a2Text.Text != "" && a3Text.Text != "" && mText.Text != "" && nText.Text != "" &&

double.TryParse(a1Text.Text, out a1) && double.TryParse(a2Text.Text, out a2) && double.TryParse(a3Text.Text, out a3)

&& double.TryParse(mText.Text, out m) && double.TryParse(nText.Text, out n))

{

var ans = Solve(a1, a2,ref a3, m, n, out j);

ansCompare.Text = a3 == m ? "А и М равны" : "А и М не равны";

if (j > n)

{

jAns.Text=($"J больше N");

}

else if (j < n)

{

jAns.Text = ($"J меньше N");

}

else

{

jAns.Text = ($"J равно N");

}

string str = "";

for(int i = 3; i < (ans.Length); ++i)

{

str += ans[i].ToString() + "\r\n";

}

order.Text = str;

}

else

{

MessageBox.Show("Неправильный ввод");

}

}

}

}

# К задаче 7

Листинг программы:

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

namespace Task\_7

{

public class HuffmanTree

{

public class Node

{

public char Symbol { get; set; }

public int Frequency { get; set; }

public Node Right { get; set; }

public Node Left { get; set; }

// Find the encoded code for a symbol from the current node

public List<bool> Traverse(char symbol, List<bool> data)

{

// Leaf

if (Right == null && Left == null)

{

if (symbol.Equals(Symbol))

{

return data;

}

return null;

}

else

{

List<bool> left = null;

List<bool> right = null;

if (Left != null)

{

List<bool> leftPathData = new List<bool>();

leftPathData.AddRange(data);

leftPathData.Add(false);

left = Left.Traverse(symbol, leftPathData);

}

if (Right != null)

{

List<bool> rightPathData = new List<bool>();

rightPathData.AddRange(data);

rightPathData.Add(true);

right = Right.Traverse(symbol, rightPathData);

}

if (left != null)

{

return left;

}

else

{

return right;

}

}

}

}

private List<Node> nodes = new List<Node>();

public Node Root { get; set; }

public Dictionary<char, int> SymbolFrequencies = new Dictionary<char, int>();

private int[] frequancies = new int[0];

public void BuildHuffmanTree(int[] source)

{

frequancies = source;

for (int i = 0; i < frequancies.Length; ++i)

{

SymbolFrequencies.Add((char)i, frequancies[i]);

nodes.Add(new Node { Symbol = (char)i, Frequency = frequancies[i] });

}

// Generate root nodes for the lowest frequencies and add it to the end of ordered nodes till only 1 node is left as main root of the complete huffman tree

while (nodes.Count >= 2)

{

// Step# 2: Sort the list of nodes based on its frequencies in ascending order

List<Node> orderedNodes = nodes.OrderBy(node => node.Frequency).ToList();

if (orderedNodes.Count >= 2)

{

// Take first two items

List<Node> taken = orderedNodes.Take(2).ToList();

// Create a parent node by combining the frequencies

Node parent = new Node

{

Symbol = '\*',

Frequency = taken[0].Frequency + taken[1].Frequency,

Left = taken[0],

Right = taken[1]

};

//Remove left and right nodes and add their parent to the end of nodes list

nodes.Remove(taken[0]);

nodes.Remove(taken[1]);

nodes.Add(parent);

}

Root = nodes.FirstOrDefault();

}

}

public Dictionary<char, string> Encode()

{

Dictionary<char, string> encode = new Dictionary<char, string>();

List<bool> encodedSource = new List<bool>();

foreach (var tmp in SymbolFrequencies)

{

List<bool> encodedSymbol = Root.Traverse(tmp.Key, new List<bool>());

encodedSource.AddRange(encodedSymbol);

string str = "";

foreach (bool bit in new BitArray(encodedSymbol.ToArray()))

str += bit ? "1" : "0";

string ans = "";

for (int i = str.Length - 1; i >= 0; --i)

ans += str[i];

encode.Add(tmp.Key, ans);

}

return encode;

}

}

public class Program

{

public static int ReadInt(int left = -10000, int right = 10000)

{

bool ok = false;

int number = 0;

do

{

try

{

number = int.Parse(Console.ReadLine());

if (number >= left && number <= right) ok = true;

else

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введено не целое число. Введите целое число.");

ok = false;

}

catch (OverflowException)

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

} while (!ok);

return number;

}

private static int size = 0;

public static int[] frequency = null;

public static Dictionary<char, string> Solve(int[] frequency)

{

HuffmanTree tree = new HuffmanTree();

tree.BuildHuffmanTree(frequency);

var dict = tree.Encode();

return dict;

}

public static void Main(string[] args)

{

Console.WriteLine("Введите количество символов");

int n = ReadInt(0);

frequency = new int[n];

for (int i = 0; i < n; ++i)

{

Console.WriteLine($"Введите частоту {i+1} символа");

frequency[i] = ReadInt(0);

}

var dict = Solve(frequency);

foreach (var tmp in dict)

Console.WriteLine(tmp.Value);

}

}

}

Листинг WinForm Designer

namespace Task7Form

{

partial class Form1

{

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

/// <param name="disposing">истинно, если управляемый ресурс должен быть удален; иначе ложно.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Код, автоматически созданный конструктором форм Windows

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

private void InitializeComponent()

{

this.label1 = new System.Windows.Forms.Label();

this.sizeText = new System.Windows.Forms.TextBox();

this.table = new System.Windows.Forms.TableLayoutPanel();

this.sizeButton = new System.Windows.Forms.Button();

this.tableButton = new System.Windows.Forms.Button();

this.SuspendLayout();

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(13, 78);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(166, 13);

this.label1.TabIndex = 0;

this.label1.Text = " Введите количество символов";

//

// sizeText

//

this.sizeText.Location = new System.Drawing.Point(185, 78);

this.sizeText.Name = "sizeText";

this.sizeText.Size = new System.Drawing.Size(100, 20);

this.sizeText.TabIndex = 1;

this.sizeText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPressInt);

//

// table

//

this.table.AutoSize = true;

this.table.ColumnCount = 3;

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 33.33333F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 33.33334F));

this.table.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 33.33334F));

this.table.Location = new System.Drawing.Point(16, 116);

this.table.Name = "table";

this.table.RowCount = 2;

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 50F));

this.table.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 50F));

this.table.Size = new System.Drawing.Size(361, 47);

this.table.TabIndex = 2;

//

// sizeButton

//

this.sizeButton.Location = new System.Drawing.Point(291, 78);

this.sizeButton.Name = "sizeButton";

this.sizeButton.Size = new System.Drawing.Size(75, 19);

this.sizeButton.TabIndex = 3;

this.sizeButton.Text = "Готово";

this.sizeButton.UseVisualStyleBackColor = true;

this.sizeButton.Click += new System.EventHandler(this.sizeButton\_Click);

//

// tableButton

//

this.tableButton.Location = new System.Drawing.Point(384, 116);

this.tableButton.Name = "tableButton";

this.tableButton.Size = new System.Drawing.Size(75, 23);

this.tableButton.TabIndex = 4;

this.tableButton.Text = "Готово";

this.tableButton.UseVisualStyleBackColor = true;

this.tableButton.Click += new System.EventHandler(this.tableButton\_Click);

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.tableButton);

this.Controls.Add(this.sizeButton);

this.Controls.Add(this.table);

this.Controls.Add(this.sizeText);

this.Controls.Add(this.label1);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox sizeText;

private System.Windows.Forms.TableLayoutPanel table;

private System.Windows.Forms.Button sizeButton;

private System.Windows.Forms.Button tableButton;

}

}

Листинг программы WinForm Form

using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task7Form

{

public partial class Form1 : Form

{

public class HuffmanTree

{

public class Node

{

public char Symbol { get; set; }

public int Frequency { get; set; }

public Node Right { get; set; }

public Node Left { get; set; }

// Find the encoded code for a symbol from the current node

public List<bool> Traverse(char symbol, List<bool> data)

{

// Leaf

if (Right == null && Left == null)

{

if (symbol.Equals(Symbol))

{

return data;

}

return null;

}

else

{

List<bool> left = null;

List<bool> right = null;

if (Left != null)

{

List<bool> leftPathData = new List<bool>();

leftPathData.AddRange(data);

leftPathData.Add(false);

left = Left.Traverse(symbol, leftPathData);

}

if (Right != null)

{

List<bool> rightPathData = new List<bool>();

rightPathData.AddRange(data);

rightPathData.Add(true);

right = Right.Traverse(symbol, rightPathData);

}

if (left != null)

{

return left;

}

else

{

return right;

}

}

}

}

private List<Node> nodes = new List<Node>();

public Node Root { get; set; }

public Dictionary<char, int> SymbolFrequencies = new Dictionary<char, int>();

private int[] frequancies = new int[0];

public void BuildHuffmanTree(int[] source)

{

frequancies = source;

for (int i = 0; i < frequancies.Length; ++i)

{

SymbolFrequencies.Add((char)i, frequancies[i]);

nodes.Add(new Node { Symbol = (char)i, Frequency = frequancies[i] });

}

// Generate root nodes for the lowest frequencies and add it to the end of ordered nodes till only 1 node is left as main root of the complete huffman tree

while (nodes.Count >= 2)

{

// Step# 2: Sort the list of nodes based on its frequencies in ascending order

List<Node> orderedNodes = nodes.OrderBy(node => node.Frequency).ToList();

if (orderedNodes.Count >= 2)

{

// Take first two items

List<Node> taken = orderedNodes.Take(2).ToList();

// Create a parent node by combining the frequencies

Node parent = new Node

{

Symbol = '\*',

Frequency = taken[0].Frequency + taken[1].Frequency,

Left = taken[0],

Right = taken[1]

};

//Remove left and right nodes and add their parent to the end of nodes list

nodes.Remove(taken[0]);

nodes.Remove(taken[1]);

nodes.Add(parent);

}

Root = nodes.FirstOrDefault();

}

}

public Dictionary<char, string> Encode()

{

Dictionary<char, string> encode = new Dictionary<char, string>();

List<bool> encodedSource = new List<bool>();

foreach (var tmp in SymbolFrequencies)

{

List<bool> encodedSymbol = Root.Traverse(tmp.Key, new List<bool>());

encodedSource.AddRange(encodedSymbol);

string str = "";

foreach (bool bit in new BitArray(encodedSymbol.ToArray()))

str += bit ? "1" : "0";

string ans = "";

for (int i = str.Length - 1; i >= 0; --i)

ans += str[i];

encode.Add(tmp.Key, ans);

}

return encode;

}

}

private int size = 0;

public int[] frequency = null;

public Form1()

{

InitializeComponent();

}

private void KeyPressInt(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\d]") && e.KeyChar != 8 && e.KeyChar != 46)

e.Handled = true;

}

private void KeyPressDouble(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\d]") && e.KeyChar != 8)

e.Handled = true;

}

private void sizeButton\_Click(object sender, EventArgs e)

{

if (sizeText.Text == "")

{

MessageBox.Show($"Заполните поле");

return;

}

size = int.Parse(sizeText.Text);

table.Controls.Clear();

table.ColumnCount = 3;

table.RowCount = 2;

for(int i=0;i<size;++i)

table.RowStyles.Add(new RowStyle());

var label1 = new Label();

var label2 = new Label();

var label3 = new Label();

label1.Text = "Символ";

label2.Text = "Частота встречаемости символа";

label3.Text = "Код символа";

table.Controls.Add(label1, 0, 0);

table.Controls.Add(label2, 1, 0);

table.Controls.Add(label3, 2, 0);

for(int i = 1; i <= size; ++i)

{

var symbol = new TextBox();

var frequency = new TextBox();

var codes = new TextBox();

frequency.KeyPress += new KeyPressEventHandler(KeyPressDouble);

codes.ReadOnly = true;

table.Controls.Add(symbol, 0, i);

table.Controls.Add(frequency, 1, i);

table.Controls.Add(codes, 2, i);

}

}

private bool HandleInput()

{

double sum = 0;

bool ok = true;

for(int i = 0; i < size && ok; ++i)

{

if (table.Controls[3+i \* 3].Text == "" && table.Controls[3+i \* 3+1].Text == "")

{

ok = false;

MessageBox.Show($"Заполните все поля");

}

double tmp=double.Parse(table.Controls[3+i \* 3 + 1].Text);

sum += tmp;

}

return ok;

}

public static Dictionary<char,string> Solve(int[] frequency)

{

HuffmanTree tree = new HuffmanTree();

tree.BuildHuffmanTree(frequency);

var dict = tree.Encode();

return dict;

}

private void tableButton\_Click(object sender, EventArgs e)

{

if (!HandleInput())

return;

frequency = new int[size];

string str = "";

for (int i = 0; i < size; ++i)

{

frequency[i] = int.Parse(table.Controls[(i+1) \* 3 + 1].Text);

}

var dict=Solve(frequency);

int iter = 0;

string stR = "";

foreach(var tmp in dict)

{

table.Controls[(iter + 1) \* 3 + 2].Text = tmp.Value;

stR += tmp.Value;

iter++;

}

}

}

}

# К задаче 8

Листинг программы:

using System;

using System.Collections.Generic;

namespace Task\_8

{

public class Program

{

static public int n;

static public List<List<int>> g = new List<List<int>>();

static public char[] cl;

static public int[] p;

static public int cycle\_st, cycle\_end;

static bool dfs(int v)

{

cl[v] = '1';

for (int i = 0; i < g[v].Count; ++i)

{

int to = g[v][i];

if (cl[to] == '0')

{

p[to] = v;

if (dfs(to)) return true;

}

else if (cl[to] == '1')

{

cycle\_end = v;

cycle\_st = to;

return true;

}

}

cl[v] = '2';

return false;

}

public static void Solve(int n,int k)

{

p = new int[n];

cl = new char[n];

for (int i = 0; i < n; ++i)

{

cycle\_st = -1;

for (int j = 0; j < n; ++j)

{

p[j] = -1;

cl[j] = '0';

}

if (dfs(i))

{

int w = 0;

for (int j = 0; j < n; ++j)

{

if (p[j] != -1) w++;

}

if (w == k - 1) break;

}

}

}

public static void Main(string[] args)

{

n = int.Parse(Console.ReadLine());

int k = int.Parse(Console.ReadLine());

g = new List<List<int>>(n);

for (int i = 0; i < n; ++i)

{

g.Add(new List<int>());

g[i] = new List<int>();

}

for (int i = 0; i < n; ++i)

{

string[] tmp = Console.ReadLine().Split(' ');

for (int j = 0; j < n; ++j)

{

int q = Int32.Parse(tmp[j]);

if (q == 1)

{

g[i].Add(j);

}

}

}

Solve(n, k);

if (cycle\_st == -1)

Console.WriteLine("Цикла такой длины нет");

else

{

Console.WriteLine("Цикл такой длины есть");

List<int> cycle = new List<int>();

cycle.Add(cycle\_st);

for (int v = cycle\_end; v != cycle\_st; v = p[v])

cycle.Add(v);

cycle.Add(cycle\_st);

for (int i = 0; i < cycle.Count; ++i)

Console.Write(cycle[i] + 1 + " ");

}

}

}

}

Листинг Генератора тестов

using System;

using System.IO;

namespace Generaror\_Task\_8

{

internal class Program

{

public static void Main(string[] args)

{

for (int i = 3; i < 100; ++i)

{

Random rnd = new Random();

StreamWriter sw = new StreamWriter($"test{i}.txt");

sw.WriteLine(i);

sw.WriteLine(rnd.Next(i));

int[,] arr = new int[i,i];

for (int j = 0; j < rnd.Next(i \* i); ++j)

{

int w = rnd.Next(i), e = rnd.Next(i);

if(w==e) { w = rnd.Next(i); e = rnd.Next(i);}

arr[w, e] = 1;

arr[e, w] = 1;

}

for (int j = 0; j < i; ++j)

{

for (int w = 0; w < i; ++w)

{

sw.Write(arr[j,w]+" ");

}

sw.WriteLine();

}

sw.Close();

}

}

}

}

Листинг WinForm Designer

namespace Task8Form

{

partial class Form1

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

this.label2 = new System.Windows.Forms.Label();

this.goButton = new System.Windows.Forms.Button();

this.label3 = new System.Windows.Forms.Label();

this.rndButton = new System.Windows.Forms.Button();

this.tableLayout = new System.Windows.Forms.TableLayoutPanel();

this.ansText = new System.Windows.Forms.TextBox();

this.label1 = new System.Windows.Forms.Label();

this.cycleText = new System.Windows.Forms.TextBox();

this.SuspendLayout();

//

// label2

//

this.label2.AutoSize = true;

this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label2.Location = new System.Drawing.Point(12, 54);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(72, 18);

this.label2.TabIndex = 2;

this.label2.Text = "Матрица:";

//

// goButton

//

this.goButton.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.goButton.Location = new System.Drawing.Point(592, 316);

this.goButton.Name = "goButton";

this.goButton.Size = new System.Drawing.Size(165, 27);

this.goButton.TabIndex = 4;

this.goButton.Text = "Готово";

this.goButton.UseVisualStyleBackColor = true;

this.goButton.Click += new System.EventHandler(this.goButton\_Click);

//

// label3

//

this.label3.AutoSize = true;

this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label3.Location = new System.Drawing.Point(488, 55);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(80, 18);

this.label3.TabIndex = 6;

this.label3.Text = "Результат";

//

// rndButton

//

this.rndButton.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.rndButton.Location = new System.Drawing.Point(592, 259);

this.rndButton.Name = "rndButton";

this.rndButton.Size = new System.Drawing.Size(165, 49);

this.rndButton.TabIndex = 7;

this.rndButton.Text = "Заполнить случайными числами";

this.rndButton.UseVisualStyleBackColor = true;

this.rndButton.Click += new System.EventHandler(this.rndButton\_Click);

//

// tableLayout

//

this.tableLayout.ColumnCount = 10;

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.Location = new System.Drawing.Point(15, 75);

this.tableLayout.Name = "tableLayout";

this.tableLayout.RowCount = 10;

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 10F));

this.tableLayout.Size = new System.Drawing.Size(467, 268);

this.tableLayout.TabIndex = 9;

//

// ansText

//

this.ansText.Location = new System.Drawing.Point(491, 77);

this.ansText.Multiline = true;

this.ansText.Name = "ansText";

this.ansText.ReadOnly = true;

this.ansText.ScrollBars = System.Windows.Forms.ScrollBars.Both;

this.ansText.Size = new System.Drawing.Size(169, 68);

this.ansText.TabIndex = 10;

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(13, 26);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(160, 13);

this.label1.TabIndex = 11;

this.label1.Text = "Цикл какой длины надо найти";

//

// cycleText

//

this.cycleText.Location = new System.Drawing.Point(180, 26);

this.cycleText.Name = "cycleText";

this.cycleText.Size = new System.Drawing.Size(39, 20);

this.cycleText.TabIndex = 12;

this.cycleText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPressDec);

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.BackColor = System.Drawing.SystemColors.Control;

this.ClientSize = new System.Drawing.Size(769, 349);

this.Controls.Add(this.cycleText);

this.Controls.Add(this.label1);

this.Controls.Add(this.ansText);

this.Controls.Add(this.tableLayout);

this.Controls.Add(this.rndButton);

this.Controls.Add(this.label3);

this.Controls.Add(this.goButton);

this.Controls.Add(this.label2);

this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;

this.Name = "Form1";

this.Load += new System.EventHandler(this.Task5Form\_Load);

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Label label2;

private System.Windows.Forms.Button goButton;

private System.Windows.Forms.Label label3;

private System.Windows.Forms.Button rndButton;

private System.Windows.Forms.TableLayoutPanel tableLayout;

private System.Windows.Forms.TextBox ansText;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox cycleText;

}

}

Литинг WinWorm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task8Form

{

public partial class Form1 : Form

{

private Random rand = new Random();

private static int[,] matr = new int[10, 10];

private static int cycle = 0;

private static int edge = 20;

public Form1()

{

InitializeComponent();

}

private void Task5Form\_Load(object sender, EventArgs e)

{

for (int i = 0; i < tableLayout.ColumnCount; i++)

{

for(int j = 0; j < tableLayout.RowCount; ++j)

{

var text = new TextBox();

text.KeyPress += new KeyPressEventHandler(KeyPressBin);

text.Name = $"text{i}{j}";

text.Text = "0";

tableLayout.Controls.Add(text, i, j);

}

}

}

private void KeyPressBin(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[0-1]") && e.KeyChar != 8 && e.KeyChar != 46)

e.Handled = true;

}

private void KeyPressDec(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\d]") && e.KeyChar != 8 && e.KeyChar != 46)

e.Handled = true;

}

private void rndButton\_Click(object sender, EventArgs e)

{

for(int i = 0; i < 100; ++i)

tableLayout.Controls[i].Text = "0";

Random rnd = new Random();

for(int i = 0; i < edge; ++i)

{

int n = rnd.Next(10), m = rnd.Next(10);

if (n != m)

{

tableLayout.Controls[n \* 10 + m].Text = "1";

tableLayout.Controls[m \* 10 + n].Text = "1";

}

}

}

private bool HandleInputs()

{

bool ok = true;

string error = "";

for (int i = 0; i < tableLayout.Controls.Count && ok; i++)

{

ok = int.TryParse(tableLayout.Controls[i].Text, out matr[i / 10, i % 10]);

if (!ok)

error = $"Данные в строке {i / 10 + 1}, столбце {i % 10 + 1} введены некорректно";

}

if (!ok)

MessageBox.Show(error, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

if (cycleText.Text != "")

{

cycle = int.Parse(cycleText.Text);

}

else

{

ok = false;

MessageBox.Show("Не правильный ввод длины цикла", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

return ok;

}

private static List<List<int>> g = new List<List<int>>(); // граф

private static int n = 10; // число вершин

private static List<bool> used = new List<bool>();

private static List<int> p = new List<int>();

private static int lenCount;

private static void dfs(int v)

{

used[v] = true;

for (var i = 0; i < g[v].Count; ++i)

{

int to = g[v][i];

if (!used[to])

{

p[v] = to;

dfs(to);

}

}

}

public static string Solve(int[,] matr,int k)

{

int n = 10;

g = new List<List<int>>();

for (int i = 0; i < n; ++i)

{

g.Add(new List<int>());

g[i] = new List<int>();

}

for (int i = 0; i < matr.GetLength(0); ++i)

{

for (int j = 0; j < matr.GetLength(1); ++j)

{

if (matr[i, j] == 1)

g[i].Add(j);

}

}

var ans = "";

bool ok = false;

int beginning = 0;

List<int> ansPoint = new List<int>();

for(int i = 0; i < n; ++i)

{

used = new List<bool>();

p = new List<int>();

ansPoint = new List<int>();

for(int j = 0; j < n; ++j)

{

used.Add(false);

p.Add(-1);

}

dfs(i);

beginning = i;

var tmp = k;

for (int v = beginning; v != -1 && tmp > 0; v = p[v], tmp--)

ansPoint.Add(v);

if (ansPoint.Count >= k)

{

ok = true;

break;

}

}

if (!ok)

ans+="Цепи такой длины нет";

else

{

for (int v=beginning; v!=-1 && k>0;v=p[v],k--)

ans+=(v + 1 + " ");

}

return ans;

}

private void goButton\_Click(object sender, EventArgs e)

{

if (!HandleInputs())

return;

var ans = Solve(matr,cycle).ToString();

ansText.Text = ans.ToString();

}

}

}

# К задаче 9

Листинг программы:

using System;

using System.Collections.Generic;

using System.IO;

namespace Task\_9

{

public class List

{

public class Node

{

public int Data { get; set; }

public Node Next { get; set; }

public Node(int item)

{

Data = item;

Next = null;

}

}

public int Length { get; protected set; }

private Node head;

private Node tail;

public List()

{

Length = 0;

head = null;

tail = null;

}

public List(int size)

{

head = null;

tail = null;

for (int i = 0; i <= size; i++)

Add(i);

}

public int this[int index]

{

get

{

if (index < 0 || index > Length) throw new IndexOutOfRangeException();

return FindNode(index).Next.Data;

}

set

{

if (index < 0 || index > Length) throw new IndexOutOfRangeException();

Node node = FindNode(index).Next;

node.Data = value;

}

}

public Node FindNode(int index)

{

Node find = head;

index--;

int ind = 0;

while (ind < index)

{

find = find.Next;

ind++;

}

return find;

}

public int Find(int value)

{

Node find = head;

int index = 0;

for (int i = 1; i <= Length; ++i)

{

if (find.Data == value) return i - 2;

find = find.Next;

}

return -1;

}

public void Add(int item)

{

Node node = new Node(item);

if (head == null) head = node;

else tail.Next = node;

Length++;

tail = node;

}

public void Remove(int index)

{

//if (index < 0) throw new ArgumentOutOfRangeException("Индекс", "Индекс должен быть выражен неотрицательным числом");

//if (index > Length) throw new ArgumentOutOfRangeException("Индекс", "Индекс должен быть меньше или равно числу элементов в листе");

if (index == 0) head = head.Next;

else

{

Node node = FindNode(index);

node.Next = node.Next.Next;

}

Length--;

}

public void Clear()

{

Node node = head;

while (node != null)

{

Node prev = node;

node = node.Next;

prev.Next = null;

}

head = null;

tail = null;

Length = 0;

}

}

internal class Program

{

public static int ReadInt(int left = -100, int right = 100)

{

bool ok = false;

int number = 0;

do

{

try

{

number = int.Parse(Console.ReadLine());

if (number >= left && number <= right) ok = true;

else

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введено не целое число. Введите целое число.");

ok = false;

}

catch (OverflowException)

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

} while (!ok);

return number;

}

public static void Main(string[] args)

{

int n = ReadInt(0);

List arr = new List(n+1);

for (int i=1;i<n+1;++i)

{

Console.Write(arr[i]+" ");

}

Console.WriteLine();

int search = arr.Find(4);

Console.WriteLine(search);

arr.Remove(4);

for (int i=1;i<arr.Length;++i)

{

Console.Write(arr[i]+" ");

}

}

}

}

Листинг WinForm Designed

namespace Task9Form

{

partial class Form1

{

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

/// <param name="disposing">истинно, если управляемый ресурс должен быть удален; иначе ложно.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Код, автоматически созданный конструктором форм Windows

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

private void InitializeComponent()

{

this.initButton = new System.Windows.Forms.Button();

this.label1 = new System.Windows.Forms.Label();

this.initText = new System.Windows.Forms.TextBox();

this.findText = new System.Windows.Forms.TextBox();

this.label2 = new System.Windows.Forms.Label();

this.delText = new System.Windows.Forms.TextBox();

this.label3 = new System.Windows.Forms.Label();

this.findButton = new System.Windows.Forms.Button();

this.deleteButton = new System.Windows.Forms.Button();

this.label4 = new System.Windows.Forms.Label();

this.arrText = new System.Windows.Forms.TextBox();

this.SuspendLayout();

//

// initButton

//

this.initButton.Location = new System.Drawing.Point(293, 73);

this.initButton.Name = "initButton";

this.initButton.Size = new System.Drawing.Size(75, 23);

this.initButton.TabIndex = 0;

this.initButton.Text = "Создать";

this.initButton.UseVisualStyleBackColor = true;

this.initButton.Click += new System.EventHandler(this.initButton\_Click);

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(90, 78);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(90, 13);

this.label1.TabIndex = 1;

this.label1.Text = "Введите размер";

//

// initText

//

this.initText.Location = new System.Drawing.Point(187, 75);

this.initText.Name = "initText";

this.initText.Size = new System.Drawing.Size(100, 20);

this.initText.TabIndex = 2;

this.initText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// findText

//

this.findText.Location = new System.Drawing.Point(187, 101);

this.findText.Name = "findText";

this.findText.Size = new System.Drawing.Size(100, 20);

this.findText.TabIndex = 4;

this.findText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// label2

//

this.label2.AutoSize = true;

this.label2.Location = new System.Drawing.Point(24, 104);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(156, 13);

this.label2.TabIndex = 3;

this.label2.Text = "Элемент который надо найти";

//

// delText

//

this.delText.Location = new System.Drawing.Point(187, 127);

this.delText.Name = "delText";

this.delText.Size = new System.Drawing.Size(100, 20);

this.delText.TabIndex = 6;

this.delText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// label3

//

this.label3.AutoSize = true;

this.label3.Location = new System.Drawing.Point(58, 128);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(122, 13);

this.label3.TabIndex = 5;

this.label3.Text = "Элемент для удаления";

//

// findButton

//

this.findButton.Location = new System.Drawing.Point(293, 99);

this.findButton.Name = "findButton";

this.findButton.Size = new System.Drawing.Size(75, 23);

this.findButton.TabIndex = 7;

this.findButton.Text = "Найти";

this.findButton.UseVisualStyleBackColor = true;

this.findButton.Click += new System.EventHandler(this.findButton\_Click);

//

// deleteButton

//

this.deleteButton.Location = new System.Drawing.Point(293, 125);

this.deleteButton.Name = "deleteButton";

this.deleteButton.Size = new System.Drawing.Size(75, 23);

this.deleteButton.TabIndex = 8;

this.deleteButton.Text = "Удалить";

this.deleteButton.UseVisualStyleBackColor = true;

this.deleteButton.Click += new System.EventHandler(this.deleteButton\_Click);

//

// label4

//

this.label4.AutoSize = true;

this.label4.Location = new System.Drawing.Point(425, 78);

this.label4.Name = "label4";

this.label4.Size = new System.Drawing.Size(46, 13);

this.label4.TabIndex = 9;

this.label4.Text = "Массив";

//

// arrText

//

this.arrText.Location = new System.Drawing.Point(428, 99);

this.arrText.Name = "arrText";

this.arrText.ReadOnly = true;

this.arrText.ScrollBars = System.Windows.Forms.ScrollBars.Vertical;

this.arrText.Size = new System.Drawing.Size(110, 20);

this.arrText.TabIndex = 10;

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.arrText);

this.Controls.Add(this.label4);

this.Controls.Add(this.deleteButton);

this.Controls.Add(this.findButton);

this.Controls.Add(this.delText);

this.Controls.Add(this.label3);

this.Controls.Add(this.findText);

this.Controls.Add(this.label2);

this.Controls.Add(this.initText);

this.Controls.Add(this.label1);

this.Controls.Add(this.initButton);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Button initButton;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox initText;

private System.Windows.Forms.TextBox findText;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox delText;

private System.Windows.Forms.Label label3;

private System.Windows.Forms.Button findButton;

private System.Windows.Forms.Button deleteButton;

private System.Windows.Forms.Label label4;

private System.Windows.Forms.TextBox arrText;

}

}

Листинг WinForm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task9Form

{

public partial class Form1 : Form

{

public class List

{

public class Node

{

public int Data { get; set; }

public Node Next { get; set; }

public Node(int item)

{

Data = item;

Next = null;

}

}

public int Length { get; protected set; }

private Node head;

private Node tail;

public List()

{

Length = 0;

head = null;

tail = null;

}

public List(int size)

{

head = null;

tail = null;

for (int i = 0; i <= size; i++)

Add(i);

}

public int this[int index]

{

get

{

if (index < 0 || index > Length) throw new IndexOutOfRangeException();

return FindNode(index).Next.Data;

}

set

{

if (index < 0 || index > Length) throw new IndexOutOfRangeException();

Node node = FindNode(index).Next;

node.Data = value;

}

}

public Node FindNode(int index)

{

Node find = head;

index--;

int ind = 0;

while (ind < index)

{

find = find.Next;

ind++;

}

return find;

}

public int Find(int value)

{

Node find = head;

int index = 0;

for(int i = 1; i <= Length; ++i)

{

if (find.Data == value) return i-2;

find = find.Next;

}

return -1;

}

public void Add(int item)

{

Node node = new Node(item);

if (head == null) head = node;

else tail.Next = node;

Length++;

tail = node;

}

public void Remove(int index)

{

if (index < 0) throw new ArgumentOutOfRangeException("Индекс", "Индекс должен быть выражен неотрицательным числом");

if (index > Length) throw new ArgumentOutOfRangeException("Индекс", "Индекс должен быть меньше или равно числу элементов в листе");

if (index == 0) head = head.Next;

else

{

Node node = FindNode(index);

node.Next = node.Next.Next;

}

Length--;

}

public void Clear()

{

Node node = head;

while (node != null)

{

Node prev = node;

node = node.Next;

prev.Next = null;

}

head = null;

tail = null;

Length = 0;

}

}

private static List arr=null;

public Form1()

{

InitializeComponent();

}

private void KeyPress(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\d]") && e.KeyChar != 8 && e.KeyChar != 46)

e.Handled = true;

}

private void ShowArr()

{

string str = "";

for(int i = 1; i < arr.Length; ++i)

{

str += arr[i].ToString() + " ";

}

arrText.Text = str;

}

private void initButton\_Click(object sender, EventArgs e)

{

if (initText.Text == "")

{

MessageBox.Show("Заполните поле");

return;

}

int n = int.Parse(initText.Text);

arr = new List(n);

ShowArr();

}

private void findButton\_Click(object sender, EventArgs e)

{

if (arr == null)

{

MessageBox.Show("Инициализируйте массив");

return;

}

if (findText.Text == "")

{

MessageBox.Show("Заполните поле");

return;

}

int find = int.Parse(findText.Text);

var findres = arr.Find(find);

if (findres== -1)

MessageBox.Show("Такого элемента нет");

else

MessageBox.Show($"Индекс этого элемента {findres}");

}

private void deleteButton\_Click(object sender, EventArgs e)

{

if (arr == null)

{

MessageBox.Show("Инициализируйте массив");

return;

}

if (delText.Text == "")

{

MessageBox.Show("Заполните поле");

return;

}

int del = int.Parse(delText.Text);

arr.Remove(del);

ShowArr();

}

}

}

# К задаче 10

Листинг программы:

using System;

namespace Task\_10

{

public class List

{

public class Node

{

public double Data { get; set; }

public Node Next { get; set; }

public Node(double item)

{

Data = item;

Next = null;

}

}

public int Length { get; protected set; }

private Node head;

private Node tail;

public List()

{

Length = 0;

head = null;

tail = null;

}

public List(double[] arr)

{

head = null;

tail = null;

for (int i = 0; i < arr.Length; i++)

Add(arr[i]);

}

public double this[int index]

{

get

{

if (index < 0 || index > Length) throw new IndexOutOfRangeException();

return FindNode(index).Next.Data;

}

set

{

if (index < 0 || index > Length) throw new IndexOutOfRangeException();

Node node = FindNode(index).Next;

node.Data = value;

}

}

public Node FindNode(int index)

{

Node find = head;

index--;

int ind = 0;

while (ind < index)

{

find = find.Next;

ind++;

}

return find;

}

public void Add(double item)

{

Node node = new Node(item);

if (head == null) head = node;

else tail.Next = node;

Length++;

tail = node;

}

public void Clear()

{

Node node = head;

while (node != null)

{

Node prev = node;

node = node.Next;

prev.Next = null;

}

head = null;

tail = null;

Length = 0;

}

}

internal class Program

{

public static int ReadInt(int left = -100, int right = 100)

{

bool ok = false;

int number = 0;

do

{

try

{

number = int.Parse(Console.ReadLine());

if (number >= left && number <= right) ok = true;

else

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введено не целое число. Введите целое число.");

ok = false;

}

catch (OverflowException)

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

} while (!ok);

return number;

}

public static double ReadD(int left = -100, int right = 100)

{

bool ok = false;

double number = 0;

do

{

try

{

number = double.Parse(Console.ReadLine());

if (number >= left && number <= right) ok = true;

else

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка. Введено не действительное число. Введите действительное число.");

ok = false;

}

catch (OverflowException)

{

Console.WriteLine($"Ошибка. Число выход за границы. Введите число большее {left} и меньшее {right}");

ok = false;

}

} while (!ok);

return number;

}

public static void Main(string[] args)

{

Console.WriteLine("Введите размер массива");

int n = ReadInt();

double[] arr = new double[n];

for (int i = 0; i < n; ++i)

{

Console.WriteLine($"Введите {i+1} элемент");

arr[i] = ReadD();

}

List ans = new List(arr);

for (int i = 0; i < n-1; ++i)

{

ans[i] = arr[i] - arr[n - 1];

Console.Write(ans[i]+" ");

}

}

}

}

Листинг WinForm Designer

namespace Task10Form

{

partial class Form1

{

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

/// <param name="disposing">истинно, если управляемый ресурс должен быть удален; иначе ложно.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Код, автоматически созданный конструктором форм Windows

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

private void InitializeComponent()

{

this.button = new System.Windows.Forms.Button();

this.label1 = new System.Windows.Forms.Label();

this.sizeText = new System.Windows.Forms.TextBox();

this.arrText = new System.Windows.Forms.TextBox();

this.label2 = new System.Windows.Forms.Label();

this.ansText = new System.Windows.Forms.TextBox();

this.label3 = new System.Windows.Forms.Label();

this.SuspendLayout();

//

// button

//

this.button.Location = new System.Drawing.Point(181, 194);

this.button.Name = "button";

this.button.Size = new System.Drawing.Size(75, 23);

this.button.TabIndex = 0;

this.button.Text = "Готово";

this.button.UseVisualStyleBackColor = true;

this.button.Click += new System.EventHandler(this.button\_Click);

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(12, 104);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(137, 13);

this.label1.TabIndex = 1;

this.label1.Text = "Введите размер массива";

//

// sizeText

//

this.sizeText.Location = new System.Drawing.Point(156, 97);

this.sizeText.Name = "sizeText";

this.sizeText.Size = new System.Drawing.Size(100, 20);

this.sizeText.TabIndex = 2;

this.sizeText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// arrText

//

this.arrText.Location = new System.Drawing.Point(156, 123);

this.arrText.Name = "arrText";

this.arrText.ReadOnly = true;

this.arrText.Size = new System.Drawing.Size(100, 20);

this.arrText.TabIndex = 4;

//

// label2

//

this.label2.AutoSize = true;

this.label2.Location = new System.Drawing.Point(12, 130);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(46, 13);

this.label2.TabIndex = 3;

this.label2.Text = "Массив";

//

// ansText

//

this.ansText.Location = new System.Drawing.Point(156, 149);

this.ansText.Name = "ansText";

this.ansText.ReadOnly = true;

this.ansText.Size = new System.Drawing.Size(100, 20);

this.ansText.TabIndex = 6;

//

// label3

//

this.label3.AutoSize = true;

this.label3.Location = new System.Drawing.Point(12, 156);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(59, 13);

this.label3.TabIndex = 5;

this.label3.Text = "Результат";

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.ansText);

this.Controls.Add(this.label3);

this.Controls.Add(this.arrText);

this.Controls.Add(this.label2);

this.Controls.Add(this.sizeText);

this.Controls.Add(this.label1);

this.Controls.Add(this.button);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Button button;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox sizeText;

private System.Windows.Forms.TextBox arrText;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox ansText;

private System.Windows.Forms.Label label3;

}

}

Листинг WinForm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task10Form

{

public class List

{

public class Node

{

public double Data { get; set; }

public Node Next { get; set; }

public Node(double item)

{

Data = item;

Next = null;

}

}

public int Length { get; protected set; }

private Node head;

private Node tail;

public List()

{

Length = 0;

head = null;

tail = null;

}

public List(double[] arr)

{

head = null;

tail = null;

for (int i = 0; i < arr.Length; i++)

Add(arr[i]);

}

public double this[int index]

{

get

{

if (index < 0 || index > Length) throw new IndexOutOfRangeException();

return FindNode(index).Next.Data;

}

set

{

if (index < 0 || index > Length) throw new IndexOutOfRangeException();

Node node = FindNode(index).Next;

node.Data = value;

}

}

public Node FindNode(int index)

{

Node find = head;

index--;

int ind = 0;

while (ind < index)

{

find = find.Next;

ind++;

}

return find;

}

public void Add(double item)

{

Node node = new Node(item);

if (head == null) head = node;

else tail.Next = node;

Length++;

tail = node;

}

public void Clear()

{

Node node = head;

while (node != null)

{

Node prev = node;

node = node.Next;

prev.Next = null;

}

head = null;

tail = null;

Length = 0;

}

}

public partial class Form1 : Form

{

private static List arr=null;

public Form1()

{

InitializeComponent();

}

private void KeyPress(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\d]") && e.KeyChar != 8 && e.KeyChar != 46)

e.Handled = true;

}

public static List Solve(int n,List arr)

{

List ans = new List();

for (int i = 0; i < n - 1; ++i)

{

ans.Add(arr[i] - arr[n - 1]);

}

return ans;

}

private void button\_Click(object sender, EventArgs e)

{

if (sizeText.Text == "")

{

MessageBox.Show("Заполните поле");

return;

}

int n = int.Parse(sizeText.Text);

if (n <= 2)

{

MessageBox.Show("Размер массива должен быть больше 2");

return;

}

Random rnd = new Random();

double[] tmp = new double[n];

for (int i = 0; i < n; ++i)

{

tmp[i] = rnd.Next(100);

}

arr = new List(tmp);

var arrAns = Solve(n, arr);

string s = "";

string sAns = "";

for (int i = 1; i < n; ++i)

s += arr[i].ToString() + " ";

for (int i = 1; i < n-1; ++i)

sAns += arrAns[i].ToString() + " ";

arrText.Text = s;

ansText.Text = sAns;

}

}

}

# К задаче 11

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |

Листинг программы:

using System;

using System.Net;

namespace Task\_11

{

public class Program

{

public static string Code(string s)

{

string code = s[0].ToString();

for (int i = 1; i < s.Length; ++i)

{

code += s[i] == s[i - 1] ? 1 : 0;

}

return code;

}

public static string Encode(string code)

{

string encode = code[0].ToString();

for (int i = 1; i < code.Length; ++i)

{

if (code[i] == '1')

{

encode += encode[i - 1];

}

else

{

encode += encode[i - 1] == '1' ? 0 : 1;

}

}

return encode;

}

public static void Main(string[] args)

{

Console.WriteLine("Введите последовательность 0 и 1 без пробелов в одну строку");

string s = Console.ReadLine();

var code = Code(s);

Console.WriteLine($"Зашифрованная последовательность {code}");

var encode = Encode(code);

Console.WriteLine($"Расшифрованная последовательность {encode}");

}

}

}

Листинг WinForm Designed

namespace Task11Form

{

partial class Form1

{

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

/// <param name="disposing">истинно, если управляемый ресурс должен быть удален; иначе ложно.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Код, автоматически созданный конструктором форм Windows

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

private void InitializeComponent()

{

this.button = new System.Windows.Forms.Button();

this.label1 = new System.Windows.Forms.Label();

this.orderText = new System.Windows.Forms.TextBox();

this.codeText = new System.Windows.Forms.TextBox();

this.label2 = new System.Windows.Forms.Label();

this.encodeText = new System.Windows.Forms.TextBox();

this.label3 = new System.Windows.Forms.Label();

this.SuspendLayout();

//

// button

//

this.button.Location = new System.Drawing.Point(277, 200);

this.button.Name = "button";

this.button.Size = new System.Drawing.Size(75, 23);

this.button.TabIndex = 0;

this.button.Text = "Готово";

this.button.UseVisualStyleBackColor = true;

this.button.Click += new System.EventHandler(this.button\_Click);

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(45, 70);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(158, 13);

this.label1.TabIndex = 1;

this.label1.Text = "Введите последовательность";

//

// orderText

//

this.orderText.Location = new System.Drawing.Point(254, 63);

this.orderText.Name = "orderText";

this.orderText.Size = new System.Drawing.Size(100, 20);

this.orderText.TabIndex = 2;

this.orderText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// codeText

//

this.codeText.Location = new System.Drawing.Point(254, 93);

this.codeText.Name = "codeText";

this.codeText.ReadOnly = true;

this.codeText.Size = new System.Drawing.Size(100, 20);

this.codeText.TabIndex = 4;

//

// label2

//

this.label2.AutoSize = true;

this.label2.Location = new System.Drawing.Point(45, 96);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(199, 13);

this.label2.TabIndex = 3;

this.label2.Text = "Зашифрованная последовательность";

//

// encodeText

//

this.encodeText.Location = new System.Drawing.Point(254, 119);

this.encodeText.Name = "encodeText";

this.encodeText.ReadOnly = true;

this.encodeText.Size = new System.Drawing.Size(100, 20);

this.encodeText.TabIndex = 6;

//

// label3

//

this.label3.AutoSize = true;

this.label3.Location = new System.Drawing.Point(45, 122);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(201, 13);

this.label3.TabIndex = 5;

this.label3.Text = "Дешифрованная последовательность";

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.encodeText);

this.Controls.Add(this.label3);

this.Controls.Add(this.codeText);

this.Controls.Add(this.label2);

this.Controls.Add(this.orderText);

this.Controls.Add(this.label1);

this.Controls.Add(this.button);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Button button;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox orderText;

private System.Windows.Forms.TextBox codeText;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox encodeText;

private System.Windows.Forms.Label label3;

}

}

Листинг WinForm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task11Form

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void KeyPress(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[0-1]") && e.KeyChar != 8 && e.KeyChar != 46)

e.Handled = true;

}

public static string Code(string str)

{

string code = str[0].ToString();

for (int i = 1; i < str.Length; ++i)

{

code += str[i] == str[i - 1] ? 1 : 0;

}

return code;

}

public static string Encode(string code)

{

string encode = code[0].ToString();

for (int i = 1; i < code.Length; ++i)

{

if (code[i] == '1')

{

encode += encode[i - 1];

}

else

{

encode += encode[i - 1] == '1' ? 0 : 1;

}

}

return encode;

}

private void button\_Click(object sender, EventArgs e)

{

if (orderText.Text == "")

{

MessageBox.Show("Заполните ячейку");

return;

}

string str = orderText.Text;

string code = Code(str);

codeText.Text = code;

string encode = Encode(code);

encodeText.Text = encode;

}

}

}

# К задаче 12

Листинг программы:

using System;

public class Programm

{

public static int[] HeapSort(int[] arr)

{

int n = arr.Length;

for (int i = n / 2 - 1; i >= 0; i--)

MakeHeap(arr, n, i);

for (int i = n - 1; i >= 0; i--)

{

Swap(ref arr[0],ref arr[i]);

MakeHeap(arr, i, 0);

}

return arr;

}

static void MakeHeap(int[] arr, int n, int i)

{

int largest = i;

int l = 2 \* i + 1; // left = 2\*i + 1

int r = 2 \* i + 2; // right = 2\*i + 2

if (l < n && arr[l] > arr[largest])

largest = l;

if (r < n && arr[r] > arr[largest])

largest = r;

if (largest != i)

{

Swap(ref arr[i],ref arr[largest]);

MakeHeap(arr, n, largest);

}

}

static void Swap(ref int l, ref int r)

{

int tmp = l;

l = r;

r = tmp;

}

public static int[] CoctailSort(int[] arr)

{

int l = 0, r = arr.Length;

while (l<r)

{

for (int i = l; i < r-1; ++i)

if(arr[i]>arr[i+1])

Swap(ref arr[i],ref arr[i+1]);

r--;

for (int i = r; i >l; --i)

if(arr[i]<arr[i-1])

Swap(ref arr[i],ref arr[i-1]);

l++;

}

return arr;

}

public static void Main()

{

int[] arr = {12, 11, 13, 5, 6, 7};

int n = arr.Length;

arr = HeapSort(arr);

Console.WriteLine("HeapSort");

for (int i = 0; i < n; ++i)

{

Console.Write(arr[i]+" ");

}

Console.WriteLine();

arr = new []{12, 11, 13, 5, 6, 7};

arr=CoctailSort(arr);

Console.WriteLine("CocktaiSort");

for (int i = 0; i < n; ++i)

{

Console.Write(arr[i]+" ");

}

}

}

Листинг WinForm Designed

namespace Task12Form

{

partial class Form1

{

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

/// <param name="disposing">истинно, если управляемый ресурс должен быть удален; иначе ложно.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Код, автоматически созданный конструктором форм Windows

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

private void InitializeComponent()

{

this.label1 = new System.Windows.Forms.Label();

this.sizeText = new System.Windows.Forms.TextBox();

this.arrText = new System.Windows.Forms.TextBox();

this.label2 = new System.Windows.Forms.Label();

this.cocktailText = new System.Windows.Forms.TextBox();

this.label3 = new System.Windows.Forms.Label();

this.heapText = new System.Windows.Forms.TextBox();

this.label4 = new System.Windows.Forms.Label();

this.button = new System.Windows.Forms.Button();

this.countCompCocText = new System.Windows.Forms.TextBox();

this.countCompHeapText = new System.Windows.Forms.TextBox();

this.countSwapCocText = new System.Windows.Forms.TextBox();

this.countSwapHeapText = new System.Windows.Forms.TextBox();

this.label5 = new System.Windows.Forms.Label();

this.label6 = new System.Windows.Forms.Label();

this.label7 = new System.Windows.Forms.Label();

this.label8 = new System.Windows.Forms.Label();

this.countSwapHeapUpText = new System.Windows.Forms.TextBox();

this.countSwapCocUpText = new System.Windows.Forms.TextBox();

this.countCompHeapUpText = new System.Windows.Forms.TextBox();

this.countCompCocUpText = new System.Windows.Forms.TextBox();

this.heapUpText = new System.Windows.Forms.TextBox();

this.label9 = new System.Windows.Forms.Label();

this.cocktailUpText = new System.Windows.Forms.TextBox();

this.label10 = new System.Windows.Forms.Label();

this.arrUpText = new System.Windows.Forms.TextBox();

this.label11 = new System.Windows.Forms.Label();

this.label12 = new System.Windows.Forms.Label();

this.label13 = new System.Windows.Forms.Label();

this.countSwapHeapDownText = new System.Windows.Forms.TextBox();

this.countSwapCocDownText = new System.Windows.Forms.TextBox();

this.countCompHeapDownText = new System.Windows.Forms.TextBox();

this.countCompCocDownText = new System.Windows.Forms.TextBox();

this.heapDownText = new System.Windows.Forms.TextBox();

this.label14 = new System.Windows.Forms.Label();

this.cocktailDownText = new System.Windows.Forms.TextBox();

this.label15 = new System.Windows.Forms.Label();

this.arrDownText = new System.Windows.Forms.TextBox();

this.label16 = new System.Windows.Forms.Label();

this.SuspendLayout();

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(46, 77);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(137, 13);

this.label1.TabIndex = 0;

this.label1.Text = "Введите размер массива";

//

// sizeText

//

this.sizeText.Location = new System.Drawing.Point(306, 70);

this.sizeText.Name = "sizeText";

this.sizeText.Size = new System.Drawing.Size(100, 20);

this.sizeText.TabIndex = 1;

this.sizeText.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.KeyPress);

//

// arrText

//

this.arrText.Location = new System.Drawing.Point(306, 96);

this.arrText.Name = "arrText";

this.arrText.ReadOnly = true;

this.arrText.Size = new System.Drawing.Size(100, 20);

this.arrText.TabIndex = 3;

//

// label2

//

this.label2.AutoSize = true;

this.label2.Location = new System.Drawing.Point(46, 103);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(166, 13);

this.label2.TabIndex = 2;

this.label2.Text = "Массив заполненный случайно";

//

// cocktailText

//

this.cocktailText.Location = new System.Drawing.Point(306, 122);

this.cocktailText.Name = "cocktailText";

this.cocktailText.ReadOnly = true;

this.cocktailText.Size = new System.Drawing.Size(100, 20);

this.cocktailText.TabIndex = 5;

//

// label3

//

this.label3.AutoSize = true;

this.label3.Location = new System.Drawing.Point(46, 129);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(254, 13);

this.label3.TabIndex = 4;

this.label3.Text = "Результат работы сортировки перемешиванием";

//

// heapText

//

this.heapText.Location = new System.Drawing.Point(306, 148);

this.heapText.Name = "heapText";

this.heapText.ReadOnly = true;

this.heapText.Size = new System.Drawing.Size(100, 20);

this.heapText.TabIndex = 7;

//

// label4

//

this.label4.AutoSize = true;

this.label4.Location = new System.Drawing.Point(46, 155);

this.label4.Name = "label4";

this.label4.Size = new System.Drawing.Size(244, 13);

this.label4.TabIndex = 6;

this.label4.Text = "Результат работы пирамидальной сортировки";

//

// button

//

this.button.Location = new System.Drawing.Point(543, 362);

this.button.Name = "button";

this.button.Size = new System.Drawing.Size(75, 23);

this.button.TabIndex = 8;

this.button.Text = "Готово";

this.button.UseVisualStyleBackColor = true;

this.button.Click += new System.EventHandler(this.button\_Click);

//

// countCompCocText

//

this.countCompCocText.Location = new System.Drawing.Point(412, 122);

this.countCompCocText.Name = "countCompCocText";

this.countCompCocText.ReadOnly = true;

this.countCompCocText.Size = new System.Drawing.Size(100, 20);

this.countCompCocText.TabIndex = 9;

//

// countCompHeapText

//

this.countCompHeapText.Location = new System.Drawing.Point(412, 148);

this.countCompHeapText.Name = "countCompHeapText";

this.countCompHeapText.ReadOnly = true;

this.countCompHeapText.Size = new System.Drawing.Size(100, 20);

this.countCompHeapText.TabIndex = 10;

//

// countSwapCocText

//

this.countSwapCocText.Location = new System.Drawing.Point(518, 122);

this.countSwapCocText.Name = "countSwapCocText";

this.countSwapCocText.ReadOnly = true;

this.countSwapCocText.Size = new System.Drawing.Size(100, 20);

this.countSwapCocText.TabIndex = 11;

//

// countSwapHeapText

//

this.countSwapHeapText.Location = new System.Drawing.Point(518, 148);

this.countSwapHeapText.Name = "countSwapHeapText";

this.countSwapHeapText.ReadOnly = true;

this.countSwapHeapText.Size = new System.Drawing.Size(100, 20);

this.countSwapHeapText.TabIndex = 12;

//

// label5

//

this.label5.AutoSize = true;

this.label5.Location = new System.Drawing.Point(431, 90);

this.label5.Name = "label5";

this.label5.Size = new System.Drawing.Size(69, 26);

this.label5.TabIndex = 13;

this.label5.Text = "Количество \r\nсравнений";

//

// label6

//

this.label6.AutoSize = true;

this.label6.Location = new System.Drawing.Point(528, 90);

this.label6.Name = "label6";

this.label6.Size = new System.Drawing.Size(78, 26);

this.label6.TabIndex = 14;

this.label6.Text = "Количество\r\nперемещений";

//

// label7

//

this.label7.AutoSize = true;

this.label7.Location = new System.Drawing.Point(528, 173);

this.label7.Name = "label7";

this.label7.Size = new System.Drawing.Size(78, 26);

this.label7.TabIndex = 26;

this.label7.Text = "Количество\r\nперемещений";

//

// label8

//

this.label8.AutoSize = true;

this.label8.Location = new System.Drawing.Point(431, 173);

this.label8.Name = "label8";

this.label8.Size = new System.Drawing.Size(69, 26);

this.label8.TabIndex = 25;

this.label8.Text = "Количество \r\nсравнений";

//

// countSwapHeapUpText

//

this.countSwapHeapUpText.Location = new System.Drawing.Point(518, 231);

this.countSwapHeapUpText.Name = "countSwapHeapUpText";

this.countSwapHeapUpText.ReadOnly = true;

this.countSwapHeapUpText.Size = new System.Drawing.Size(100, 20);

this.countSwapHeapUpText.TabIndex = 24;

//

// countSwapCocUpText

//

this.countSwapCocUpText.Location = new System.Drawing.Point(518, 205);

this.countSwapCocUpText.Name = "countSwapCocUpText";

this.countSwapCocUpText.ReadOnly = true;

this.countSwapCocUpText.Size = new System.Drawing.Size(100, 20);

this.countSwapCocUpText.TabIndex = 23;

//

// countCompHeapUpText

//

this.countCompHeapUpText.Location = new System.Drawing.Point(412, 231);

this.countCompHeapUpText.Name = "countCompHeapUpText";

this.countCompHeapUpText.ReadOnly = true;

this.countCompHeapUpText.Size = new System.Drawing.Size(100, 20);

this.countCompHeapUpText.TabIndex = 22;

//

// countCompCocUpText

//

this.countCompCocUpText.Location = new System.Drawing.Point(412, 205);

this.countCompCocUpText.Name = "countCompCocUpText";

this.countCompCocUpText.ReadOnly = true;

this.countCompCocUpText.Size = new System.Drawing.Size(100, 20);

this.countCompCocUpText.TabIndex = 21;

//

// heapUpText

//

this.heapUpText.Location = new System.Drawing.Point(306, 231);

this.heapUpText.Name = "heapUpText";

this.heapUpText.ReadOnly = true;

this.heapUpText.Size = new System.Drawing.Size(100, 20);

this.heapUpText.TabIndex = 20;

//

// label9

//

this.label9.AutoSize = true;

this.label9.Location = new System.Drawing.Point(46, 238);

this.label9.Name = "label9";

this.label9.Size = new System.Drawing.Size(244, 13);

this.label9.TabIndex = 19;

this.label9.Text = "Результат работы пирамидальной сортировки";

//

// cocktailUpText

//

this.cocktailUpText.Location = new System.Drawing.Point(306, 205);

this.cocktailUpText.Name = "cocktailUpText";

this.cocktailUpText.ReadOnly = true;

this.cocktailUpText.Size = new System.Drawing.Size(100, 20);

this.cocktailUpText.TabIndex = 18;

//

// label10

//

this.label10.AutoSize = true;

this.label10.Location = new System.Drawing.Point(46, 212);

this.label10.Name = "label10";

this.label10.Size = new System.Drawing.Size(254, 13);

this.label10.TabIndex = 17;

this.label10.Text = "Результат работы сортировки перемешиванием";

//

// arrUpText

//

this.arrUpText.Location = new System.Drawing.Point(306, 179);

this.arrUpText.Name = "arrUpText";

this.arrUpText.ReadOnly = true;

this.arrUpText.Size = new System.Drawing.Size(100, 20);

this.arrUpText.TabIndex = 16;

//

// label11

//

this.label11.AutoSize = true;

this.label11.Location = new System.Drawing.Point(46, 186);

this.label11.Name = "label11";

this.label11.Size = new System.Drawing.Size(240, 13);

this.label11.TabIndex = 15;

this.label11.Text = "Массив - возрастающая последовательность";

//

// label12

//

this.label12.AutoSize = true;

this.label12.Location = new System.Drawing.Point(528, 258);

this.label12.Name = "label12";

this.label12.Size = new System.Drawing.Size(78, 26);

this.label12.TabIndex = 38;

this.label12.Text = "Количество\r\nперемещений";

//

// label13

//

this.label13.AutoSize = true;

this.label13.Location = new System.Drawing.Point(431, 258);

this.label13.Name = "label13";

this.label13.Size = new System.Drawing.Size(69, 26);

this.label13.TabIndex = 37;

this.label13.Text = "Количество \r\nсравнений";

//

// countSwapHeapDownText

//

this.countSwapHeapDownText.Location = new System.Drawing.Point(518, 316);

this.countSwapHeapDownText.Name = "countSwapHeapDownText";

this.countSwapHeapDownText.ReadOnly = true;

this.countSwapHeapDownText.Size = new System.Drawing.Size(100, 20);

this.countSwapHeapDownText.TabIndex = 36;

//

// countSwapCocDownText

//

this.countSwapCocDownText.Location = new System.Drawing.Point(518, 290);

this.countSwapCocDownText.Name = "countSwapCocDownText";

this.countSwapCocDownText.ReadOnly = true;

this.countSwapCocDownText.Size = new System.Drawing.Size(100, 20);

this.countSwapCocDownText.TabIndex = 35;

//

// countCompHeapDownText

//

this.countCompHeapDownText.Location = new System.Drawing.Point(412, 316);

this.countCompHeapDownText.Name = "countCompHeapDownText";

this.countCompHeapDownText.ReadOnly = true;

this.countCompHeapDownText.Size = new System.Drawing.Size(100, 20);

this.countCompHeapDownText.TabIndex = 34;

//

// countCompCocDownText

//

this.countCompCocDownText.Location = new System.Drawing.Point(412, 290);

this.countCompCocDownText.Name = "countCompCocDownText";

this.countCompCocDownText.ReadOnly = true;

this.countCompCocDownText.Size = new System.Drawing.Size(100, 20);

this.countCompCocDownText.TabIndex = 33;

//

// heapDownText

//

this.heapDownText.Location = new System.Drawing.Point(306, 316);

this.heapDownText.Name = "heapDownText";

this.heapDownText.ReadOnly = true;

this.heapDownText.Size = new System.Drawing.Size(100, 20);

this.heapDownText.TabIndex = 32;

//

// label14

//

this.label14.AutoSize = true;

this.label14.Location = new System.Drawing.Point(46, 323);

this.label14.Name = "label14";

this.label14.Size = new System.Drawing.Size(244, 13);

this.label14.TabIndex = 31;

this.label14.Text = "Результат работы пирамидальной сортировки";

//

// cocktailDownText

//

this.cocktailDownText.Location = new System.Drawing.Point(306, 290);

this.cocktailDownText.Name = "cocktailDownText";

this.cocktailDownText.ReadOnly = true;

this.cocktailDownText.Size = new System.Drawing.Size(100, 20);

this.cocktailDownText.TabIndex = 30;

//

// label15

//

this.label15.AutoSize = true;

this.label15.Location = new System.Drawing.Point(46, 297);

this.label15.Name = "label15";

this.label15.Size = new System.Drawing.Size(254, 13);

this.label15.TabIndex = 29;

this.label15.Text = "Результат работы сортировки перемешиванием";

//

// arrDownText

//

this.arrDownText.Location = new System.Drawing.Point(306, 264);

this.arrDownText.Name = "arrDownText";

this.arrDownText.ReadOnly = true;

this.arrDownText.Size = new System.Drawing.Size(100, 20);

this.arrDownText.TabIndex = 28;

//

// label16

//

this.label16.AutoSize = true;

this.label16.Location = new System.Drawing.Point(46, 271);

this.label16.Name = "label16";

this.label16.Size = new System.Drawing.Size(224, 13);

this.label16.TabIndex = 27;

this.label16.Text = "Массив - убывающая последовательность";

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.label12);

this.Controls.Add(this.label13);

this.Controls.Add(this.countSwapHeapDownText);

this.Controls.Add(this.countSwapCocDownText);

this.Controls.Add(this.countCompHeapDownText);

this.Controls.Add(this.countCompCocDownText);

this.Controls.Add(this.heapDownText);

this.Controls.Add(this.label14);

this.Controls.Add(this.cocktailDownText);

this.Controls.Add(this.label15);

this.Controls.Add(this.arrDownText);

this.Controls.Add(this.label16);

this.Controls.Add(this.label7);

this.Controls.Add(this.label8);

this.Controls.Add(this.countSwapHeapUpText);

this.Controls.Add(this.countSwapCocUpText);

this.Controls.Add(this.countCompHeapUpText);

this.Controls.Add(this.countCompCocUpText);

this.Controls.Add(this.heapUpText);

this.Controls.Add(this.label9);

this.Controls.Add(this.cocktailUpText);

this.Controls.Add(this.label10);

this.Controls.Add(this.arrUpText);

this.Controls.Add(this.label11);

this.Controls.Add(this.label6);

this.Controls.Add(this.label5);

this.Controls.Add(this.countSwapHeapText);

this.Controls.Add(this.countSwapCocText);

this.Controls.Add(this.countCompHeapText);

this.Controls.Add(this.countCompCocText);

this.Controls.Add(this.button);

this.Controls.Add(this.heapText);

this.Controls.Add(this.label4);

this.Controls.Add(this.cocktailText);

this.Controls.Add(this.label3);

this.Controls.Add(this.arrText);

this.Controls.Add(this.label2);

this.Controls.Add(this.sizeText);

this.Controls.Add(this.label1);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox sizeText;

private System.Windows.Forms.TextBox arrText;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox cocktailText;

private System.Windows.Forms.Label label3;

private System.Windows.Forms.TextBox heapText;

private System.Windows.Forms.Label label4;

private System.Windows.Forms.Button button;

private System.Windows.Forms.TextBox countCompCocText;

private System.Windows.Forms.TextBox countCompHeapText;

private System.Windows.Forms.TextBox countSwapCocText;

private System.Windows.Forms.TextBox countSwapHeapText;

private System.Windows.Forms.Label label5;

private System.Windows.Forms.Label label6;

private System.Windows.Forms.Label label7;

private System.Windows.Forms.Label label8;

private System.Windows.Forms.TextBox countSwapHeapUpText;

private System.Windows.Forms.TextBox countSwapCocUpText;

private System.Windows.Forms.TextBox countCompCocUpText;

private System.Windows.Forms.TextBox heapUpText;

private System.Windows.Forms.Label label9;

private System.Windows.Forms.TextBox cocktailUpText;

private System.Windows.Forms.Label label10;

private System.Windows.Forms.TextBox arrUpText;

private System.Windows.Forms.Label label11;

private System.Windows.Forms.Label label12;

private System.Windows.Forms.Label label13;

private System.Windows.Forms.TextBox countSwapHeapDownText;

private System.Windows.Forms.TextBox countSwapCocDownText;

private System.Windows.Forms.TextBox countCompHeapDownText;

private System.Windows.Forms.TextBox countCompCocDownText;

private System.Windows.Forms.TextBox heapDownText;

private System.Windows.Forms.Label label14;

private System.Windows.Forms.TextBox cocktailDownText;

private System.Windows.Forms.Label label15;

private System.Windows.Forms.TextBox arrDownText;

private System.Windows.Forms.Label label16;

private System.Windows.Forms.TextBox countCompHeapUpText;

}

}

Листинг WinForm Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Task12Form

{

public partial class Form1 : Form

{

private static int countSwapCoc = 0, countCompCoc = 0;

private static int countSwapHeap = 0, countCompHeap = 0;

private static int countSwapCocUp = 0, countCompCocUp = 0;

private static int countSwapHeapUp = 0, countCompHeapUp = 0;

private static int countSwapCocDown = 0, countCompCocDown = 0;

private static int countSwapHeapDown = 0, countCompHeapDown = 0;

public Form1()

{

InitializeComponent();

}

private void KeyPress(object sender, KeyPressEventArgs e)

{

if (!Regex.IsMatch(e.KeyChar.ToString(), @"[\d]") && e.KeyChar != 8 && e.KeyChar != 46)

e.Handled = true;

}

public static int[] HeapSort(int[] arr, ref int countSwap, ref int countComp)

{

int n = arr.Length;

for (int i = n / 2 - 1; i >= 0; i--)

MakeHeap(arr, n, i, ref countSwap, ref countComp);

for (int i = n - 1; i >= 0; i--)

{

Swap(ref arr[0], ref arr[i],ref countSwap);

MakeHeap(arr, i, 0,ref countSwap,ref countComp);

}

return arr;

}

static void MakeHeap(int[] arr, int n, int i,ref int countSwap,ref int countComp)

{

int largest = i;

int l = 2 \* i + 1;

int r = 2 \* i + 2;

if (l < n && arr[l] > arr[largest])

largest = l;

countComp++;

if (r < n && arr[r] > arr[largest])

largest = r;

countComp++;

if (largest != i)

{

Swap(ref arr[i], ref arr[largest],ref countSwap);

MakeHeap(arr, n, largest,ref countSwap, ref countComp);

}

countComp++;

}

static void Swap(ref int l, ref int r,ref int count)

{

int tmp = l;

l = r;

r = tmp;

count++;

}

public static int[] CoctailSort(int[] arr, ref int countComp, ref int countSwap)

{

int l = 0, r = arr.Length;

while (l < r)

{

for (int i = l; i < r - 1; ++i)

{

if (arr[i] > arr[i + 1])

Swap(ref arr[i], ref arr[i + 1], ref countSwap);

countComp++;

}

r--;

for (int i = r; i > l; --i)

{

if (arr[i] < arr[i - 1])

Swap(ref arr[i], ref arr[i - 1], ref countSwap);

countComp++;

}

l++;

}

return arr;

}

private void ArrRnd()

{

countSwapCoc = 0; countCompCoc = 0;

countSwapHeap = 0; countCompHeap = 0;

int n = int.Parse(sizeText.Text);

var arr = new List<int>();

var rnd = new Random();

for (int i = 0; i < n; ++i)

arr.Add(rnd.Next(100));

var cocktail = arr.ToArray();

var heap = arr.ToArray();

cocktail = CoctailSort(cocktail, ref countCompCoc, ref countSwapCoc);

heap = HeapSort(heap, ref countCompHeap, ref countSwapHeap);

Show(arr, cocktail, heap);

}

private void ArrUp()

{

countSwapCocUp = 0; countCompCocUp = 0;

countSwapHeapUp = 0; countCompHeapUp = 0;

int n = int.Parse(sizeText.Text);

var arr = new List<int>();

for (int i = 0; i < n; ++i)

arr.Add(i);

var cocktail = arr.ToArray();

var heap = arr.ToArray();

cocktail = CoctailSort(cocktail, ref countCompCocUp, ref countSwapCocUp);

heap = HeapSort(heap, ref countCompHeapUp, ref countSwapHeapUp);

ShowUp(arr, cocktail, heap);

}

private void ArrDown()

{

countSwapCocDown = 0; countCompCocDown = 0;

countSwapHeapDown = 0; countCompHeapDown = 0;

int n = int.Parse(sizeText.Text);

var arr = new List<int>();

for (int i = 0; i < n; ++i)

arr.Add(n-i);

var cocktail = arr.ToArray();

var heap = arr.ToArray();

cocktail = CoctailSort(cocktail, ref countCompCocDown, ref countSwapCocDown);

heap = HeapSort(heap, ref countCompHeapDown, ref countSwapHeapDown);

ShowDown(arr, cocktail, heap);

}

private void button\_Click(object sender, EventArgs e)

{

if (sizeText.Text == "")

{

MessageBox.Show("Заполните поле");

return;

}

ArrRnd();

ArrUp();

ArrDown();

}

private void Show(List<int> arr,int[] cocktail,int[] heap)

{

int n = arr.Count;

string cocktailStr = "", heapStr = "", arrStr = "";

for (int i = 0; i < n; ++i)

arrStr += arr[i].ToString() + " ";

for (int i = 0; i < n; ++i)

cocktailStr += cocktail[i].ToString() + " ";

for (int i = 0; i < n; ++i)

heapStr += heap[i].ToString() + " ";

arrText.Text = arrStr;

cocktailText.Text = cocktailStr;

countCompCocText.Text = countCompCoc.ToString();

countSwapCocText.Text = countSwapCoc.ToString();

heapText.Text = heapStr;

countCompHeapText.Text = countCompHeap.ToString();

countSwapHeapText.Text = countSwapHeap.ToString();

}

private void ShowUp(List<int> arr, int[] cocktail, int[] heap)

{

int n = arr.Count;

string cocktailStr = "", heapStr = "", arrStr = "";

for (int i = 0; i < n; ++i)

arrStr += arr[i].ToString() + " ";

for (int i = 0; i < n; ++i)

cocktailStr += cocktail[i].ToString() + " ";

for (int i = 0; i < n; ++i)

heapStr += heap[i].ToString() + " ";

arrUpText.Text = arrStr;

cocktailUpText.Text = cocktailStr;

countCompCocUpText.Text = countCompCocUp.ToString();

countSwapCocUpText.Text = countSwapCocUp.ToString();

heapUpText.Text = heapStr;

countCompHeapUpText.Text = countCompHeapUp.ToString();

countSwapHeapUpText.Text = countSwapHeapUp.ToString();

}

private void ShowDown(List<int> arr, int[] cocktail, int[] heap)

{

int n = arr.Count;

string cocktailStr = "", heapStr = "", arrStr = "";

for (int i = 0; i < n; ++i)

arrStr += arr[i].ToString() + " ";

for (int i = 0; i < n; ++i)

cocktailStr += cocktail[i].ToString() + " ";

for (int i = 0; i < n; ++i)

heapStr += heap[i].ToString() + " ";

arrDownText.Text = arrStr;

cocktailDownText.Text = cocktailStr;

countCompCocDownText.Text = countCompCocDown.ToString();

countSwapCocDownText.Text = countSwapCocDown.ToString();

heapDownText.Text = heapStr;

countCompHeapDownText.Text = countCompHeapDown.ToString();

countSwapHeapDownText.Text = countSwapHeapDown.ToString();

}

}

}