



**Федеральное агентство по рыболовству**  
**Федеральное государственное бюджетное образовательное учреждение высшего образования**  
**«Астраханский государственный технический университет»**  
Система менеджмента качества в области образования, воспитания, науки и инноваций сертифицирована DQS по международному стандарту ISO 9001:2015

Институт информационных технологий и коммуникаций  
Направление подготовки 09.03.04 Программная инженерия  
Профиль «Разработка программно-информационных систем»  
Кафедра «Автоматизированные системы обработки информации и управления»

**КУРСОВОЙ ПРОЕКТ**  
**Учебно-демонстрационная программа**  
**«Методы сортировок: сортировка Шелла»**  
по дисциплине «Программирование и информатика»

Допущен к защите  
«\_\_» \_\_\_\_ 20\_\_ г.  
Руководитель

Оценка, полученная на защите  
«\_\_\_\_\_»

Проект выполнен  
обучающимся группы ДИПРб-11  
Катуниным С.С.

Руководитель  
ст. преп. Толасова В.В.

Члены комиссии:

Лаптев В.В.

Куркурин Н.Д.

Толасова В.В.

**Астрахань – 2019**

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ ПО РЫБОЛОВСТВУ**

**АСТРАХАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**УТВЕРЖДАЮ**

Заведующий кафедрой

к.т.н., доцент

С.В. Белов \_\_\_\_\_

Кафедра

«Автоматизированные системы

обработки информации и управления»

«\_\_\_\_\_\_» 201 \_\_\_\_ г.

**ЗАДАНИЕ**  
**на выполнение курсового проекта**

Обучающийся *Катунин Сергей Сергеевич*

Группа **ДИПРб-11**

Дисциплина **Программирование и информатика**

Тема **Учебно-демонстрационная программа «Методы сортировок: сортировка Шелла»**

Дата получения задания «\_\_\_\_\_\_» 201 \_\_\_\_ г.

Срок представления обучающимся КП на кафедру «\_\_\_\_\_\_» 201 \_\_\_\_ г.

Руководитель *ст. преподаватель* **Толасова В.В.** «\_\_\_\_\_\_» 201 \_\_\_\_ г.

должность, степень, звание подпись ФИО

Обучающийся **Катунин С.С.** «\_\_\_\_\_\_» 201 \_\_\_\_ г.

подпись ФИО

**Задачи**

Разработка программного продукта, который

- предоставляет пользователю теоретический материал по теме «Методы сортировок: сортировка Шелла»;
- визуализирует работу алгоритма сортировки Шелла;
- предоставляет пользователю возможность тестирования по теме «Методы сортировок: сортировка Шелла»;
- предоставляет пользователю режим администратора, который даёт возможность расшифровать и зашифровать файл с базой вопросов и файл с теорией.

**Список рекомендуемой литературы**

1. Златопольский Д.М. Программирование: типовые задачи, алгоритмы, методы 2-е изд. – М.: БИНОМ. Лаборатория знаний, 2012. – 223 с.: ил.
2. Лаптев В.В. C++. Экспресс-курс. – СПб.: БХВ-Петербург, 2004. – 512 с.: ил.

**УТВЕРЖДАЮ**

Заведующий кафедрой

к.т.н., доцент

С.В. Белов \_\_\_\_\_

« \_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ г.

К заданию на курсовой проект  
по дисциплине  
«Программирование и информатика»

**КАЛЕНДАРНЫЙ ГРАФИК**  
курсового проектирования

№ п/п	Разделы, темы и их содержание, графический материал	Дата сдачи	Объем, %
1	Выбор темы	25.02.2019	1
2	Техническое задание	11.03.2019	3
3	Разработка модели, проектирование системы <ul style="list-style-type: none"> <li>■ <i>введение,</i></li> <li>■ <i>технический проект,</i></li> <li>■ <i>программа и методика испытаний,</i></li> <li>■ <i>литература</i></li> </ul>	15.04.2019	25
4	Программная реализация системы <ul style="list-style-type: none"> <li>■ <i>работающая программа,</i></li> <li>■ <i>рабочий проект</i></li> <li>■ <i>корректированное техническое задание (при необходимости)</i></li> </ul>	13.05.2019	40
5	Тестирование и отладка системы, эксперименты <ul style="list-style-type: none"> <li>■ <i>работающая программа с внесёнными изменениями,</i></li> <li>■ <i>окончательные тексты всех разделов</i></li> </ul>	20.05.2019	50
6	Компоновка текста Подготовка презентации и доклада <ul style="list-style-type: none"> <li>■ <i>пояснительная записка</i></li> <li>■ <i>презентация</i></li> <li>■ <i>электронный носитель с текстом пояснительной записи, исходным кодом проекта, презентацией и готовым программным продуктом</i></li> </ul>	27.05.2019	59
7	Защита курсового проекта	11.06.2019- 15.06.2019	60-100

С графиком ознакомлен « \_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ г.

Катунин С.С. \_\_\_\_\_, обучающийся группы ДИПРб-11  
(фамилия, инициалы, подпись)

График курсового проектирования выполнен  
без отклонений / с незначительными отклонениями / со значительными отклонениями

нужное подчеркнуть

Руководитель курсового проекта \_\_\_\_\_ ст. преподаватель Толасова В.В.  
подпись, ученая степень, звание, фамилия, инициалы

## СОДЕРЖАНИЕ

Введение.....	6
1 Технический проект.....	7
1.1 Анализ предметной области.....	7
1.1.1 Алгоритм сортировки.....	7
1.1.2 Сортировка вставками.....	7
1.1.3 Сортировка Шелла.....	7
1.1.4 Проверка знаний.....	8
1.1.5 Режим администратора.....	8
1.2 Технология обработки информации.....	9
1.2.1 Форматы данных.....	10
1.2.2 Алгоритм заполнения массива случайными числами.....	11
1.2.3 Алгоритм заполнения массива значениями с клавиатуры.....	11
1.2.4 Алгоритм заполнения массива значениями из файла.....	11
1.2.5 Алгоритм сохранения состояния массива.....	12
1.2.6 Алгоритм сортировки Шелла.....	12
1.2.7 Алгоритм вывода состояния массива.....	13
1.2.8 Алгоритм визуализации алгоритма сортировки Шелла.....	13
1.2.9 Алгоритм шифрования файла.....	13
1.2.10 Алгоритм расшифровки файла.....	13
1.2.11 Алгоритм вывода теоретического материала.....	14
1.2.12 Алгоритм чтения базы вопросов из файла.....	14
1.2.13 Алгоритм тестирования.....	15
1.2.14 Режим администратора.....	15
1.2.15 Основной алгоритм.....	16
1.3 Входные и выходные данные.....	16
1.4 Системные требования.....	16
2 Рабочий проект.....	17
2.1 Общие сведения о работе системы.....	17
2.2 Функциональное назначение программного продукта.....	17
2.3 Инсталляция и выполнение программного продукта.....	17
2.4 Описание программы.....	18
2.5 Разработанные меню и интерфейсы.....	22
2.6 Сообщения системы.....	32
3 Программа и методика испытаний.....	33

3.1 Проверка работоспособности режима чтения теории.....	33
3.2 Проверка работоспособности режима демонстрации.....	33
3.3 Проверка работоспособности режима тестирования.....	34
3.4 Проверка работоспособности режима администратора.....	35
Заключение.....	37
Список использованных источников.....	38
Приложение 1 Техническое задание.....	39
Приложение 2 База текстовых вопросов.....	43

## **ВВЕДЕНИЕ**

Любому начинающему программисту не раз приходилось сталкиваться с сортировкой массивов, то есть с упорядочиванием элементов в массиве по возрастанию или по убыванию. Сортировка данных – очень частая задача, которая необходима, в первую очередь, для эффективного поиска и представления данных. Некоторые задачи с неотсортированными данными решить очень трудно, а некоторые просто невозможно.

Существует большое количество различных видов алгоритмов сортировок, выполняющих одну и ту же задачу, многие из которых являются оптимальными, а большинство имеет какие-либо преимущества по сравнению с остальными. Поэтому иногда от выбора оптимального алгоритма сортировки может зависеть эффективность программы. Чаще всего у студентов возникают некоторые сложности в понимании принципа работы этих сортировок, а также в процессе применения их при решении различных задач. Студент, не уделивший должного внимания этой теме, не может считаться хорошим программистом, так как не может реализовать эффективную сортировку данных для необходимой задачи. Таким образом, разработка учебно-демонстрационной программы, предлагающей теоретический материал по теме «Сортировка Шелла», а также тестирование с вопросами по этой теме, не только позволит начинающему программисту, впервые столкнувшемуся с таким методом сортировки, понять принцип его работы, но и поможет быстро научиться применять приобретенные знания на практике студенту, у которого возникают трудности с использованием сортировки Шелла.

Целью создания учебно-демонстрационной программы «Методы сортировок: сортировка Шелла» является автоматизация обучения и контроля знаний по теме «сортировка Шелла».

Назначение программы – повышение качества знаний студентов, снижение нагрузки на преподавателя.

## 1 ТЕХНИЧЕСКИЙ ПРОЕКТ

### 1.1 Анализ предметной области

#### 1.1.1 Алгоритм сортировки

Алгоритм сортировки – это алгоритм для упорядочивания элементов в массиве. В случае, когда элемент массива имеет несколько полей, поле, служащее критерием порядка, называется ключом сортировки. На практике в качестве ключа часто выступает число, а в остальных полях хранятся какие-либо данные, никак не влияющие на работу алгоритма.

#### 1.1.2 Сортировка вставками

Сортировка вставками – алгоритм сортировки, в котором элементы входной последовательности просматриваются по одному, и каждый новый поступивший элемент размещается в подходящее место среди ранее упорядоченных элементов. На рисунке 1.1 визуализирована сортировка вставками.

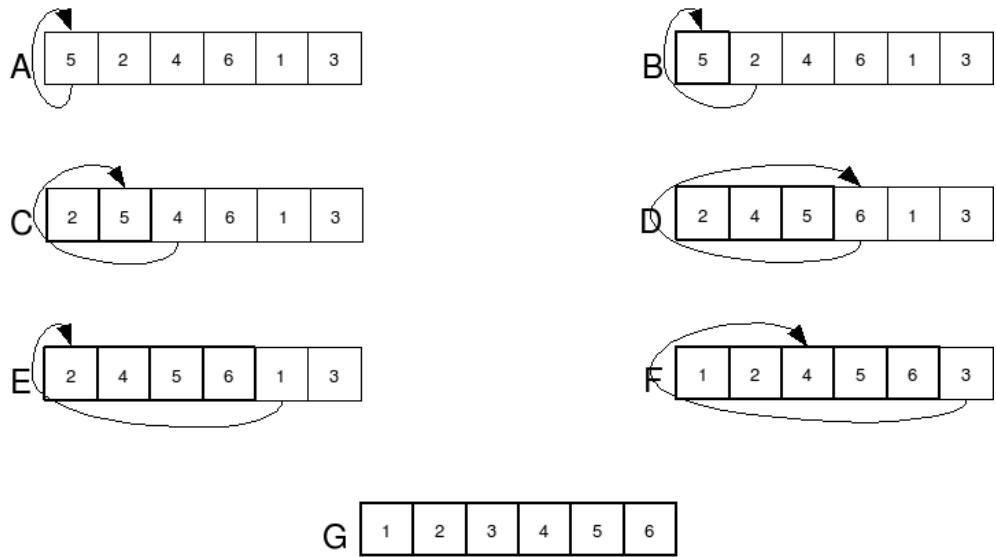


Рисунок 1.1 — Алгоритм сортировки вставками

#### 1.1.3 Сортировка Шелла

Сортировка Шелла – алгоритм сортировки, являющийся усовершенствованным вариантом сортировки вставками. Идея метода Шелла состоит в сравнении элементов, стоящих не только рядом, но и на определённом расстоянии друг от друга (расстояние  $d$ ). Иными словами — это сортировка вставками с предварительными «грубыми» проходами.

Пример: Пусть дан массив  $A = (23, 59, 17, 44, 25, 76, 43, 21, 88, 93, 33, 54, 69, 33)$  и выполняется его сортировка методом Шелла, а в качестве значений  $d$  выбраны  $(5, 3, 1)$ . На первом шаге сортируются подмассивы  $A$ , составленные из всех элементов  $A$ , отличающихся на 5 позиций, то есть подмассивы  $A_{5,1} = (23, 76, 33)$ ,  $A_{5,2} = (59, 43, 54)$ ,  $A_{5,3} = (17, 21, 69)$ ,

$A_{5,4} = (44, 88, 33)$ ,  $A_{5,5} = (25, 93)$ . В полученном массиве на втором шаге вновь сортируются подмассивы из отстоящих на 3 позиции элементов. Процесс завершается обычной сортировкой вставками получившегося массива. В таблице 1.1 визуализирован этот пример сортировки Шелла.

Таблица 1.1 — Алгоритм сортировки Шелла

Состояние массива	Массив	Кол-во обменов
Исходный массив	23 59 17 44 25 76 43 21 88 93 33 54 69 33	—
После сортировки с шагом 5	23 43 17 33 25 33 54 21 44 93 76 59 69 88	5 обменов
После сортировки с шагом 3	23 21 17 33 25 33 54 43 44 69 76 59 93 88	4 обмена
После сортировки с шагом 1	17 21 23 25 33 33 43 44 54 59 69 76 88 93	9 обменов

#### 1.1.4 Проверка знаний

После того, как студент ознакомился с теоретическим материалом по теме «Методы сортировок: сортировка Шелла», он может пройти тестирование, для того чтобы выяснить, насколько хорошо был усвоен полученный теоретический материал.

В случае если вопросов будет слишком много, студент может утомиться в процессе тестирования, однако, если вопросов будет слишком мало, то по результату тестирования нельзя будет утверждать, что тестируемый усвоил материал. Следует предоставить студенту пять вопросов закрытого типа с выбором одного правильного ответа из четырех предложенных и подсчитать процент верных ответов (традиционно считается, что уровень знаний тестируемого достаточен, если количество правильных ответов составляет не менее 60%).

В базе будет содержаться не менее десяти вопросов, пять из которых будут выбраны случайным образом и добавлены в тест. Порядок вариантов ответов каждого вопроса при повторном прохождении теста будет меняться.

По завершении тестирования студент сможет ознакомиться со своим результатом. Тестирование считается успешным, если количество правильных ответов не менее 3.

#### 1.1.5 Режим администратора

Файл с базой вопросов и файл с теoriей должны быть зашифрованы, чтобы студент не мог узнать правильные ответы на вопросы и не мог редактировать эти файлы по своему желанию.

Однако, возникают такие ситуации, когда преподавателю необходимо внести корректирующие изменения, если были найдены ошибки в формулировке вопроса, либо был указан неверный вариант ответа для вопроса, или необходимо изменить текст теоретического материала с целью улучшения.

Режим администратора позволяет расшифровать файлы, и после внесенных корректирующих изменений — зашифровать. Для того, чтобы перейти в режим администратора, необходимо ввести правильные логин и пароль во время авторизации.

## 1.2 Технология обработки информации

Анализ предметной области показал, что программа рассчитана на двух пользователей.

На рис. 1.2 показана диаграмма вариантов использования.

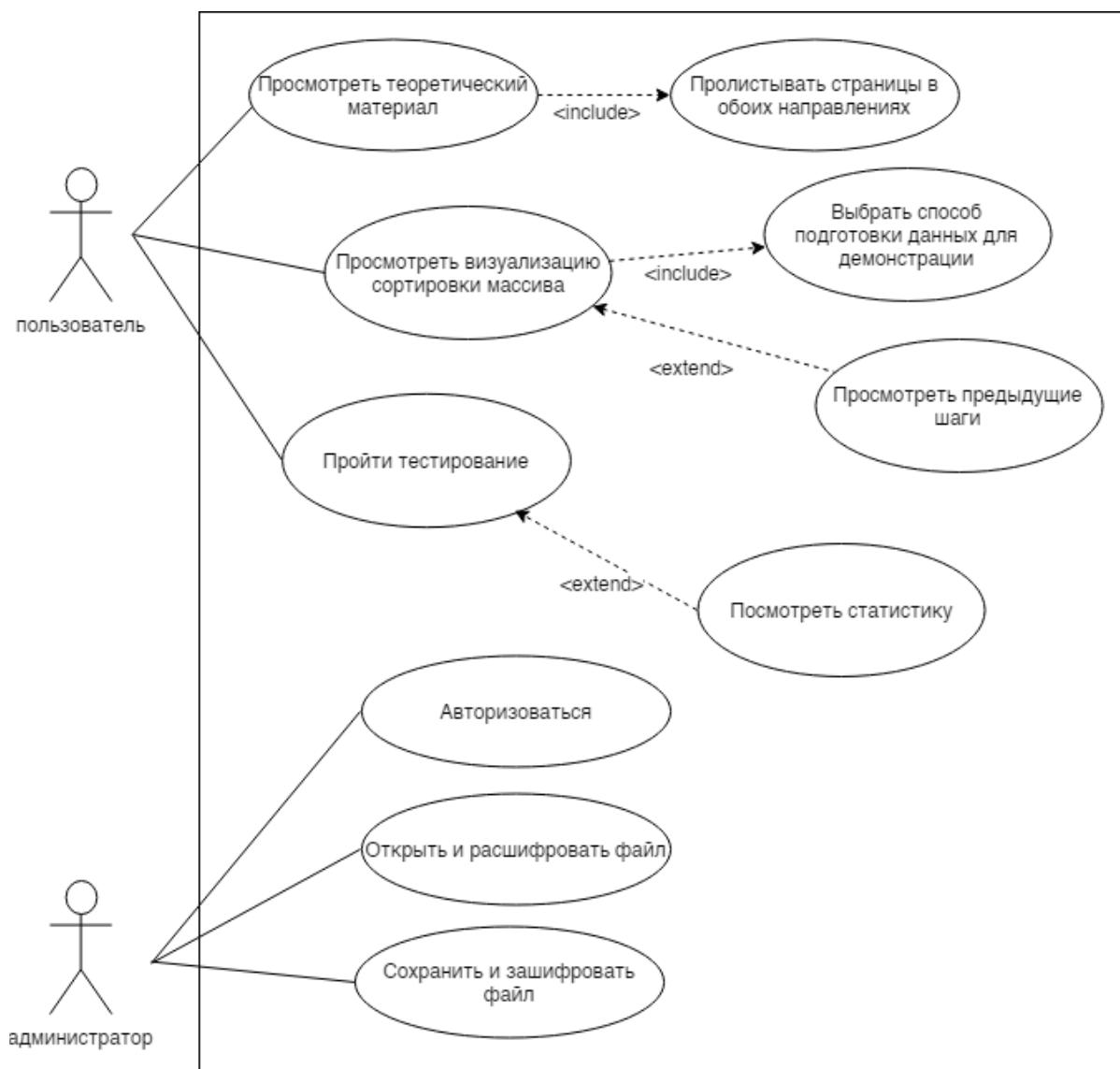


Рисунок 1.2 – Диаграмма вариантов использования

Вариант использования «Просмотреть теоретический материал» включает обязательную функцию «Пролистывать страницы в обоих направлениях».

Вариант использования «Просмотреть визуализацию сортировки массива» расширен функцией «Просмотреть предыдущие шаги», а также включает обязательную функцию «Выбрать способ подготовки данных для демонстрации».

Вариант использования «Пройти тестирование» расширен функцией «Посмотреть статистику».

### 1.2.1 Форматы данных

Теоретический материал, база вопросов и статистика результатов тестирования хранятся в зашифрованных файлах с названиями «teo.dat», «quest.dat» и «stats.dat» соответственно. Файлы зашифрованы (закодированы) таким образом, чтобы его нельзя было прочесть стандартными средствами операционной системы. Шифрование осуществляется путём сложения по модулю 2 (XOR-шифрование) ключа (символ S под кодом 83) с кодом каждого символа шифруемой строки. Администратор имеет возможность расшифровать файлы с теоретическим материалом и с базой вопросов, для того, чтобы можно было внести корректирующие изменения в расшифрованные текстовые файлы, используя текстовый редактор операционной системы. А также возможность зашифровать текстовый файл, после того, как в него были внесены изменения. Логин и пароль необходимые для авторизации под именем администратора представляют собой две строковые константы: «work» - логин и «work\_asoiu» - пароль.

Теоретический материал представляет собой текстовый файл, содержащий текст, который разбит на страницы не менее 10 и не более 20 строк по 60 символов в строке, перенос осуществляется по словам, выравнивание по ширине. Текст в файле должен быть отформатирован вручную. Для разбития на страницы должен использоваться тег **[next\_page]** на отдельной строке. Для того, чтобы выделить цветом слово, необходимо поместить тег **[color]** перед словом и тег **[/color]** после слова.

Для визуализации алгоритма сортировки Шелла используется исходный массив элементов, значение  $d$  (расстояние между сортируемыми элементами), матрица состояний массива, в которой хранится состояние массива на каждом шагу сортировки, и матрица, содержащая дополнительную информацию о шаге: элементы  $I$  и  $J$ , сортируемые на данном шаге работы алгоритма, а также значение  $d$  и цвет выделения элементов (зелёным цветом выделяются сортируемые элементы, а жёлтым — те, что были отсортированы на этом шаге сортировки). Матрица состояний состоит из  $N$  столбцов, где  $N$  — количество элементов исходного массива, и  $M$  строк, где  $M$  — количество состояний массива (количество шагов сортировки). Соответственно матрица состояний и матрица, содержащая дополнительную информацию о шаге имеют одинаковое количество строк.

База вопросов представляет собой текстовый файл, который имеет следующую структуру: каждый вопрос занимает пять строк — на первой строке находится текст вопроса, на

оставшихся четырех — дистракторы, причем первый вариант ответа является правильным. В программе вопрос будет представлен в виде структуры, в которой хранятся строка с вопросом и четыре дистрактора. Дистрактор, в свою очередь, тоже является структурой и состоит из строки (вариата ответа), и переменной логического типа, которая определяет правильность данного ответа.

### 1.2.2 Алгоритм заполнения массива случайными числами

**Дано:**  $A[N]$  — массив целых чисел,  $N$  — размер массива;  $\max$ ,  $\min$  — границы генерации случайных чисел.

1. цел  $chislo$
  2. цел  $ostatok = (\max - \min) + 1$
  3. нц для  $i$  от 0 до  $N$ 
    - |     $chislo = \min + \text{rand}() \% (ostatok)$
    - |     $A[i] = chislo$
- кц

### 1.2.3 Алгоритм заполнения массива значениями с клавиатуры

**Дано:**  $A[N]$  — массив целых чисел,  $N$  — размер массива.

1. цел  $chislo$
  2. нц для  $i$  от 0 до  $N$ 
    - |    вывод "Для",  $i+1$ , "-го:"
    - |    ввод  $chislo$
    - |     $A[i] = chislo$
- кц

### 1.2.4 Алгоритм заполнения массива значениями из файла

**Дано:**  $A[N]$  — массив целых чисел,  $N$  — размер массива,  $filename$  — название файла

1. цел  $chislo$
2. Если файл существует то открыть файл (" $filename$ ")
  - |    нц для  $i$  от 0 до  $N$
  - |    ввод из файла  $chislo$ ;
  - |     $A[i] = chislo$
  - |    кц

Иначе вывод "Не удалось открыть файл"

Конец ветвления

### 1.2.5 Алгоритм сохранения состояния массива

**Дано:**  $A[N]$  – массив,  $N$  – количество элементов,  $States[M][N]$  – матрица состояний массива  $A$ ,  $M$  — количество состояний массива  $A$  (количество шагов сортировки),  $counterStates[M][4]$  – матрица, содержащая дополнительную информацию о шаге,  $d$  – расстояние между сортируемыми элементами,  $I$  – первый сортируемый элемент,  $J$  – второй сортируемый элемент,  $color$  – цвет (желтый или зеленый, в зависимости от ситуации).

1. Поместить  $A[N]$  в матрицу  $States[M][N]$  в качестве новой строки
2. Объявить  $temp[4]$  – массив целых чисел
3.  $temp[0] = d$
4.  $temp[1] = I$
5.  $temp[2] = J$
6.  $temp[3] = color$
7. Поместить  $temp[4]$  в матрицу  $counterStates[M][4]$

### 1.2.6 Алгоритм сортировки Шелла

**Дано:**  $A[N]$  – массив,  $N$  – количество элементов,  $States[M][N]$  – матрица состояний массива  $A$ ,  $M$  — количество состояний массива  $A$  (количество шагов сортировки),  $d$  – расстояние между сортируемыми элементами, функция 1.2.5 для сохранения состояния массива принимает параметры (значение  $d$ , первый элемент, второй элемент, цвет подсветки этих элементов).

1.  $d = N / 2$
2. сохранить состояние текущего массива и шага  $d$  в матрицу состояний массива  $States[M][N]$
3. нц пока ( $d > 0$ )
  - | нц для  $i$  от 0 до  $N-d$ 
    - | |  $j = i$
    - | | сохранить состояние текущего массива ( $d, j, j+d$ , зеленый цвет)
    - | | нц пока ( $j >= 0$  и  $A[j] > A[j+d]$ )
      - | | |  $temp = A[j]$
      - | | |  $A[j] = A[j+d]$
      - | | |  $A[j+d] = temp$
      - | | | сохранить состояние текущего массива ( $d, j, j+d$ , желтый цвет)
      - | | | уменьшить  $j$  на 1
    - | | | если шаг  $d = 1$  то
      - | | | | сохранить состояние текущего массива ( $d, j, j+d$ , зеленый цвет)
    - | | кв

| | кц

| кц

|  $d := 2;$

кц

### 1.2.7 Алгоритм вывода состояния массива

**Дано:**  $A[N]$  – массив,  $N$  – количество элементов,  $States[M][N]$  – матрица состояний массива  $A$ ,  $M$  – количество состояний массива  $A$  (количество шагов сортировки), **НомерСостояния** – номер состояния, которое необходимо вывести на экран.

нц для  $i$  от 0 до  $N$

| вывести  $M[\text{НомерСостояния}][i]$

кц

### 1.2.8 Алгоритм визуализации алгоритма сортировки Шелла

**Дано:**  $A[N]$  – массив,  $N$  – количество элементов, **НомерСостояния** — номер состояния, которое необходимо вывести на экран, **КоличествоСостояний** — общее количество шагов сортировки.

1. **НомерСостояния** = 0

2. нц пока пользователь не дал команду завершить процесс демонстрацию сортировки

| если пользователь запросил следующий шаг сортировки и **НомерСостояния** <

| **КоличествоСостояний**-1 то увеличить **НомерСостояния** на 1

| если пользователь запросил предыдущий шаг сортировки и **НомерСостояния** >

| 0 то уменьшить **НомерСостояния** на 1

| очистить экран и вывести исходный массив  $A$  (первое состояние) и текущий  $d$

| вывести состояние массива (**НомерСостояния**)

| выделить цветом сортируемые элементы  $A[j]$  и  $A[j+d]$

кц

### 1.2.9 Алгоритм шифрования файла

**Дано:** **Input** – входной файл с расширением .txt, **Output** – выходной файл с расширением .dat

нц пока (в файле **Input** есть строки с текстом)

| прочитать строку

| зашифровать строку

| записать строку в файл **Output**

кц

### 1.2.10 Алгоритм расшифровки файла

**Дано:** **Input** – входной файл с расширением .dat, **Output** – выходной файл с расширением .txt

нц пока (в файле **Input** есть строки с текстом)

- | прочитать строку
- | расшифровать строку
- | записать строку в файл **Output**

кц

### 1.2.11 Алгоритм вывода теоретического материала

**Дано:** *Page[M]* — страница, представляющая из себя массив строк, где *M* – количество строк.

*Text[N]* – массив страниц, где *N* – количество страниц в тексте, *crypt.dat* – зашифрованный файл с теорией, *num* – номер страницы.

1. нц пока (в зашифрованном файле есть строки с текстом)

- | прочитать строку
- | расшифровать строку
- | если строка = “*[next\_page]*” то
  - | | записать текущую страницу *Page[M]* в текст *Text[N]*
  - | | очистить текущую страницу *Page[M]*
- | иначе записать строку в страницу *Page[M]*

кц

2. нц пока пользователь не дал команду выйти в меню

- | если пользователь запросил следующую страницу и *num* < *N*-1 то *num* := *num* + 1
- | если пользователь запросил предыдущую страницу и *num* > 0 то *num* := *num* - 1
- | очистить экран и вывести страницу под номером *num*

кц

### 1.2.12 Алгоритм чтения базы вопросов из файла

**Дано:** *Answer* — структура с полями *text* (строковая константа, текст варианта ответа) и переменной логического типа *right* (определяет правильность ответа), *Question* — структура с полями *text* (строковая константа, текст вопроса) и *arr[4]* – массив с элементами типа *Answer*. *Questions[N]* – массив с элементами типа *Question*, *N* – количество вопросов в базе вопросов, *quest.dat* – зашифрованный файл с базой вопросов.

нц пока (в зашифрованном файле есть строки с текстом)

- | объявить переменную *temp* типа *Question*
- | нц для *i* от 0 до 4
  - | | прочитать строку
  - | | расшифровать строку
  - | | если *i* = 0 то записать эту строку в *temp* в качестве текста вопроса

```

|   |   иначе записать строку в temp в качестве варианта ответа под индексом i-1
|   кц
|   сохранить полученный вопрос temp в массив Questions[N]
кц

```

### 1.2.13 Алгоритм тестирования

**Дано:** *Answer* — структура с полями *text* (строковая константа, текст варианта ответа) и переменной логического типа *right* (определяет правильность ответа), *Question* — структура с полями *text* (строковая константа, текст вопроса) и *arr[4]* — массив с элементами типа *Answer*. *Questions[N]* — массив с элементами типа *Question*, *N* — количество вопросов в базе вопросов.

1. перемешать вопросы в массиве
2. перемешать варианты ответов в вопросах
3. цел *size* = 5
4. цел *i* = 0
5. *ch[size]* — массив целых чисел, заполненный значением -1
6. вывести инструкцию, как проходить тестирование
7. нц пока пользователь не дал команду завершить тестирование
  - | вывести на экран вопрос и четыре варианта ответа (вывести *Questions[i]*)
  - | если *ch[i]* = -1 то записать ответ пользователя в *ch[i]*
  - | если пользователь запросил следующий вопрос и *i < size - 1* то увеличить *i* на 1
  - | если пользователь запросил предыдущий вопрос и *i > 0* то уменьшить *i* на 1
 кц
8. подсчитать количество правильных, неправильных и пропущенных ответов
9. вывести результаты тестирования

### 1.2.14 Режим администратора

**Дано:** *ch* – выбор пользователем пункта меню

1. авторизация пользователя
  2. вывести условия меню администратора
  3. ввод *ch*
  4. вывести инструкцию по работе с файлами в режиме администратора
  5. если *ch* = 1 то открыть и расшифровать файл с теорией
  6. если *ch* = 2 то открыть и расшифровать файл с базой вопросов
  7. если *ch* = 3 то выйти в главное меню
- иначе зашифровать текстовый файл, а после удалить

### 1.2.15 Основной алгоритм

Дано:  $ch$  – выбор пользователем пункта меню

1. Вывести условия меню
2.  $ch = 0$
3. нц пока  $ch \neq 5$ 
  - | ввод  $ch$
  - | если  $ch = 1$  то вызвать алгоритм вывода теоретического материала 1.2.11
  - | если  $ch = 2$  то вызвать алгоритм визуализации алгоритма сортировки Шелла 1.2.8
  - | если  $ch = 3$  то вызвать алгоритм тестирования 1.2.13
  - | если  $ch = 4$  то вызвать алгоритм режим администратора 1.2.14
- кц

## 1.3 Входные и выходные данные

Входные данные для пользователя:

- Выбор пунктов меню.
- Выбор способа заполнения сортируемого массива.
- Целое число – выбор пользователя варианта ответа на вопрос во время тестирования.

Выходные данные для пользователя:

- Теоретический материал, предоставляемый пользователю для изучения.
- Состояние массива на каждом шаге сортировки Шелла
- Целое число – оценка, полученная в результате тестирования.

Входные данные для администратора:

- Выбор пунктов меню администратора.
- Логин и пароль, необходимые для авторизации
- Измененный расшифрованный текстовый файл, который необходимо зашифровать

Выходные данные для администратора: исходный расшифрованный текстовый файл.

## 1.4 Системные требования

Рекомендуемая конфигурация:

- Intel-совместимый процессор с частотой не менее 1,6 ГГц;
- не менее 512 МБ ОЗУ;
- не менее 20 МБ свободного места на диске;
- дисковод CD-ROM/DVD-ROM

Операционная система: Windows XP (x86) с пакетом обновления 3 (SP3) или более поздние. Среда разработки – интегрированная среда Code::Blocks 17.12, язык C++ (стандарт C++ 11 и более поздние).

## 2 РАБОЧИЙ ПРОЕКТ

### 2.1 Общие сведения о работе системы

Программный продукт разработан в интегрированной среде Code::Blocks (версия 17.12) на языке C++ (стандарт C++ 11 и более поздние), с использованием компилятора MinGW GCC (версия 8.2.0-3). Программа работает под управлением операционной системы Windows XP (x86) Professional (SP3) и более поздними.

### 2.2 Функциональное назначение программного продукта

Разработанный программный продукт предназначен для повышения качества знаний пользователей по теме «Сортировки Шелла», а также для снижения нагрузки на преподавателя. Программа имеет следующие функциональные возможности:

- предоставление пользователю теоретического материала по теме «Методы сортировок: сортировка Шелла»;
- визуализация работы алгоритма сортировки Шелла;
- тестирование пользователя по теме «Методы сортировок: сортировка Шелла»;
- предоставление пользователю режима администратора, который даёт возможность расшифровать и зашифровать файл с базой вопросов и файл с теорией;

Программа имеет следующие функциональные ограничения:

- элементы сортируемого элемента должны быть целыми числами из диапазона [-50; 50],
- количество элементов сортируемого массива должно быть не менее 5 и не более 15,
- программа не должна обеспечивать редактирование статистики результатов тестирования.

### 2.3 Инсталляция и выполнение программного продукта

Программу можно запустить с помощью проводника операционной системы, запустив скомпилированный исполняемый .exe файл.

Для выполнения программы необходимо:

1. Скопировать на жесткий диск компьютера папку coursework\_shell, содержащую исполняемый файл coursework\_shell.exe, необходимые файлы для функционирования программы (quest.dat, stats.dat, teo.dat), и папку source с исходными текстами программы.
2. Запустить исполняемый файл coursework\_shell.exe.
3. Сменить шрифт в настройках консольного окна на Lucida Console для отображения кириллицы и увеличить размер шрифта при необходимости (рекомендуется 16 или 18 пунктов).

## 2.4 Описание программы

В таблице 2.1 приведено описание структуры Answer, используемые в программе (модуль test.cpp и заголовочный файл test.h).

Таблица 2.1 – Описание структуры Answer

Поле	Тип	Назначение
text	string	Текст варианта ответа (дистрактора)
right	bool	Определяет правильность варианта ответа (дистрактора)

В таблице 2.2 приведено описание структуры Question, используемые в программе (модуль test.cpp и заголовочный файл test.h).

Таблица 2.2 – Описание структуры Question

Поле	Тип	Назначение
text	string	Текст вопроса
arr	array<Answer, 4>	4 варианта ответа на вопрос

В таблице 2.3 приведены функции и процедуры, используемые в программе (модуль main.cpp).

Таблица 2.3 – Функции и процедуры модуля main.cpp

Прототип	Назначение
void display_menu (const int &punkt)	Условия главного меню программы
void input_menu (int &punkt, bool &ready)	Процедура, необходимая для функционирования главного меню (выбор пункта меню путем считывания нажатой клавиши с клавиатуры)
void teo ()	Процедура, обеспечивающая переключение в режим чтения теории
void demo ()	Процедура, обеспечивающая переключение в режим демонстрации сортировки Шелла
void admin_menu ()	Процедура, обеспечивающая переключение в режим администратора
bool auth ()	Авторизация, необходимая для перехода в режим администратора
bool correct_symbols (const char &symbol)	Функция, проверяющая на корректность введенный символ в момент ввода пароля в авторизации
void display_admin_menu (const int &punkt)	Условия меню администратора

Продолжение таблицы 2.3

<b>Прототип</b>	<b>Назначение</b>
void input_admin_menu (int &punkt, bool &ready)	Процедура, необходимая для функционирования меню администратора (выбор пункта меню путем считывания нажатой клавиши с клавиатуры)
void ShowTitle()	Показывает информацию об авторе программы и о самой программе
void pre_crypt (int ch)	Процедура, занимающаяся расшифровкой и шифрованием файлов для режима администратора
void ShowAdminHelp()	Вывод инструкции для режима администратора
void gotoXY( short x, short y )	Перейти на нужную строку консоли (используется для изменения текста на экране с минимальными затратами ресурсов компьютера вместо очистки всего экрана)
void set_console_buffer_size(int Y)	Изменение размера буфера консольного окна
void SetColor(int textcolor, int backgroundcolor)	Установить цвет текста и цвет фона выделения текста
void hide_cursor(bool hide)	Спрятать курсор в консольном окне

В таблице 2.4 приведены функции и процедуры, используемые в программе (модуль teo.cpp).

Таблица 2.4 – Функции и процедуры модуля teo.cpp

<b>Прототип</b>	<b>Назначение</b>
void ShowTeoHelp()	Вывод инструкции для режима чтения теории
void creation_text (text &text)	Чтение текста из зашифрованного файла и формирование страниц
void one_page_display (const page &onepage)	Вывод одной страницы вместе с выделением цветом слов
void teo_display (text TEXT)	Вывод теоретического материала (текста)
void crypt (string filename)	Шифрование файла
void decrypt (string filename)	Расшифровка файла

В таблице 2.5 приведены функции и процедуры, используемые в программе (модуль demo.cpp).

Таблица 2.5 – Функции и процедуры модуля demo.cpp

Прототип	Назначение
void GenerateMas()	Заполнение массива случайными числами
void GenerateMasFromKeyboard()	Заполнение массива с клавиатуры
bool GenerateMasFromFile(string filename)	Заполнения массива из файла
void sposob_zapolneniya ()	Выбор способа заполнения массива
void Init(int input_N)	Инициализация массива (генерирование массива и его сортировка)
void SaveState(int d, int I, int J, int color)	Сохранение состояния массива (сохранение шага сортировки)
void Sort()	Сортировка массива
void PreviousStep()	Уменьшить шаг на 1
void NextStep()	Увеличить шаг на 1
void ShowDefaultState()	Показать изначальное состояния массива (до сортировки)
void ShowState()	Вывод состояния массива на определенном шаге сортировки
void Demonstrate(int input_N)	Демонстрация сортировки Шелла
void SaveToFile()	Сохранение результатов сортировки в файл
void ShowHelp()	Инструкция для режима сортировки

В таблице 2.6 приведены функции и процедуры, используемые в программе (модуль test.cpp).

Таблица 2.6 – Функции и процедуры модуля test.cpp

Прототип	Назначение
void ShowTestHelp()	Инструкция для режима тестирования
void add_stat (const vector<Question> &Questions, const vector<int> &ch, const int &right, const string &username)	Запись результата сортировки в зашифрованный файл со статистикой
void out_stats ()	Вывести содержимое файла со статистикой
void test ()	Переход в режим тестирования

Продолжение таблицы 2.6

<b>Прототип</b>	<b>Назначение</b>
void display_test_menu (const int &punkt)	Условия меню тестирования
void input_test_menu (int &punkt, bool &ready)	Процедура, необходимая для функционирования меню тестирования (выбор пункта меню путем считывания нажатой клавиши с клавиатуры)
void test_menu ()	Меню тестирования
void display_test_menu2 (const int &punkt)	Условия вспомогательного меню тестирования (меню, где предлагается посмотреть правильные ответы после тестирования)
bool test_menu2 ()	Вспомогательное меню тестирования
void creation_quest(vector<Question> &Questions)	Чтение базы вопросов из зашифрованного файла
int rand_answ (int ostatok)	Необходима для генерирования случайного индекса варианта ответа
void mixing_answers (vector <Question> &Questions)	Перемешивание порядка дистракторов
void mixing_questions (vector <Question> &Questions)	Перемешивание порядка вопросов
void display_question (const Question &question, const int &ch, const int &number, bool showright)	Вывод одного вопроса и подсветка ответа пользователя (в случае если showright = true, то и подсветка правильных ответов теста)
void test_results (int right, int wrong, int skip, unsigned int time)	Вывод результата тестирования
void input_test (int &ch, int &i, const int &size, bool &ready)	Ответ на вопрос путем нажатия цифр от 1 до 4 на клавиатуре
void check_answers (const vector<Question> &Questions, const vector<int> &ch, int &right, int &wrong, int &skip)	Подсчет результатов (количество правильных, неправильных ответов и пропущенных вопросов)
void display_test (const vector<Question> &Questions)	Тестирование (вывод вопросов, переключение между ними)

В таблице 2.7 приведены важнейшие переменные, используемые в программе.

Таблица 2.7 – Важнейшие переменные программы

Имя	Тип	Назначение
punkt_menu	int	Выбор пункта меню
ready	bool	Выбрал ли пункт пользователь (подтвердил ли выбор нажатием Enter на клавиатуре)
console	HANDLE	Дескриптор консоли (необходимо для выделения текста цветом, для перехода на заданную строку в консоли)
csbi	CONSOLE_SCREEN_BUFFER_INFO	Информация о свойствах буфера консольного окна (необходимо для изменения размера буфера консольного окна)

## 2.5 Разработанные меню и интерфейсы

После запуска программы на выполнение в консольном окне появится титульный лист (рис. 2.1), который содержит информацию о теме курсового проекта и об авторе программы.

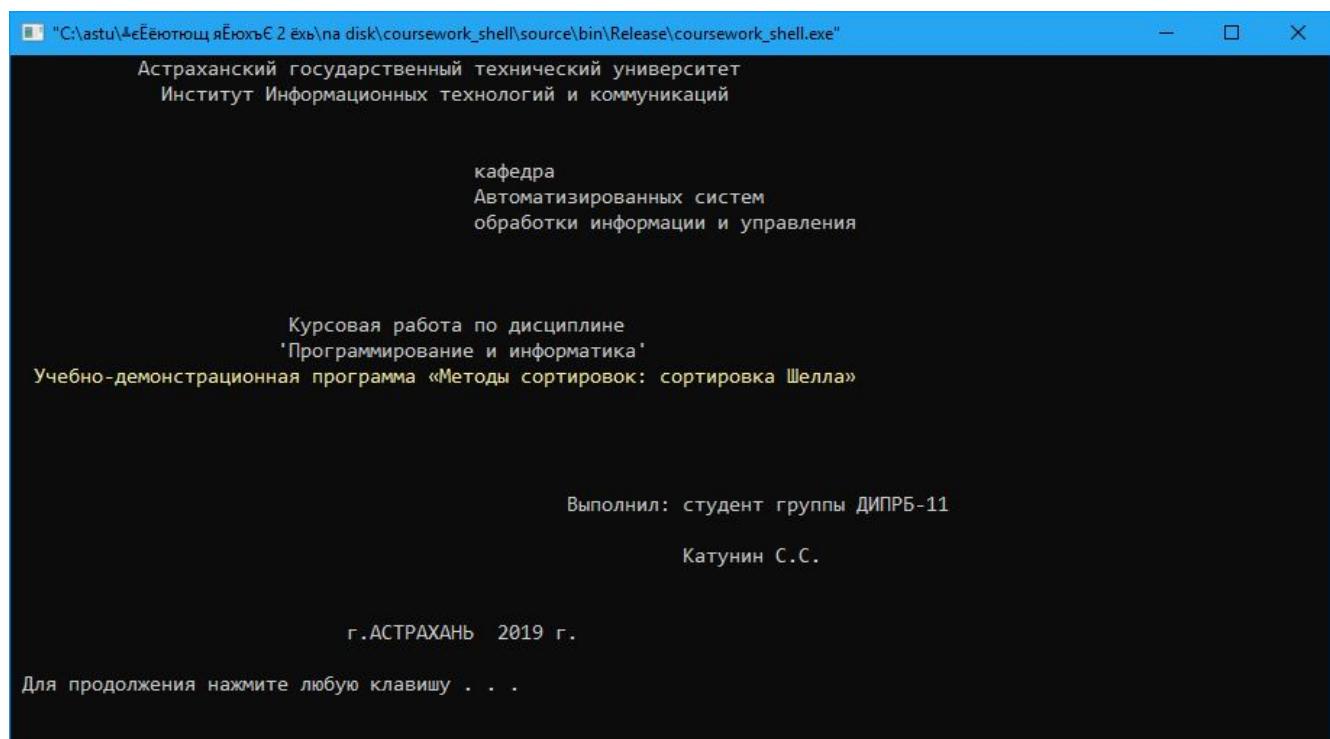


Рисунок 2.1 – Консоль программы с титульным листом

При нажатии любой клавиши в консольном окне появится меню (рис. 2.2), которое позволяет перейти в режим чтения теории, режим демонстрации сортировки Шелла, режим тестирования и режим администратора либо выйти из программы.

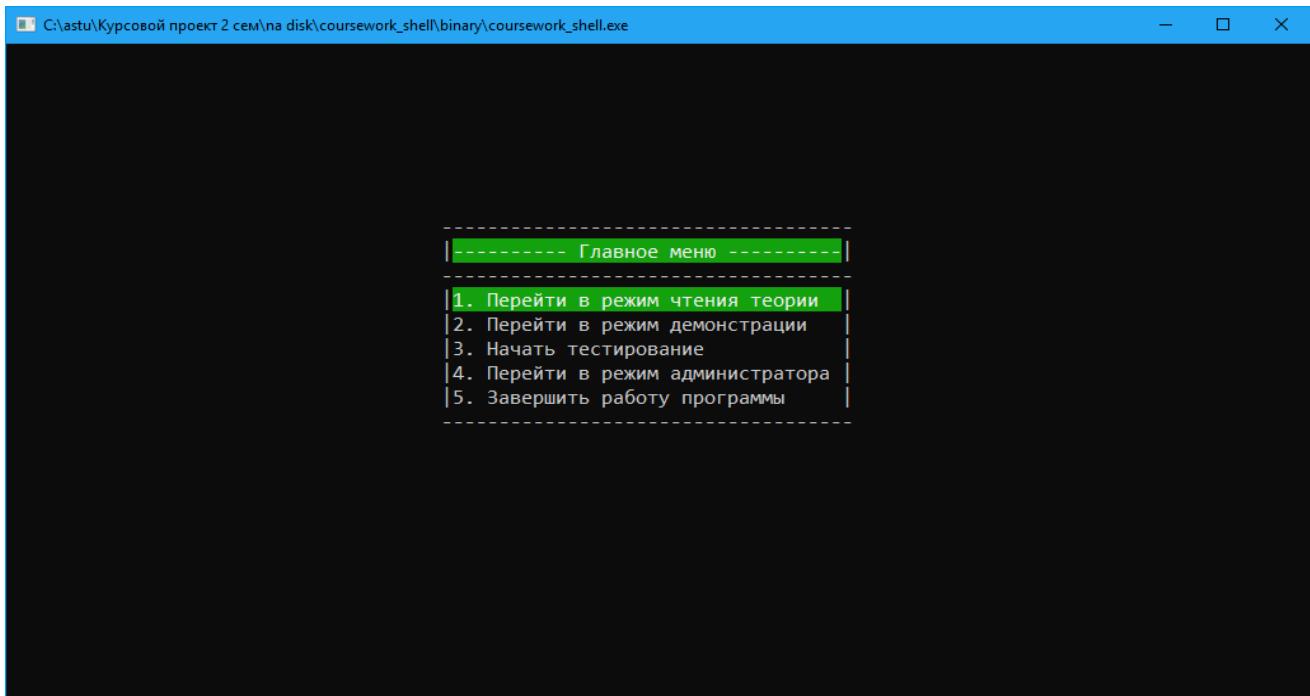


Рисунок 2.2 – Консоль программы с главным меню

При нажатии клавиши «Enter» или «1» пользователю будет представлена инструкция по работе в режиме чтения теории (рис. 2.3).

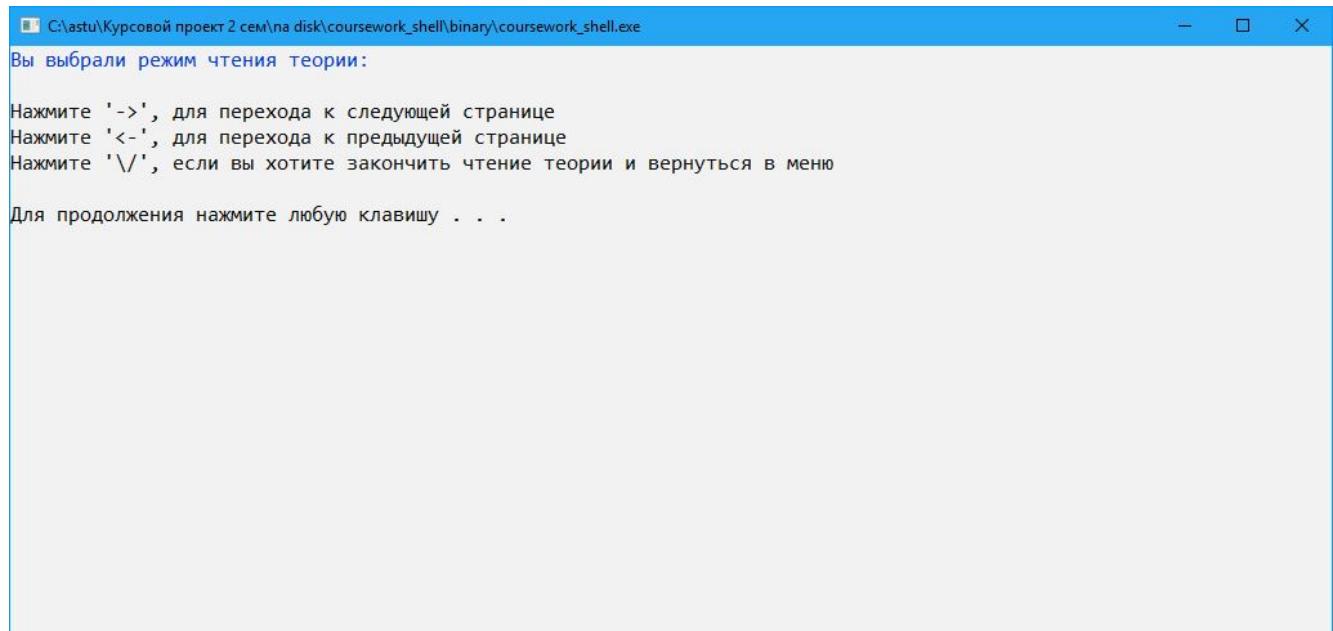


Рисунок 2.3 – Инструкция по работе в режиме чтения теории

После нажатия любой клавиши в консоли появится первая страница текста теории (рис. 2.4).

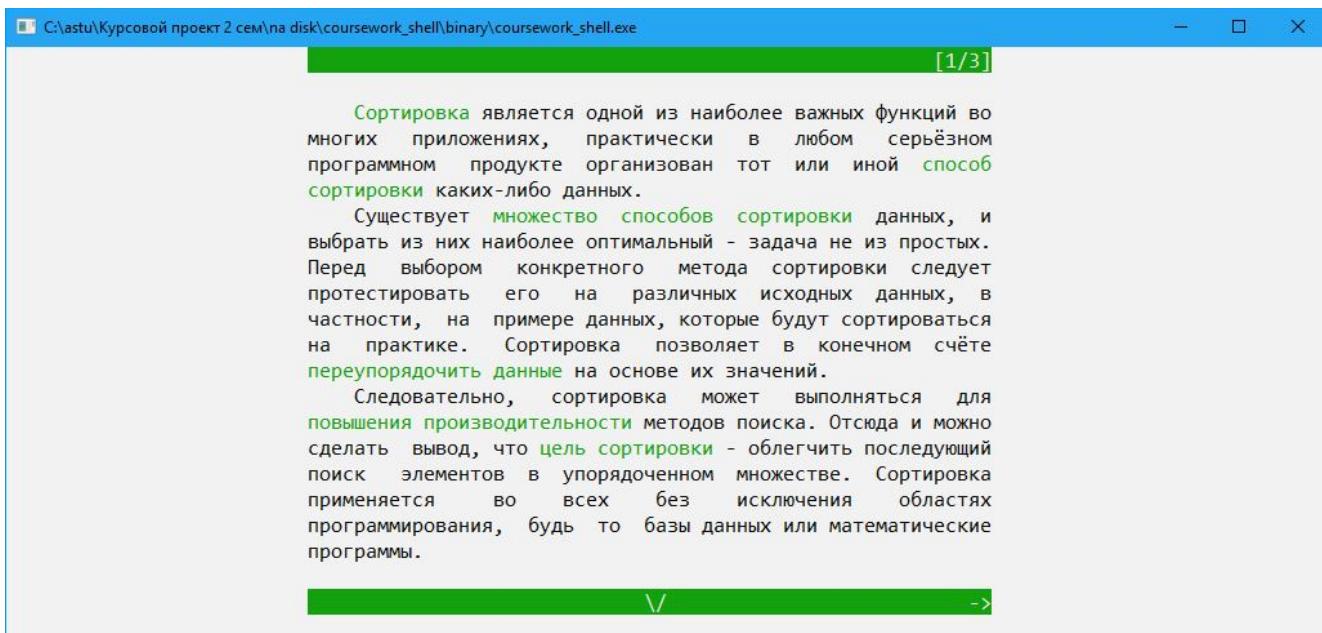


Рисунок 2.4 – Страница теоретического материала

При помощи нажатия на клавишу «стрелка вправо» осуществляется переход к следующей странице, а при помощи нажатия на клавишу «стрелка влево» – переход к предыдущей.

При нажатии клавиши «стрелка вниз» происходит возврат в главное меню (рис. 2.2). Если выбрать второй пункт меню — выводится инструкция по работе в режиме демонстрации сортировки Шелла и запрашивается размер N сортируемого массива (рис. 2.5).

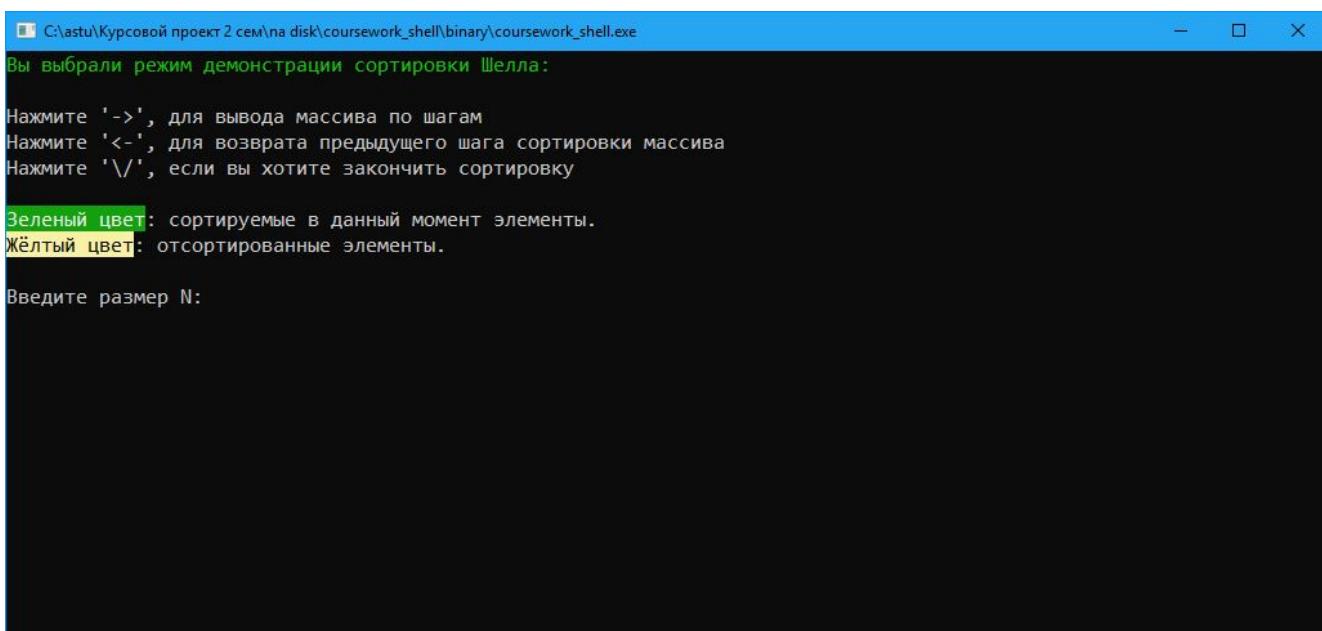


Рисунок 2.5 – Инструкция по работе в режиме демонстрации сортировки

После ввода размера N пользователю предлагается выбрать способ заполнения (рис. 2.6).

```
Вы выбрали режим демонстрации сортировки Шелла:

Нажмите '->', для вывода массива по шагам
Нажмите '<-', для возврата предыдущего шага сортировки массива
Нажмите '\V', если вы хотите закончить сортировку

Зеленый цвет: сортируемые в данный момент элементы.
Жёлтый цвет: отсортированные элементы.

*** Выбор способа заполнения ***
1. Заполнение с клавиатуры.
2. Заполнение случайными числами.
3. Заполнение из файла.
Выберите способ заполнения:
```

Рисунок 2.6 – Выбор способа заполнения сортируемого массива

После того, как пользователь выбрал способ заполнения массива, массив выводится на экран (рис. 2.7).

```
Вы выбрали режим демонстрации сортировки Шелла:

Нажмите '->', для вывода массива по шагам
Нажмите '<-', для возврата предыдущего шага сортировки массива
Нажмите '\V', если вы хотите закончить сортировку

Зеленый цвет: сортируемые в данный момент элементы.
Жёлтый цвет: отсортированные элементы.

Исходный массив:
9 45 7 -36 41 48 -38 1 -22 14

Текущий шаг d: 5

Текущий массив:
9 45 7 -36 41 48 -38 1 -22 14
```

Рисунок 2.7 – Исходный массив

При помощи нажатия на клавишу «стрелка вправо» осуществляется переход к следующему шагу сортировки. Сортируемые элементы подсвечиваются зеленым цветом (рис. 2.8).

```
C:\astu\Курсовой проект 2 сем\на disk\coursework_shell\binary\coursework_shell.exe

Вы выбрали режим демонстрации сортировки Шелла:

Нажмите '->', для вывода массива по шагам
Нажмите '<-', для возврата предыдущего шага сортировки массива
Нажмите '\V', если вы хотите закончить сортировку

Зеленый цвет: сортируемые в данный момент элементы.
Жёлтый цвет: отсортированные элементы.

Исходный массив:
9 45 7 -36 41 48 -38 1 -22 14

Текущий шаг d: 5

Текущий массив:
9 45 7 -36 41 48 -38 1 -22 14
```

Рисунок 2.8 – Подсветка сортируемых элементов

Если элементы были отсортированы на данном шаге сортировки, то элементы подсвечиваются желтым цветом (рис. 2.9).

```
C:\astu\Курсовой проект 2 сем\на disk\coursework_shell\binary\coursework_shell.exe
Вы выбрали режим демонстрации сортировки Шелла:

Нажмите '->', для вывода массива по шагам
Нажмите '<-', для возврата предыдущего шага сортировки массива
Нажмите '/\\', если вы хотите закончить сортировку

Зеленый цвет: сортируемые в данный момент элементы.
Жёлтый цвет: отсортированные элементы.

Исходный массив:
9 45 7 -36 41 48 -38 1 -22 14

Текущий шаг d: 5

Текущий массив:
9 -38 7 -36 41 48 45 1 -22 14
```

Рисунок 2.9 – Подсветка отсортированных элементов на данном шаге сортировки

В процессе сортировки меняется значение шага d. Когда значение шага d будет равно 1, будет выведено соответствующее сообщение о переходе к сортировке вставками (рис. 2.10).

```
C:\astu\Курсовой проект 2 сем\на disk\coursework_shell\binary\coursework_shell.exe
Вы выбрали режим демонстрации сортировки Шелла:

Нажмите '->', для вывода массива по шагам
Нажмите '<-', для возврата предыдущего шага сортировки массива
Нажмите '/\\', если вы хотите закончить сортировку

Зеленый цвет: сортируемые в данный момент элементы.
Жёлтый цвет: отсортированные элементы.

Исходный массив:
9 45 7 -36 41 48 -38 1 -22 14

Так как значение d = 1, переходим к сортировке вставками.

Текущий массив:
-38 -36 1 9 7 14 -22 41 45 48
```

Рисунок 2.10 – Сообщение о переходе к сортировке вставками

Когда массив будет отсортирован и сортировка будет закончена — на экран выведется соответствующее сообщение (рис. 2.11).

C:\astu\Курсовой проект 2 сем\на disk\coursework\_shell\binary\coursework\_shell.exe

Вы выбрали режим демонстрации сортировки Шелла:

Нажмите '>', для вывода массива по шагам  
Нажмите '<', для возврата предыдущего шага сортировки массива  
Нажмите '\V', если вы хотите закончить сортировку

Зеленый цвет: сортируемые в данный момент элементы.  
Жёлтый цвет: отсортированные элементы.

Исходный массив:  
9 45 7 -36 41 48 -38 1 -22 14

Так как значение d = 1, переходим к сортировке вставками.

Текущий массив:  
-38 -36 -22 1 7 9 14 41 45 48

Массив отсортирован (Array is sorted)

Нажмите 's', если вы хотите сохранить в файл все шаги сортировки  
Нажмите '\V' для возврата в Главное меню...

Рисунок 2.11 – Сообщение о завершенной сортировке

При нажатии на клавишу «стрелка вниз» произойдет возврат в главное меню (рис. 2.2).  
При выборе третьего пункта меню откроется меню тестирования (рис. 2.12).

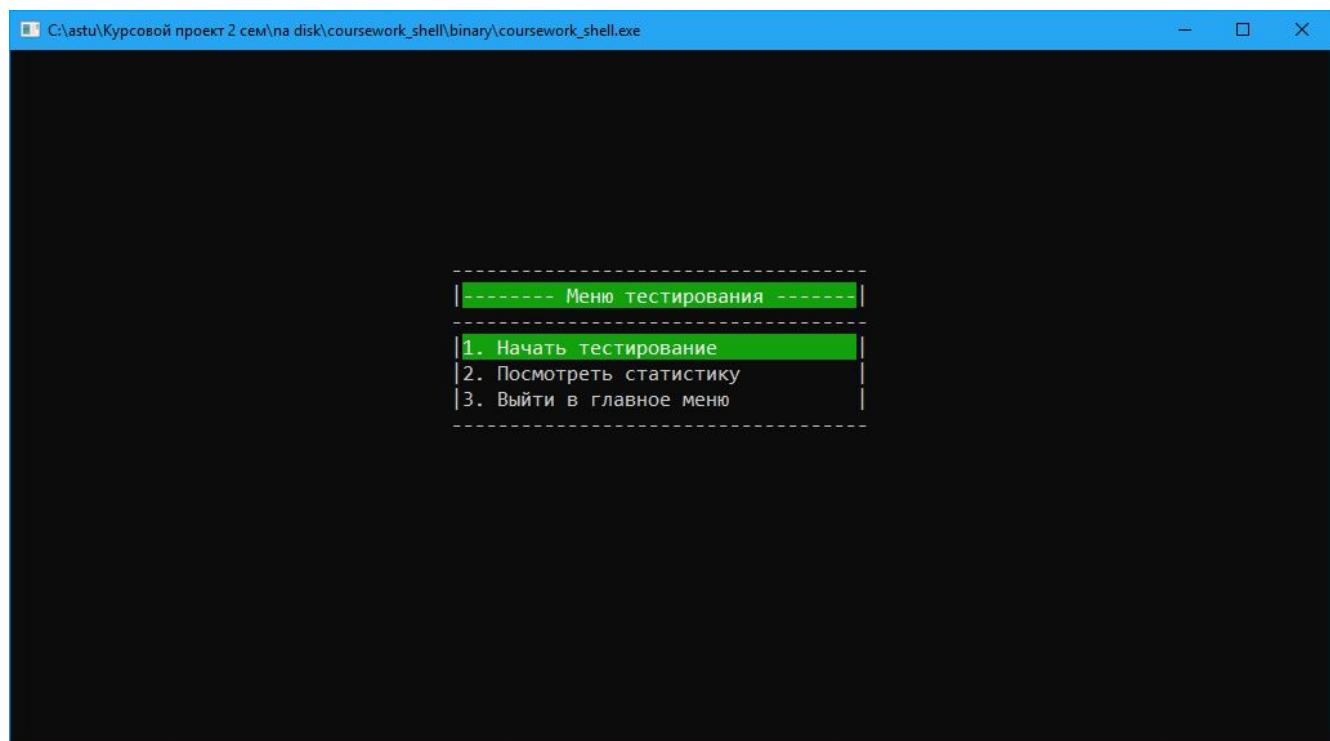
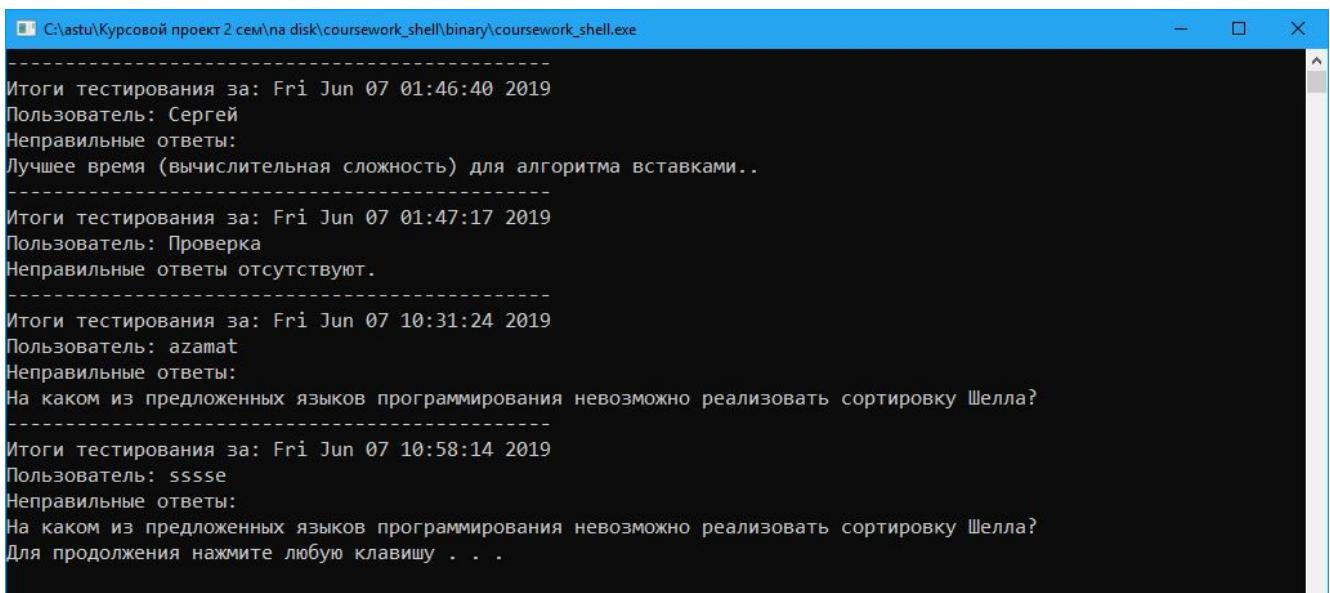


Рисунок 2.12 – Меню тестирования

При выборе второго пункта меню открывается содержимое файла со статистикой результатов тестирования (рис. 2.13).



```
C:\astu\Курсовой проект 2 сем\на disk\coursework_shell\binary\coursework_shell.exe

Итоги тестирования за: Fri Jun 07 01:46:40 2019
Пользователь: Сергей
Неправильные ответы:
Лучшее время (вычислительная сложность) для алгоритма вставками..

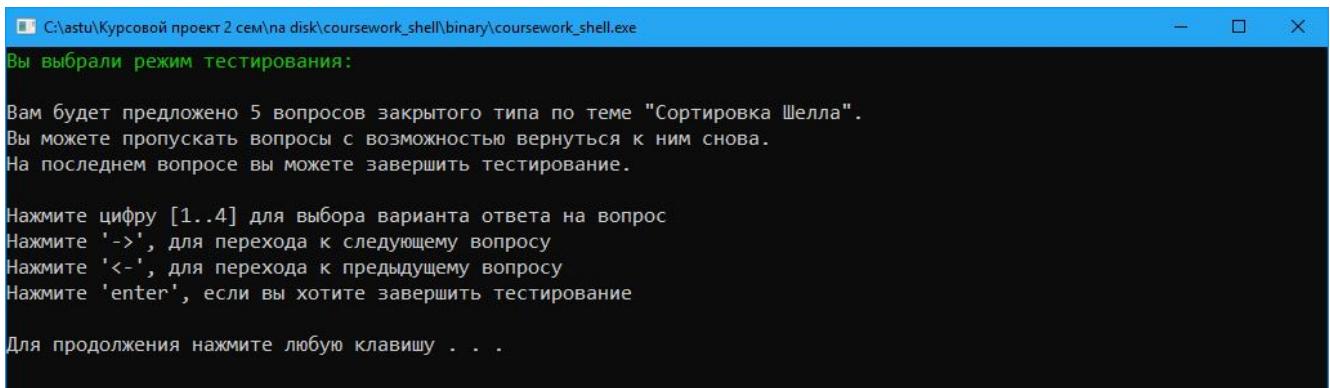
Итоги тестирования за: Fri Jun 07 01:47:17 2019
Пользователь: Проверка
Неправильные ответы отсутствуют.

Итоги тестирования за: Fri Jun 07 10:31:24 2019
Пользователь: azamat
Неправильные ответы:
На каком из предложенных языков программирования невозможно реализовать сортировку Шелла?

Итоги тестирования за: Fri Jun 07 10:58:14 2019
Пользователь: sssse
Неправильные ответы:
На каком из предложенных языков программирования невозможно реализовать сортировку Шелла?
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2.13 – Статистика результатов тестирования

После нажатия любой клавиши осуществляется возврат в меню тестирования (рис. 2.12). При нажатии первого пункта меню на экран будет выведена инструкция по работе в режиме тестирования (рис. 2.14).



```
C:\astu\Курсовой проект 2 сем\на disk\coursework_shell\binary\coursework_shell.exe

Вы выбрали режим тестирования:

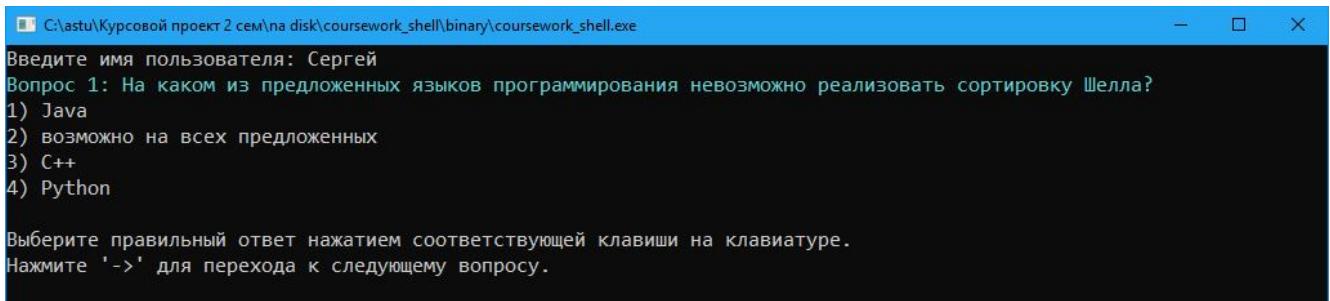
Вам будет предложено 5 вопросов закрытого типа по теме "Сортировка Шелла".
Вы можете пропускать вопросы с возможностью вернуться к ним снова.
На последнем вопросе вы можете завершить тестирование.

Нажмите цифру [1..4] для выбора варианта ответа на вопрос
Нажмите '->', для перехода к следующему вопросу
Нажмите '<-', для перехода к предыдущему вопросу
Нажмите 'enter', если вы хотите завершить тестирование

Для продолжения нажмите любую клавишу . . .
```

Рисунок 2.14 – Инструкция по работе в режиме тестирования

После нажатия любой клавиши начнется тестирование (рис. 2.15).



```
C:\astu\Курсовой проект 2 сем\на disk\coursework_shell\binary\coursework_shell.exe

Введите имя пользователя: Сергей
Вопрос 1: На каком из предложенных языков программирования невозможно реализовать сортировку Шелла?
1) Java
2) возможно на всех предложенных
3) C++
4) Python

Выберите правильный ответ нажатием соответствующей клавиши на клавиатуре.
Нажмите '->' для перехода к следующему вопросу.
```

Рисунок 2.15 – Тестирование

При выборе правильного ответа (в данном случае пункт 2), он выделится зеленым цветом (рис. 2.16).

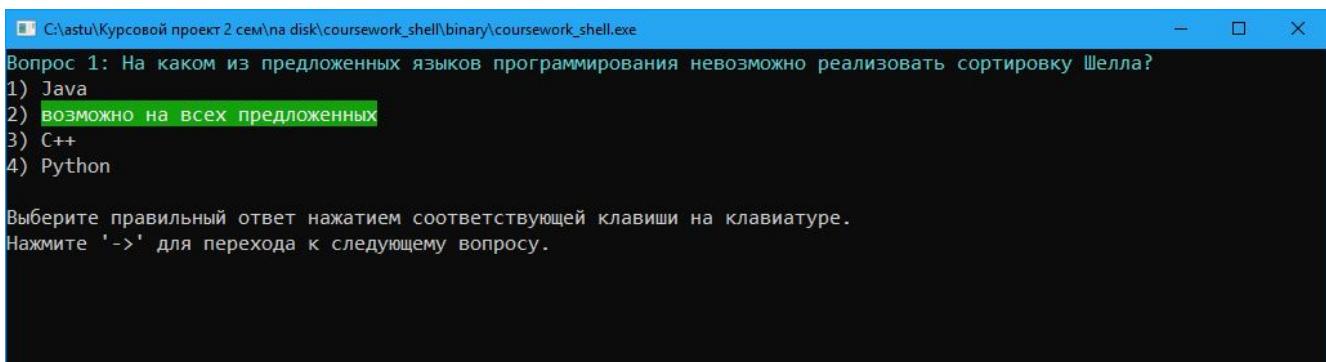


Рисунок 2.16 – Правильный ответ на вопрос

А при выборе неправильного ответа на вопрос, этот вариант выделится красным цветом (рис. 2.17).

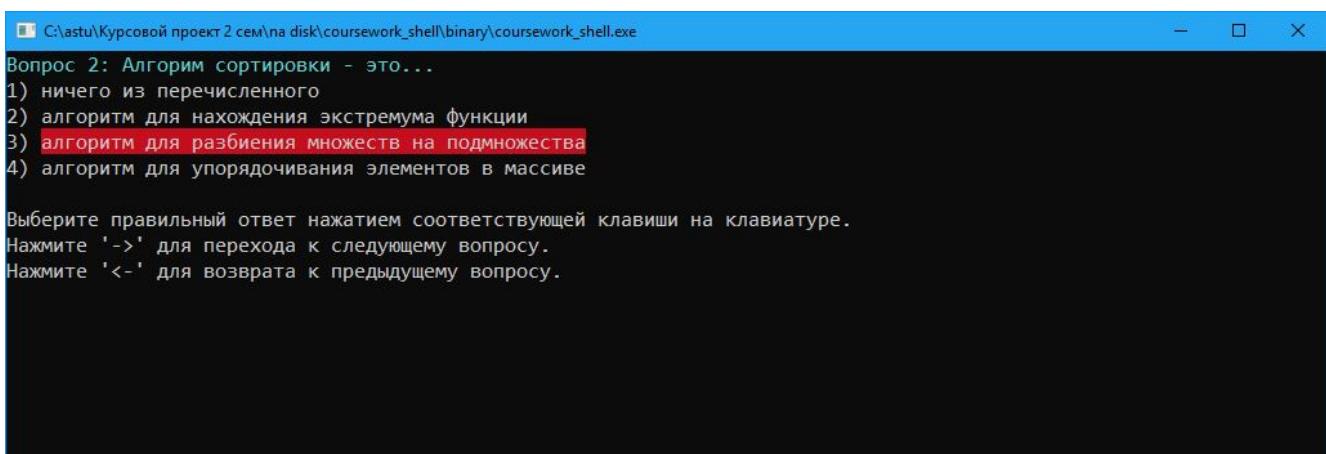


Рисунок 2.17 – Неправильный ответ на вопрос

В конце тестирования на экран будут выведены результаты (рис. 2.18).

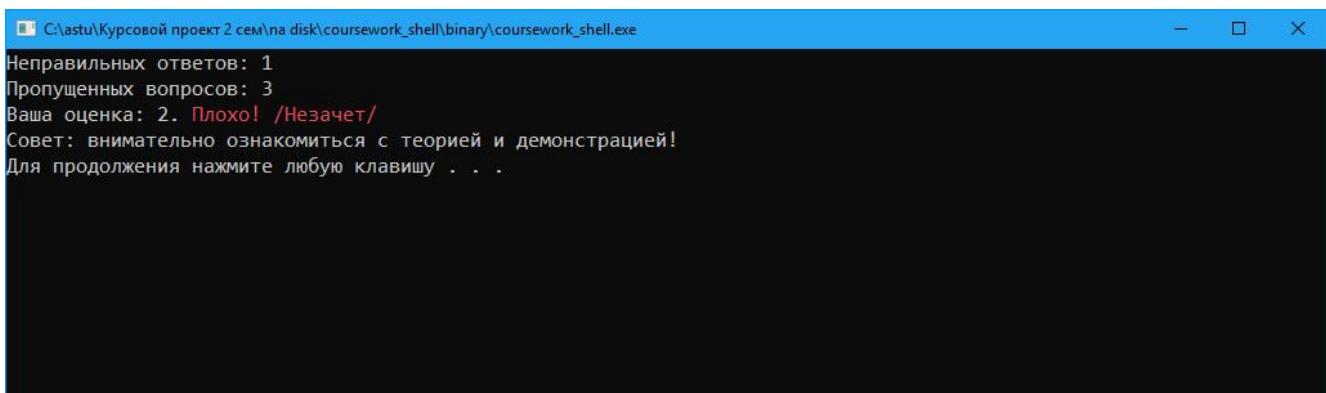


Рисунок 2.18 – Результаты тестирования

После пройденного тестирования на экране будет отображено меню, где можно будет выбрать вариант просмотра правильных ответов, а можно выйти в главное меню (рис. 2.19).

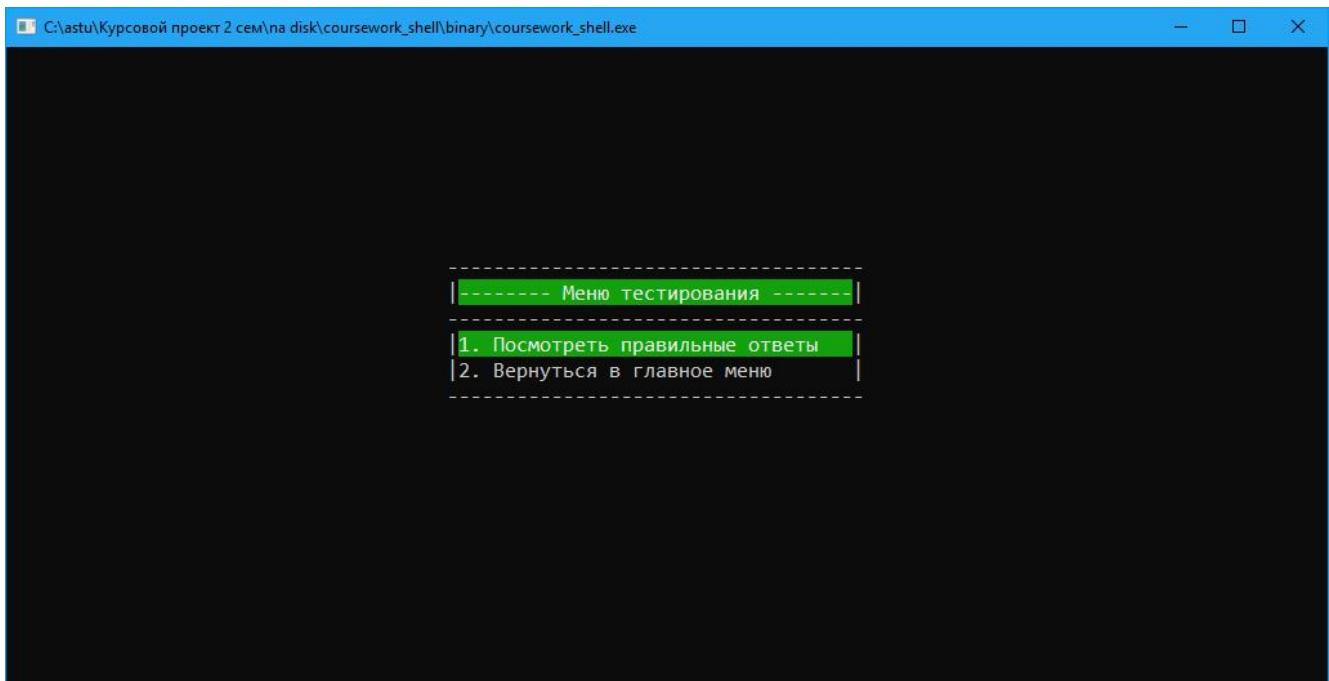


Рисунок 2.19 – Вспомогательное меню тестирования

При выборе пункта 2 осуществляется выход в главное меню (рис 2.2). Если в главном меню выбрать пункт 5 начнется процесс авторизации (ввод логина и пароля администратора) (рис. 2.20).

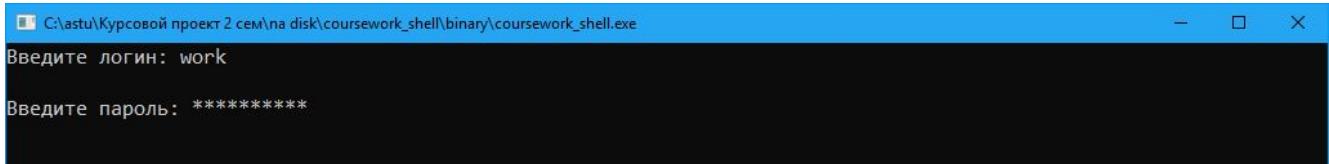


Рисунок 2.20 – Авторизация

После правильно введенного логина и пароля пользователю будет представлена инструкция по работе в режиме администратора (рис. 2.21).

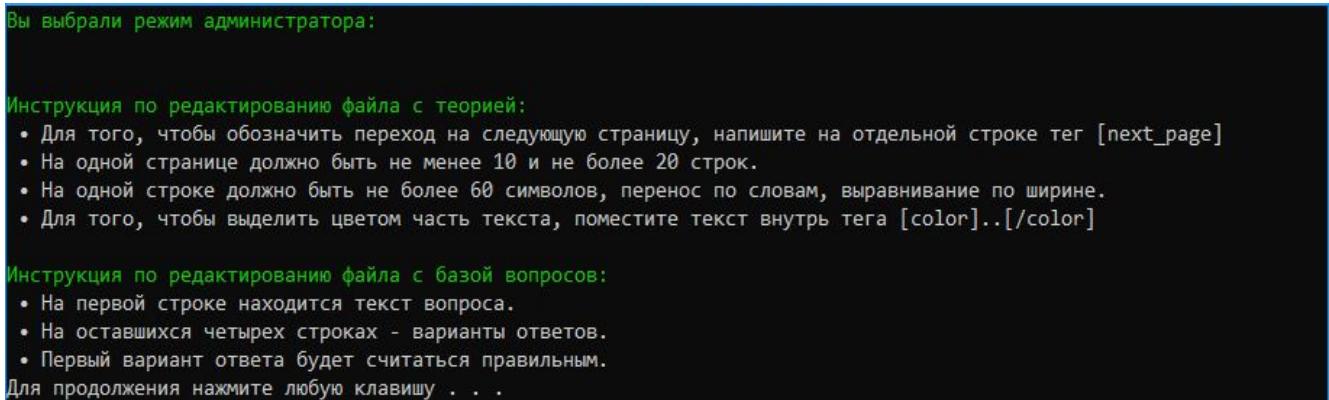


Рисунок 2.21– Инструкция по работе в режиме администратора

После нажатия любой клавиши откроется меню администратора (рис. 2.22).

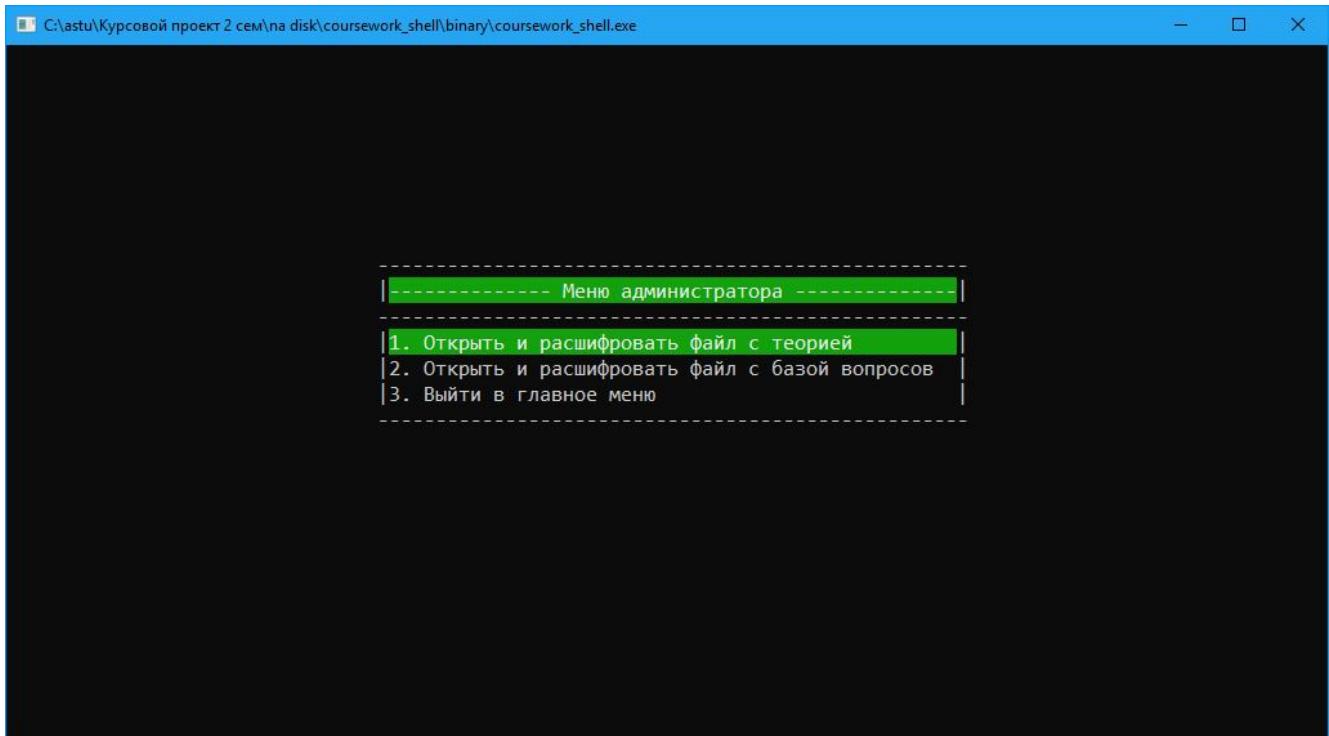


Рисунок 2.22 – Меню администратора

При выборе пункта 1 начнется процесс расшифровки файла (рис. 2.23), а также откроется текстовый файл (рис. 2.24).



Рисунок 2.23 – Расшифровка файла

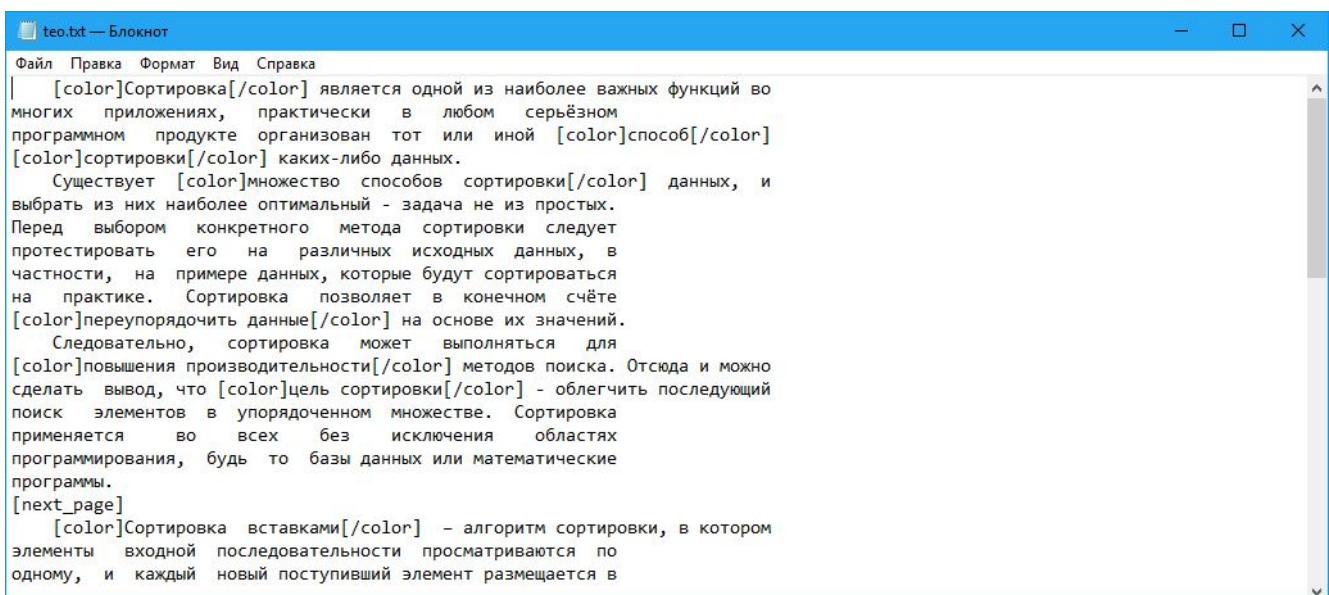


Рисунок 2.24 – Расшифрованный текстовый файл

После того, как пользователь сохранит и закроет текстовый документ, программа автоматически внесет изменения в зашифрованный файл и удалит текстовый, а после перейдет в главное меню (рис. 2.2). После выбора 5 пункта меню программа завершит свою работу.

## 2.6 Сообщения системы

В таблице 2.8 приведены сообщения системы.

Таблица 2.8 – Сообщения системы

№ п\п	Сообщение	Причина возникновения, способ устранения
1	Массив отсортирован (Array is sorted)	Алгоритм сортировки закончил свою работу
2	Ошибка!	Пользователь ввел недопустимый символ
3	Расшифровываю файл ...	Происходит процесс расшифровки файла в режиме администратора
4	Файл был успешно расшифрован и сохранен под именем ...	Процесс расшифровки завершился успешно
5	Неправильных ответов: .. Пропущенных вопросов: ... Ваша оценка: ..	Итоги тестирования
6	Произведено сохранение всех шагов сортировки в файл sortsteps.txt	Пользователь решил сохранить все шаги сортировки в файл

В случае появления других сообщений следует обратиться к разработчику.

### 3 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

#### 3.1 Проверка работоспособности режима чтения теории

1. Запустить программу на выполнение. Появится титульный лист (см. рис. 2.1).
2. Нажать любую клавишу, убедиться, что на экране появилось меню (см. рис. 2.2.)
3. Выбрать пункт 1, убедиться, что открылась инструкция (см. рис.2.3).
4. Нажать любую клавишу, убедиться, что на экране отображена первая страница теоретического материала (см. рис. 2.4).
5. Нажать на клавишу «стрелка вправо», убедиться, что программа отобразила вторую страницу (см. рис. 2.5).
6. Снова нажать клавишу «стрелка вправо», убедиться, что программа отобразила третью страницу (см. рис. 2.6).
7. Повторить пункт 5 и убедиться, что содержимое консольного окна не изменилось.
8. Нажать на клавишу «стрелка влево», убедиться, что программа отобразила вторую страницу (см. рис. 2.5).
9. Снова нажать на клавишу «стрелка влево», убедиться, что программа отобразила первую страницу (см. рис. 2.4).
10. Нажать клавишу «стрелка вниз», убедиться, что произошел возврат в главное меню (см. рис. 2.2).
11. Выбрать пункт 5, убедиться, что программа завершила свою работу.

#### 3.2 Проверка работоспособности режима демонстрации

1. Запустить программу на выполнение. Появится титульный лист (см. рис. 2.1).
2. Нажать любую клавишу, убедиться, что на экране появилось меню (см. рис. 2.2.)
3. Выбрать пункт 2, убедиться, что открылась инструкция (см. рис. 2.5).
4. Ввести некорректный размер N (меньше 5 или больше 15), убедиться, что программа вывела сообщение 2 (см. табл. 2.8) и запросила повторный ввод.
5. Ввести корректный размер N, убедиться, что программа предложила варианты заполнения массива (см. рис. 2.6).
6. Выбрать заполнение случайными числами, убедиться, что на экране выведен массив (см. рис. 2.7).
7. Нажать клавишу «стрелка вправо», убедиться, что программа подсвечивает зеленым цветом сортируемые элементы (см. рис. 2.8).

8. Нажимать клавишу «стрелка вправо» пока первый элемент меньше второго, после этого снова нажать клавишу «стрелка вправо», убедиться, что программа меняет элементы местами и подсвечивает их желтым цветом (см. рис. 2.9).
9. Повторять пункты 6 и 7 пока сортировка не будет закончена и не выведется сообщение 1 (см. табл. 2.8 и рис. 2.11).
10. Нажать на клавишу «s», убедиться, что программа сохранила все шаги сортировки в файл SortSteps.txt и вывела сообщение 6 (см. табл. 2.8).
11. Нажать клавишу «стрелка вниз», убедиться, что произошел возврат в главное меню (см. рис. 2.2).
12. Повторить пункт 3, 4, 5, выбрать другой способ заполнения массива, повторить пункты 7, 8, 9, 11.
13. Повторить пункт 12.
14. Выбрать пункт 5, убедиться, что программа завершила свою работу.

### 3.3 Проверка работоспособности режима тестирования

1. Запустить программу на выполнение. Появится титульный лист (см. рис. 2.1).
2. Нажать любую клавишу, убедиться, что на экране появилось меню (см. рис. 2.2.)
3. Выбрать пункт 3, убедиться, что открылось меню тестирования (см. рис. 2.14).
4. Выбрать пункт 2, убедиться, что на экран вывелась статистика результатов тестирования (см. рис. 2.13).
5. Нажать любую клавишу, убедиться, что произошел возврат в меню тестирования (см. рис. 2.12).
6. Выбрать пункт 1, убедиться, что на экран вывелась инструкция (см. рис. 2.14).
7. Нажать любую клавишу, убедиться, что программа предлагает ввести имя тестируемого.
8. Ввести имя, убедиться, что на экране отображен первый вопрос тестирования. (см. рис. 2.15).
9. Правильно ответить на вопрос, убедиться, что выбранный дистрактор выделен зеленым цветом (см. рис. 2.16).
10. Неправильно ответить на вопрос, убедиться, что выбранный дистрактор выделен красным цветом (см. рис. 2.17).
11. Завершить тестирование, убедиться, что было выведено сообщение 5 (см. табл. 2.8 и рис. 2.18).
12. Нажать любую клавишу, убедиться, что открылось меню с возможностью посмотреть правильные ответы. (см. рис. 2.19).

13. Выбрать 1 пункт и убедиться, что программа выводит все 5 вопросов и выделяет зеленым цветом правильные ответы на вопросы.
14. Дойти до 5 вопроса и нажать Enter, убедиться, что произошел выход в меню тестирования. (см. рис. 2.12).
15. Выбрать пункт 2, убедиться, что в статистику была добавлена новая запись с введенным именем (пункт 8).
16. Нажать любую клавишу, убедиться, что произошел возврат в меню тестирования. (см. рис. 2.12.)
17. Повторить пункты 6, 7, 8, 9, 10, 11, 12, 13, 14, 16 несколько раз, таким образом, чтобы получить оценки 5, 4, 3 и 2 в сообщении 5 (см. табл. 2.8 и рис. 2.18).
18. Выбрать 3 пункт, убедиться, что произошел возврат в главное меню (см. рис 2.2).
19. Выбрать пункт 5, убедиться, что программа завершила свою работу.

### 3.4 Проверка работоспособности режима администратора

1. Запустить программу на выполнение. Появится титульный лист (см. рис. 2.1).
2. Нажать любую клавишу, убедиться, что на экране появилось меню (см. рис. 2.2).
3. Выбрать пункт 4, убедиться, что началась авторизация, и программа запросила логин (см. рис. 2.3).
4. Ввести логин «work» и пароль «work\_asoiu» (см.рис. 2.20), убедиться, что на экране отобразилась инструкция для администратора (см. рис 2.23).
5. Нажать любую клавишу, убедиться, что программа отобразила меню администратора (см. рис. 2.22).
6. Выбрать 1 пункт, убедиться, что программа начала расшифровывать файл teo.dat и открыла текстовый файл teo.txt (см. рис. 2.23 и 2.24), а также вывела сообщения 3 и 4 (см. табл. 2.8).
7. Внести изменения, изменив слово на первой странице текста, сохранить, закрыть текстовый файл, убедиться, что произошел возврат в меню администратора.
8. Выбрать пункт 3, убедиться, что произошел возврат в главное меню (см. рис. 2.2).
9. Выбрать пункт 1, убедиться, что открылась инструкция (см. рис.2.3).
10. Нажать любую клавишу, убедиться, что на экране отображена первая страница теоретического материала и она изменилась в соответствии с пунктом 7 (см. рис. 2.4).
11. Нажать клавишу «стрелка вниз», убедиться, что произошел возврат в главное меню (см. рис. 2.2).

12. Повторить пункт 3, 4, 5. Выбрать 2 пункт, убедиться, что программа начала расшифровывать файл quest.dat и открыла текстовый файл quest.dat (см. рис. 2.23 и рис. 2.24), а также вывела сообщения 3 и 4 (см. табл. 2.8).
13. Внести изменения, немного изменив текст вопросов или дистракторов, сохранить, закрыть текстовый файл, убедиться, что произошел возврат в меню администратора.
14. Повторить пункт 8. Выбрать пункт 3, убедиться, что открылось меню тестирования (см. рис. 2.14).
15. Выбрать пункт 1, убедиться, что на экран вывелась инструкция (см. рис. 2.14).
16. Нажать любую клавишу, убедиться, что программа предлагает ввести имя тестируемого.
17. Ввести имя, убедиться, что на экране отображен первый измененный вопрос тестирования в соответствии с пунктом 13. (см. рис. 2.15).
18. Завершить тестирование, убедиться, что было выведено сообщение 5 (см. табл. 2.8 и рис. 2.18).
19. Выбрать пункт 2, убедиться, что произошел возврат в меню тестирования (см. рис. 2.12).
20. Выбрать 3 пункт, убедиться, что произошел возврат в главное меню (см. рис 2.2).
21. Повторить пункт 3, ввести неправильные логин и пароль, убедиться, что произошел возврат в главное меню (см. рис 2.2).
22. Выбрать пункт 5, убедиться, что программа завершила свою работу.

## **ЗАКЛЮЧЕНИЕ**

В результате курсового проектирования разработана учебно-демонстрационная программа «Методы сортировок: сортировка Шелла». Программа предоставляет теоретический материал по теме «Методы сортировок: сортировка Шелла», визуализирует работу алгоритма сортировки Шелла, позволяет пройти тестирование, по окончании которого выдаётся сообщение о том, насколько успешно был выполнен тест.

Программа отвечает поставленным требованиям и может быть использована для обучения студентов, обучающихся программированию.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Златопольский Д.М. Программирование: типовые задачи, алгоритмы, методы 2-е изд. – М.: БИНОМ. Лаборатория знаний, 2012. – 223 с.: ил.
2. Лаптев В.В. С++. Экспресс-курс. – СПб.: БХВ-Петербург, 2004. – 512 с.: ил.
3. Сортировка Шелла – [Электронный ресурс] режим доступа:  
[https://ru.wikipedia.org/wiki/Сортировка\\_Шелла](https://ru.wikipedia.org/wiki/Сортировка_Шелла) (04.03.2019)
4. Сортировка Шелла – [Электронный ресурс] режим доступа: <http://kvodo.ru/sortirovka-shella.html> (04.03.2019)
5. Сортировка вставками – [Электронный ресурс] режим доступа:  
[https://ru.wikipedia.org/wiki/Сортировка\\_вставками](https://ru.wikipedia.org/wiki/Сортировка_вставками) (04.03.2019)
6. Алгоритм сортировки – [Электронный ресурс] режим доступа:  
[https://ru.wikipedia.org/wiki/Алгоритм\\_сортировки](https://ru.wikipedia.org/wiki/Алгоритм_сортировки) (04.03.2019)
7. Практика применения XOR в программировании – [Электронный ресурс] режим доступа:  
<https://habr.com/ru/post/183462/> (07.05.2019)
8. Сортировка вставками – [Электронный ресурс] режим доступа:  
[https://cpp.com.ru/shildt\\_spr\\_po\\_c/21/2106.html](https://cpp.com.ru/shildt_spr_po_c/21/2106.html) (24.05.2019)

**ПРИЛОЖЕНИЕ 1****ТЕХНИЧЕСКОЕ ЗАДАНИЕ  
на выполнение курсового проекта**

по дисциплине «Программирование и информатика»

Направление 090304 – Программная инженерия

Исполнитель: обучающийся гр. ДИПР611 Катунин С.С.

**Тема: Учебно-демонстрационная программа «Методы сортировок: сортировка Шелла»**

**1 Назначение, цели и задачи разработки**

**Цель разработки** – автоматизация обучения и контроля знаний по теме «сортировка Шелла».

**Назначение разработки:**

- повышение качества знаний пользователей;
- снижение нагрузки на преподавателя.

**Основные задачи**, решаемые разработчиком в процессе выполнения курсового проекта:

- анализ предметной области;
- разработка программного продукта в соответствии с требованиями;
- документирование проекта в соответствии с установленными требованиями.

**2 Характер разработки:** прикладная квалификационная работа.**3 Основания для разработки**

- Учебный план направления 09.03.04 «Программная инженерия» 2018 года набора.
- Рабочая программа дисциплины Программирование и информатика».
- Распоряжение по кафедре АСОИУ №\_\_\_\_ от «\_\_\_\_» 201\_г.

**4 Плановые сроки выполнения** – весенний семестр 2018/19 учебного года:

Начало «25» февраля 2019 г.

Окончание «15» июня 2019 г.

**5 Требования к проектируемой системе****5.1 Требования к функциональным характеристикам**

Проектируемая система представляет собой консольное приложение и должна обеспечивать выполнение следующих основных функций:

- Предоставление пользователю теоретического материала по теме «Методы сортировок: сортировка Шелла»
  - ✓ текст разбит на страницы не менее 10 и не более 20 строк по 60 символов в строке, перенос осуществляется по словам, выравнивание по ширине;

- ✓ программа должна обеспечивать пролистывание как в прямом, так и в обратном направлении;
- ✓ текст с теоретическим материалом разбит на абзацы и структурирован;
- ✓ текст хранится в файле, должен быть зашифрован (закодирован) таким образом, чтобы его нельзя было прочесть стандартными средствами операционной системы.
- Визуализация работы алгоритма сортировки Шелла:
  - ✓ пошаговая демонстрация работы алгоритма сортировки Шелла: на каждом шаге происходит вывод массива и значения  $d$  (расстояние между сортируемыми элементами) на экран; подсвечиваются сортируемые элементы, соответствующие данному шагу; после завершения сортировки будет выведено соответствующее сообщение;
  - ✓ должна быть предусмотрена возможность просмотра предыдущих шагов;
  - ✓ пользователю предоставлен выбор способа подготовки данных для демонстрации (генерация случайных чисел, ввод с клавиатуры, чтение из заранее заготовленного файла).
- Тестирование пользователя по теме «Методы сортировок: сортировка Шелла»:
  - ✓ вопросы закрытого типа с выбором одного правильного ответа, количество предлагаемых вариантов ответа – 4, при показе пользователю вопроса варианты ответа должны быть перемешаны;
  - ✓ при тестировании пользователю предлагается 5 вопросов, общее количество вопросов в базе не менее 10, база представляет собой файл, который должен быть зашифрован (закодирован) таким образом, чтобы его нельзя было прочесть стандартными средствами операционной системы;
  - ✓ формат файла и вопросы разрабатываются исполнителем самостоятельно;
  - ✓ перед тестированием пользователю должна быть предоставлена инструкция о проведении теста;
  - ✓ программа должна подсчитывать количество правильных и неправильных ответов; пользователь вводит номер выбранного им варианта ответа с клавиатуры; ответ считается верным, если выбран номер варианта, отмеченного как правильный;
  - ✓ если ответ правильный, то к количеству правильных ответов добавляется 1 балл, в противном случае 1 балл добавляется к количеству неправильных;
  - ✓ пользователь может пропустить вопрос с возможностью вернуться к нему;
  - ✓ после ввода ответа выводится сообщение о текущем результате, по завершении тестирования общий итог выводится как на экран, так и в зашифрованный файл (с

указанием данных о тестируемом, даты и времени прохождении теста, текста вопросов, в которых была допущена ошибка);

- ✓ тестирование считается успешным, если количество правильных ответов не менее 3.
- Предоставление пользователю режима администратора, который даёт возможность расшифровать и зашифровать файл с базой вопросов и файл с теорией.
- **Интерфейс программы:** текст русский, шрифт кириллический, заголовки, термины и другая важная информация выделены цветом.

Система имеет функциональные ограничения:

- элементы сортируемого элемента должны быть целыми числами из диапазона [-50; 50];
- количество элементов сортируемого массива должно быть не менее 5 и не более 15;
- программа не должна обеспечивать редактирование статистики результатов тестирования.

## 5.2 Требования к эксплуатационным характеристикам

Программа не должна аварийно завершаться при любых действиях пользователя

Время реакции программы на действия пользователя не должно превышать 10 секунд.

## 5.3 Требования к программному обеспечению:

Средства разработки: интегрированная среда Code::Blocks 17.12, язык C++ (стандарт C++ 11 и более поздние).

Операционная система: Windows XP (x86) с пакетом обновления 3 (SP3) или более поздние.

## 5.4 Требования к аппаратному обеспечению:

Рекомендуемая конфигурация:

- Intel-совместимый процессор с частотой не менее 1,6 ГГц;
- не менее 512 МБ ОЗУ;
- не менее 20 МБ свободного места на диске;
- Дисковод CD-ROM/DVD-ROM.

# 6 Стадии и этапы разработки

## 6.1 Эскизный проект (ЭП)

- Анализ предметной области.
- Подготовка проектной документации.

## 6.2 Технический проект (ТП)

- Разработка структур и форм представления данных.
- Разработка структуры программного комплекса.

- Подготовка пояснительной записки.

### **6.3 Рабочий проект (РП)**

- Программная реализация.
- Тестирование и отладка программы.
- Подготовка программной и эксплуатационной документации.

### **6.4 Эксплуатация (Э)**

Описание и анализ результатов проведенного исследования.

## **7 Требования к документированию проекта**

К защите курсового проекта должны быть представлены следующие документы:

- Пояснительная записка к курсовому проекту;
- Презентация доклада.
- Программа, презентация и пояснительная записка к курсовому проекту на оптическом носителе.

Требования к структуре документов определены соответствующими стандартами ЕСПД.

Требования к оформлению определены соответствующими методическими указаниями.

## **8 Порядок контроля и приемки**

Контроль выполнения курсового проекта проводится руководителем поэтапно в соответствии с утвержденным графиком выполнения проекта.

На завершающем этапе руководитель осуществляет нормоконтроль представленной исполнителем документации и принимает решение о допуске (недопуске) проекта к защите.

Защита курсового проекта проводится комиссией в составе не менее двух человек, включая руководителя проекта.

В процессе защиты проекта исполнитель представляет документацию, делает краткое сообщение по теме разработки и демонстрирует ее программную реализацию.

При выставлении оценки учитывается:

- степень соответствия представленной разработки требованиям технического задания;
- качество программной реализации, документации и доклада по теме проекта;
- соблюдение исполнителем графика выполнения курсового проекта.

## **9 Литература**

1. Златопольский Д.М. Программирование: типовые задачи, алгоритмы, методы 2-е изд. – М.: БИНОМ. Лаборатория знаний, 2012. – 223 с.: ил.
2. Лаптев В.В. С++. Экспресс-курс. – СПб.: БХВ-Петербург, 2004. – 512 с.: ил.

**ПРИЛОЖЕНИЕ 2****База текстовых вопросов**

**(содержание файла quest.dat в расшифрованном виде с выделенными жирным шрифтом вопросами и подчеркнутыми правильными ответами)**

**В чем заключается основное преимущество сортировки вставками?**

естественное поведение алгоритма

неестественное поведение алгоритма

экономное использование временной памяти

скорость

**В какой из сортировок требуется нахождение минимального элемента?**

сортировка выбором

пузырьковая сортировка

сортировка Шелла

сортировка вставками

**Сортировка Шелла является усовершенствованным вариантом...**

сортировки вставками

сортировки выбором

пузырьковой сортировки

является уникальной сортировкой

**Алгоритм сортировки - это...**

алгоритм для упорядочивания элементов в массиве

алгоритм для нахождения экстремума функции

алгоритм для разбиения множеств на подмножества

ничего из перечисленного

**Найдите несуществующую сортировку**

сортировка Белла

сортировка вставками

пузырьковая сортировка

сортировка Шелла

**Естественность поведения - это...**

эффективность алгоритма при обработке уже упорядоченных или частично упорядоченных данных

когда алгоритм не меняет взаимного расположения элементов с одинаковыми ключами

отсутствие потребности алгоритма в дополнительной памяти

когда алгоритм, использует для сортировки сравнение элементов между собой

**На каком из предложенных языков программирования невозможно реализовать сортировку Шелла?**

возможно на всех предложенных

C++

Python

Java

**Лучшее время (вычислительная сложность) для алгоритма вставками..**

O(n) сравнений, 0 обменов

O( $n^2$ ) сравнений, n обменов

O(n) сравнений,  $n^2$  обменов

O(n) сравнений, n обменов

**В сортировке вставками...**

элементы просматриваются по одному, и каждый новый элемент размещается в подходящее место среди ранее упорядоченных элементов

ищется номер минимального значения среди всех элементов

сравниваются элементы, стоящие не только рядом, но и на определённом расстоянии друг от друга

ничего из перечисленного

**Сортировка Шелла - это...**

ничего из перечисленного

поиск наименьшего или наибольшего элемента и помещение его в начало или конец упорядоченного массива

когда произвольно перемешивают массив, а потом проверяют порядок

рекурсивный алгоритм сортировки