|  |  |  |
| --- | --- | --- |
| лого для документов 2022 | ***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 09.03.01 Информатика и вычислительная техника  Профиль Автоматизированные системы обработки информации и управления  КафедраАвтоматизированные системы обработки информации и управления  от \_\_\_\_.\_\_\_\_\_\_\_\_\_\_\_\_2023 № \_\_\_\_\_\_ | | |
| **КУРСОВОЙ ПРОЕКТ**  **Учебно-демонстрационная программа**  **«односвязный циклический список»**  по дисциплине «Основы алгоритмизации и программирования» | | |
| Допущен к защите  «\_\_» \_\_\_\_\_\_\_\_2023г.  Руководитель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Оценка, полученная на защите  «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_» | | Проект выполнил:  обучающийся группы ДИНРБ-11  Самодуров Вячеслав Александрович  Руководитель  асс. Кравченкова Е.П. | |
| Члены комиссии:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |  | |
| Астрахань 2023 | | |

**АСТРАХАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Институт информационных технологий и коммуникаций**

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | Кафедра «Автоматизированные системы  обработки информации и управления» |
| Заведующий кафедрой  д.т.н., проф.  Т.В. Хоменко \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г. |

**ЗАДАНИЕ**

**на выполнение курсового проекта**

Обучающийся ***Самодуров Вячеслав Александрович***

Группа ***ДИНРб-11***

Дисциплина ***Основы алгоритмизации и программирования***

Тема ***Учебно-ознакомительная программа «Односвязный циклический список»***

Дата получения задания «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023г.  
Срок представления обучающимся КП на кафедру «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023г.

Руководитель ***ассистент***\_\_\_\_\_\_\_\_\_ ***Кравченкова Е.П.*** «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023г.

должность, степень, званиеподписьФИО

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ***Самодуров В.А.*** «\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023г.

подпись ФИО

**Задачи**

Разработка программного продукта, который

* предоставляет пользователю теоретический материал на тему односвязный циклический список;
* демонстрирует принцип работы односвязного циклического списка;
* тестирует пользователя по пройденному материалу.

**Список рекомендуемой литературы**

1. Кнут Д.Э. Искусство программирования, том 3. Сортировка и поиск, 2-е изд. : Пер. с англ. – М.: ООО «И.Д. Вильямс», 2018. – 832 с. : ил. – Парал. тит. англ.
2. Дж. Макконелл Анализ алгоритмов. Активный обучающий подход. — 3-е дополненное издание. М: Техносфера, 2009. -416с.

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | К заданию на курсовой проектпо дисциплине  «Основы алгоритмизации и программирования» |
| Заведующий кафедрой  д.т.н., профессор  Т.В. Хоменко\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023г. |

**КАЛЕНДАРНЫЙ ГРАФИК**

курсового проектирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Разделы, темы и их содержание, графический материал | Дата сдачи | Объем, % |
| 1 | Выбор темы | 23.01.2022 | 1 |
| 2 | Техническое задание | 08.02.2022 | 3 |
| 3 | Разработка модели, проектирование системы   * *введение,* * *технический проект,* * *программа и методика испытаний,* * *литература* | 15.02.2022 | 25 |
| 4 | Программная реализация системы   * *работающая программа,* * *рабочий проект* * *скорректированное техническое задание (при необходимости)* | 30.03.2022 | 40 |
| 5 | Тестирование и отладка системы, эксперименты   * *работающая программа с внесёнными изменениями,* * *окончательные тексты всех разделов* | 11.04.2022 | 50 |
| 6 | Компоновка текста  Подготовка презентации и доклада   * *пояснительная записка* * *презентация* * *электронный носитель с текстом пояснительной записки, исходным кодом проекта, презентацией и готовым программным продуктом* | 02.05.2022 | 59 |
| 7 | Защита курсового проекта | 10.05.2022-11.05.2022 | 60-100 |

С графиком ознакомлен «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023г.

Самодуров В.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, обучающийся группы ДИНРб-11

фамилия, инициалы, подпись

**График курсового проектирования выполнен**

**без отклонений / с незначительными отклонениями / со значительными отклонениями**

нужное подчеркнуть

Руководитель курсового проекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ассистент Кравченкова Е.П.

подпись, ученая степень, звание, фамилия, инициалы

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 6](#_Toc133939123)

[1 ТЕХНИЧЕСКИЙ ПРОЕКТ 7](#_Toc133939124)

[1.1 Анализ предметной области 7](#_Toc133939125)

[1.1.1 Односвязный циклический список. Понятие, назначение 7](#_Toc133939126)

[1.1.2 Операции 8](#_Toc133939127)

[1.1.3 Изучение односвязных циклических списков 11](#_Toc133939128)

[1.1.4 Проверка знаний пользователя 12](#_Toc133939129)

[1.2 Технология обработки данных 12](#_Toc133939130)

[1.2.1 Форматы данных 13](#_Toc133939131)

[1.2.2 Алгоритм демонстрации теоретического материала 13](#_Toc133939132)

[1.2.3 Алгоритм добавления элементов в конец списка 14](#_Toc133939133)

[1.2.4 Алгоритм добавления элементов между другими элементами списка 14](#_Toc133939134)

[1.2.5 Алгоритм вывода элементов списка 15](#_Toc133939135)

[1.2.6 Алгоритм поиска элемента по условию 16](#_Toc133939136)

[1.2.7 Алгоритм удаления элемента из списка 16](#_Toc133939137)

[1.2.8 Алгоритм удаления списка 17](#_Toc133939138)

[1.3 Входные и выходные данные 17](#_Toc133939139)

[1.4 Системные требования 18](#_Toc133939140)

[2 РАБОЧИЙ ПРОЕКТ 19](#_Toc133939141)

[2.1 Общие сведения о работе системы 19](#_Toc133939142)

[2.2 Функциональное назначение программного продукта 19](#_Toc133939143)

[2.3 Инсталляция и выполнение программного продукта 19](#_Toc133939144)

[2.4 Описание программы 20](#_Toc133939145)

[2.5 Разработанные меню и интерфейсы 22](#_Toc133939146)

[2.6 Сообщение системы 26](#_Toc133939147)

[3 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ 28](#_Toc133939148)

[3.1 Проверка работоспособности тем 28](#_Toc133939149)

[3.1 Проверка работоспособности теста 29](#_Toc133939150)

[ЗАКЛЮЧЕНИЕ 30](#_Toc133939151)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 31](#_Toc133939152)

[ПРИЛОЖЕНИЕ 1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ 32](#_Toc133939153)

[ПРИЛОЖЕНИЕ 2 БАЗА ВОПРОСОВ ДЛЯ ТЕСТИРОВАНИЯ 35](#_Toc133939154)

# ВВЕДЕНИЕ

*Динамической структурой* называется упорядоченное множество объектов, состав и взаимное расположение которых в процессе выполнения программы может динамически изменяться. Динамические структуры конструируются программистом с использованием связанной организации памяти и метода хранимого адреса.

Операции по модификации динамических структур:

* + создание/разрушение структуры;
  + включение объектов в структуру/исключение объектов из структуры;
  + выделение подмножества объектов структуры по определенным признакам;
  + объединение нескольких подмножеств объектов в определенном порядке в единую структуру.

В зависимости от отношения порядка, определенного на множестве объектов, различают линейные и нелинейные структуры данных.

*Линейной динамической структурой* *(списком)* называется множество объектов (элементов, узлов) ***S={si}, i=1, ... ,n****,* на котором определены отношения предшествования/следования, причем для любого объекта ***si, i=2, ... ,n-1*** существует единственный “предшественник” ***si-1*** и единственный “последователь” ***si+1***. Объект ***s1*** не имеет предшественника и является первым элементом списка, объект ***sn*** не имеет последователя и является “хвостом” списка. [[1]](#Источник1)

Целью создания учебно-ознакомительной программы является понятное и малозатратное обучение односвязным циклическим спискам.

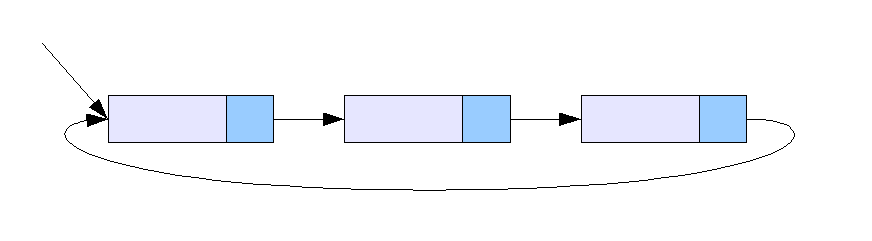
Назначение программы – повышение качества знаний студентов и их снижение затрат сил и времени.

# 1 ТЕХНИЧЕСКИЙ ПРОЕКТ

## Анализ предметной области

### 1.1.1 Односвязный циклический список. Понятие, назначение

*Односвязный циклический список* (сокращенно – *циклический список*) обладает той особенностью, что поле связи его последнего элемента указывает на первый элемент списка. В целях удобства обработки в структуру циклического списка включают специальный элемент с особым содержанием информационного поля (на рисунке 1.1 это первый элемент), называемый головой списка или “сторожем”. Голова списка является дополнительным элементом. Назначение этого элемента состоит в том, чтобы отметить точку входа в циклический список, а также упростить включение элементов в начало списка и исключение элементов из начала списка. На рисунке 1.1 показана структура односвязного циклического списка с головным элементом.



**Голова**

**Хвост**

Рисунок 1.1 - Структура односвязного циклического списка с головным элементом

Односвязные циклические списки можно использовать для реализации линейных структур, таких как очереди, стеки, списки произвольного вида. При создании *очереди* новый элемент включается в “хвост” списка, то есть “перед” головным элементом (рисунок 1.2). При создании *стека* новый элемент включается в начало списка, то есть непосредственно “за” головным элементом (рисунок 1.3). [1]

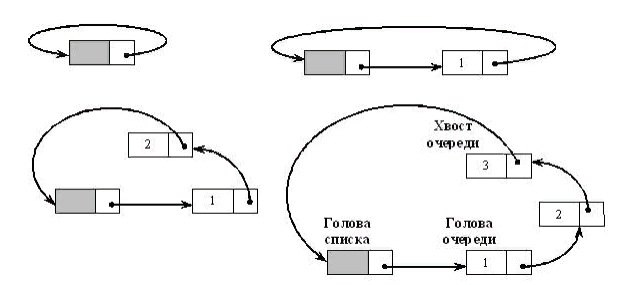


Рисунок 1.2 - Создание очереди на структуре односвязного циклического списка

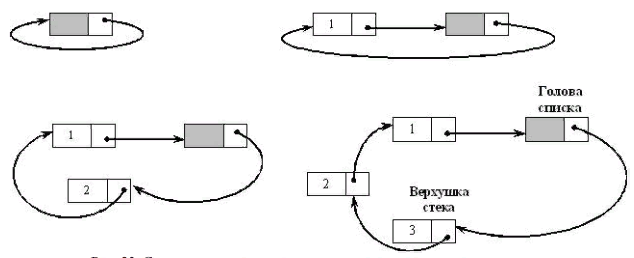


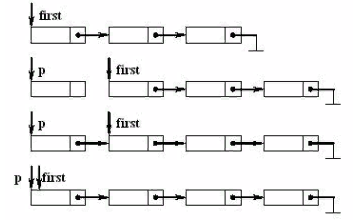
Рисунок 1.3 - Создание стека на структуре односвязного циклического списка

### 1.1.2 Операции

#### 1.1.2.1 Включение элемента в начало списка

Одна из самых простых операций по модификации спиччска - включение нового элемента в его начало (рисунок 1.4): элемент хранения типа ***List*** размещается в памяти и указатель на него присваивается некоторой вспомогательной переменной ***p***, затем устанавливается связь между вставленным элементом и списком, после чего указатель на первый элемент списка получает новое значение и указателю последнего элемента присваивается этот элемент.





**Голова**

**Голова**

**Голова**

**Голова**

**Хвост**

**Хвост**

**Хвост**

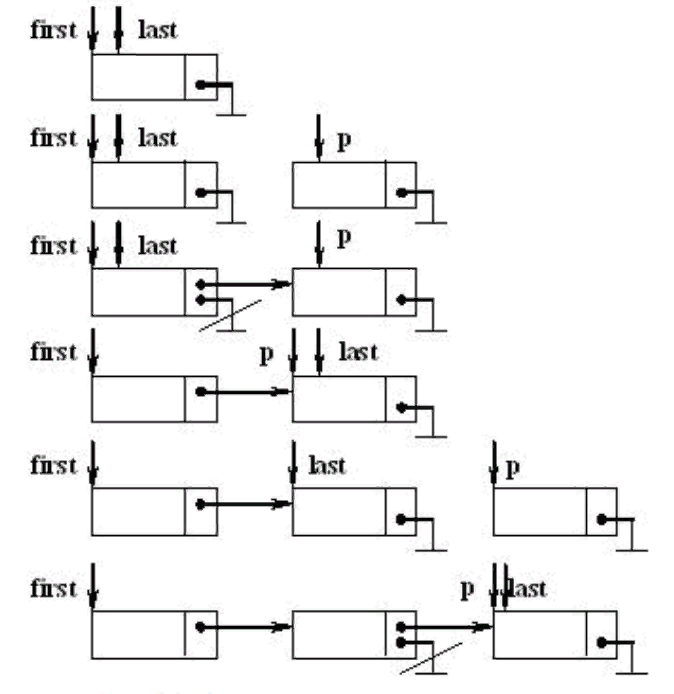
**Хвост**



Рисунок 1.4 - Включение элемента в начало списка

#### 1.1.2.2 Добавление элемента в конец списка

Первый элемент создается отдельно (так как включить элемент «за» несуществующим элементом невозможно), а остальные ***(n-1)*** элементов создаются и включаются в хвост списка одинаковым образом. При этом удобно использовать вспомогательный указатель на последний добавленный элемент. Значение этого указателя изменяется в процессе создания списка, значение указателя на первый элемент списка не изменяется после создания первого элемент. Порядок следования элементов в списке получается прямым, так как в начале списка находится тот элемент, который был включен в список первым (рисунок 1.5).



**Хвост**

**Хвост**

**Хвост**

**Хвост**

**Хвост**

**Хвост**

**Голова**

**Голова**

**Голова**

**Голова**

**Голова**

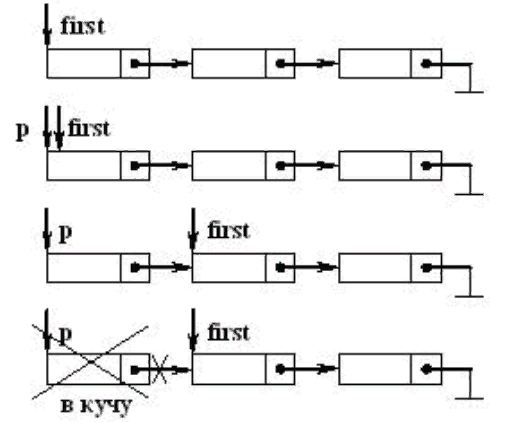
**Голова**



Рисунок 1.5 - Включение элемента в конец списка

#### 1.1.2.3 Исключение элемента из начала списка

Для того чтобы исключить из списка первый элемент, необходимо установить на него вспомогательный указатель, присвоить указателю на начало списка и указателю хвоста адрес следующего элемента списка, после чего область памяти, занятую первым элементом списка, вернуть в кучу (рисунок 1.6).



**Голова**

**Голова**

**Голова**

**Голова**

**Хвост**

**Хвост**

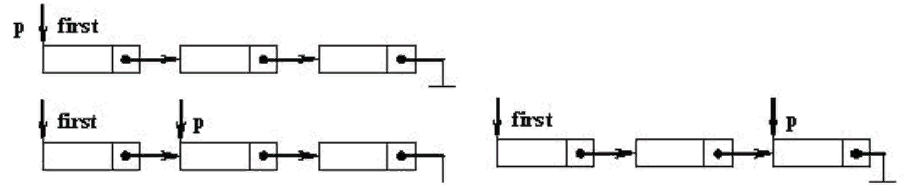
**Хвост**

**Хвост**

Рисунок 1.6 - Исключение элемента из начала списка

#### 1.1.2.4 Переустановка указателя (вывод списка)

Доступ к объектам динамической структуры может быть получен с помощью единственного вспомогательного указателя, который будет последовательно изменяться, всякий раз принимая значение адреса соседнего объекта, в направлении стрелки, изображающей связь. Адрес соседнего объекта извлекается из поля связи того элемента списка, на который в текущий момент ссылается указатель, затем полученный адрес присваивается этому указателю, который теперь открывает доступ к соседнему элементу списка. Такая операция называется перестановкой указателя (рисунок 1.7). Операция переустановки указателя используется, если необходимо единообразно обработать все или несколько следующих подряд элементов списка (для этого следует организовать цикл, включающий операции обработки элемента и перестановки указателя). В этом случае последовательность операций переустановки указателя обеспечивает проход по списку.



**Хвост**

**Хвост**

**Хвост**

**Голова**

**Голова**

**Голова**

Рисунок 1.7 - Переустановка указателя

#### 1.1.2.5 Поиск элемента в списке по заданному условию

Условием поиска элемента в списке может быть:

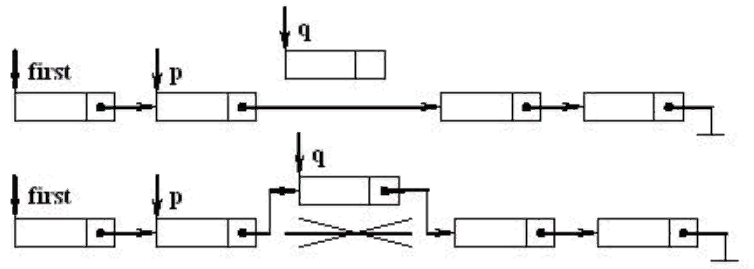
* + значение информационного поля элемента;
  + порядковый номер элемента в списке, начиная от первого узла;
  + адрес элемента списка, который хранится в некотором указателе.

Поиск элемента в списке по заданному условию обычно организуется в цикле, включающем операции проверки выполнения условия для текущего элемента, на который ссылается указатель, и переустановки указателя на соседний элемент списка (то есть поиск осуществляется в процессе прохода по списку). Проверка условия связана с вычислением булевских выражений в условных операторах или операторах цикла, использующих доступ к атрибутам объектов через указатели.

Поиск заканчивается либо при обнаружении элемента списка, соответствующего заданному условию (результатом поиска в этом случае является значение указателя, установленного на искомый элемент), либо при достижении конца списка, если элемент, соответствующий условию поиска, не обнаружен (результатом поиска в этом случае является ***NULL***).

#### 1.1.2.6 Включение нового элемента в список за элементом

Для того чтобы включить в список новый элемент за элементом необходимо установить на него указатель и разместить элемент хранения в области динамической памяти и выполнить последовательность операций, которая иллюстрируется рисунком 1.8. После выполнения операции вставки значение указателя на первый элемент списка не изменяется. Значение указателя на тот элемент, за которым выполнена вставка, также не изменяется.



**Голова**

**Голова**

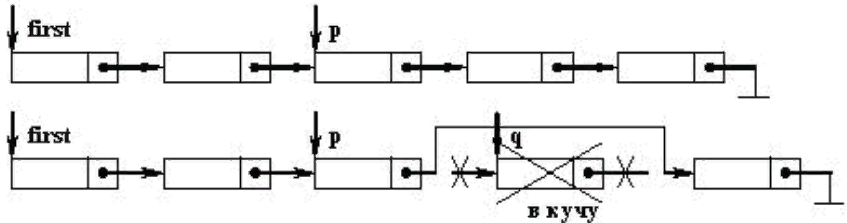
**Хвост**

**Хвост**

Рисунок 1.8 - Включение узла в список за тем узлом, на который установлен указатель

#### 1.1.2.7 Исключение из списка элемента за элементом

Исключение из списка элемента за элементом выполняется с использованием вспомогательного указателя на исключаемый элемент. Последовательность операций исключения иллюстрируется рисунком 1.9. После выполнения операции исключения значение указателя на первый элемент списка не изменяется. Значение указателя на тот элемент, за которым выполнено исключение, также не изменяется.



**Голова**

**Голова**

**Хвост**

**Хвост**

Рисунок 1.9 - Исключение элемента за тем элементом, на который установлен указатель

#### 1.1.2.8 Разрушение списка

Операция разрушения списка реализуется в процессе прохода по списку так: содержимое всех элементов списка, начиная с первого, последовательно исключаются из списка и возвращаются в кучу. В результате выполнения операции разрушения список становится пустым, то есть значение указателя на первый элемент равно NULL. Для разрушения списка недостаточно просто присвоить NULL указателю на его первый элемент, так как в этом случае память, занятая содержимым элементов списка, в действительности не освобождается и в кучу не возвращается.

### 1.1.3 Изучение односвязных циклических списков

Тема односторонних циклических списков в высшем учебном заведении подразумевает самостоятельное её изучение. Но для программистов необходимо изучать все методы и структуры языков и в совершенстве владеть ими, чтобы быть и оставаться востребованным в своей сфере. Поэтому данная программа поможет студентам узнать, как работают, создаются, редактируются и выводятся односторонние циклические списки.

Программа предоставит возможность изучить темы в любом порядке неограниченное количество раз, пока пользователь не пожелает закончить изучение. [1]

### 1.1.4 Проверка знаний пользователя

Для проверки знаний создан тест из 9 вопросов (1 вопрос на одну тему), в каждом из которых 3 варианта ответа, только один из них верный. Тест доступен пользователю только после прохождения всех тем, после прохождения также даётся чит-код (1489), который даёт возможность пользователю после повторного открытия программы активировать его в меню и получить доступ к тесту сразу. Вопросы и варианты ответов к ним находятся внутри кода.

Пользователь может:

* + после просмотра всех тем выйти из программы или начать тестирование;
  + при решении теста ввести вариант ответа, который является числом в диапазоне [1, 3];
  + увидеть результаты тестирования:

1. количество правильных ответов;
2. количество неправильных ответов;
3. оценку теста и рекомендации, основанные на количестве верных ответов.
   * после завершения теста вернуться в меню и повторить темы для повторного тестирования.

Все вопросы созданы на основе теоретического материала, предоставленного пользователю до этого в виде тем. По окончании тестирования пользователь возвращается в меню и может продолжить изучение тем, завершить работу программы либо пройти тестирование еще раз.

## 1.2 Технология обработки данных

Вариант использования «Меню» включает одну из 4 обязательных функций: «О программе», «Выбор темы», «Тест» и «Выход из программы».

Вариант использования «О программе» включает одну обязательную функцию «Обратно в меню».

Вариант использования «Выбор темы» включает одну из обязательных 11 функций, где 9 из них это темы обучения: «Об односвязном циклическом списке», «Создание структуры списка», «Создание класса списка», «Добавление элементов в конец списка», «Добавление элементов между другими элементами списка», «Вывод элементов списка», «Поиск элемента по условию», «Удаление элемента из списка», «Удаление списка»; и остальные это «Обратно в меню» и «Выход из программы».

Вариант использования 9 вышенаписанных тем включает в себя одну из 6 обязательных функций: «Повтор темы», «Следующая тема», «Предыдущая тема», «Обратно к темам», «Обратно в меню», «Показать список» и «Выход из программы».

Вариант использования «Тест» включает одну обязательную функцию «Ответить на вопрос».

### 1.2.1 Форматы данных

Изучение тем представляет собой вывод информации и нужного сегмента кода текстом и результат, в зависимости от темы. На рисунке 1.11 приведён пример, как будет выглядеть изучение.

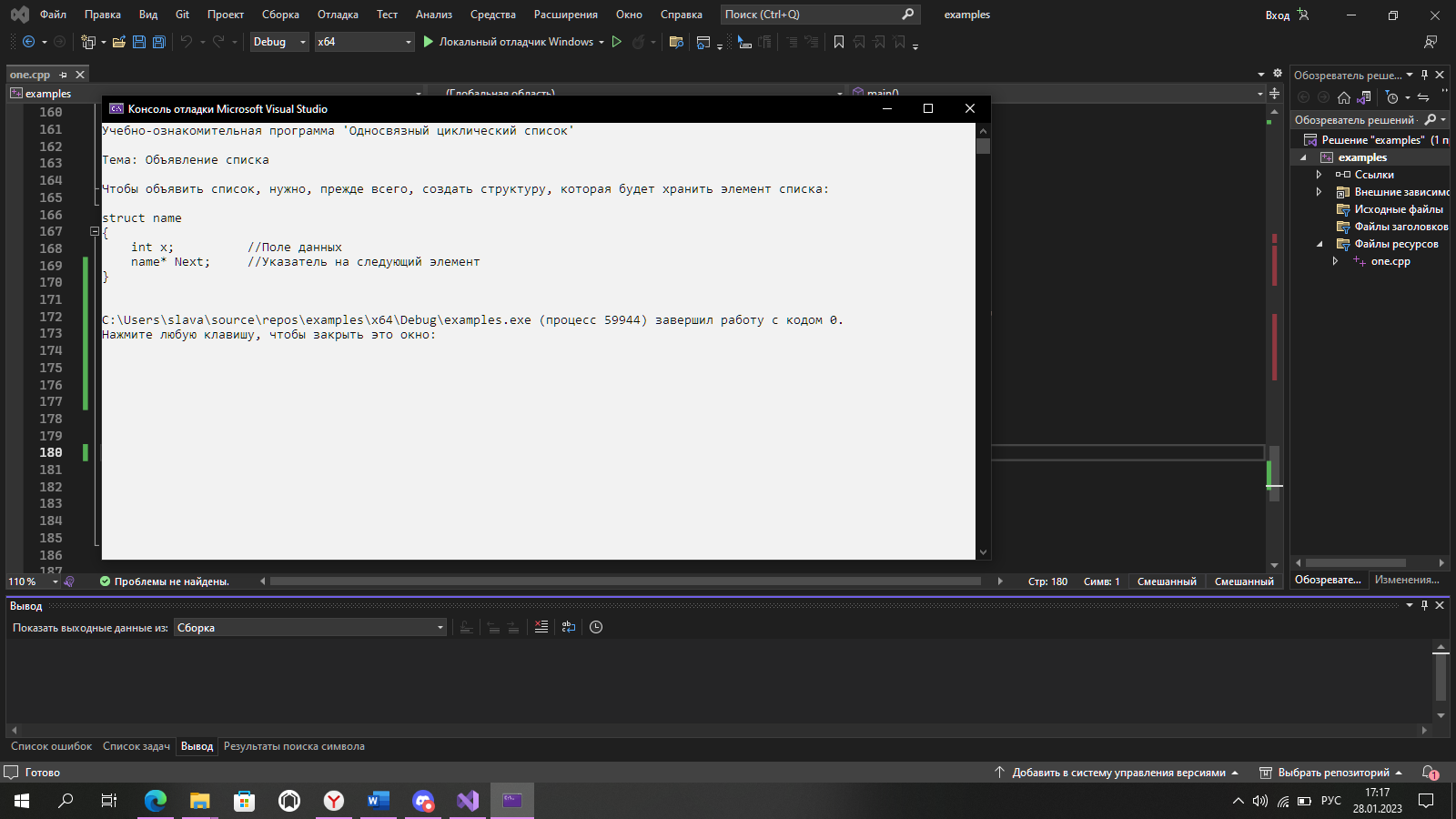


Рисунок 1.11 – Пример визуализации тем на консоли

Данные фрагменты кодов будут доступны для копирования, чтобы пользователь мог составить свой код и проверить его работу.

### 1.2.2 Алгоритм демонстрации теоретического материала

**Дано:** меню.

**Получить:** свободное изучение теоретического материала.

1. Вывод информации о программе и меню действий: «О программе», «Выбор темы», «Тест» и «Выход из программы»;
2. если пользователь выбрал «О программе», вывести информацию о программе и дать выбор из одного действия - «Обратно в меню», после которого перейти к пункту 1;
3. если пользователь выбрал «Выбор темы», вывести список из 9 тем: «Об односвязном циклическом списке», «Создание структуры списка», «Создание класса списка», «Добавление элементов в конец списка», «Добавление элементов между другими элементами списка», «Вывод элементов списка», «Поиск элемента по условию», «Удаление элемента из списка», «Удаление списка»; и 2 функции: «Обратно в меню» и «Выход из программы»;
4. если пользователь выбрал темы «Об односвязном циклическом списке», «Создание структуры списка» и «Создание класса списка», то перейти к пункту 5, после чего к пункту 6;
5. вывести теоретический материал;
6. запросить выбор из следующих действий:
   1. повтор темы;
   2. следующая тема;
   3. предыдущая тема;
   4. обратно к темам;
   5. обратно в меню;
   6. показать список;
   7. выход из программы.
7. если выбрано действие «a» то повторить тему;
8. если выбрано действие «b», то вывести теоретический материал следующей темы;
9. если выбрано действие «c», то вывести теоретический материал прошлой темы;
10. если выбрано действие «d», то вернуться к пункту 3;
11. если выбрано действие «e», то вернуться к пункту 1;
12. если выбрано действие «c», то вывести содержание списка;
13. если пользователь выбрал «Выход из программы», то завершить программу и выйти;
14. если пользователь выбрал темы «Добавление элементов в конец списка», «Добавление элементов между другими элементами списка», «Вывод элементов списка», «Поиск элемента по условию», «Удаление элемента из списка», «Удаление списка», то перейти к пункту 5, после перейти к пункту 15, а затем к пункту 6;
15. показать теорию на практике, запрашивая данные для списка (алгоритмы изложены ниже);
16. если пользователь изучил все темы, дать ему доступ к тесту и чит-код к нему;
17. если пользователь в пункте 1 выбрал «Тест» до изучения тем, то вывести ему сообщение о прохождении всех тем;
18. если пользователь в пункте 1 выбрал «Тест» после изучения всех тем, то предоставить ему 9 вопросов по 3 варианта ответа;

### 1.2.3 Алгоритм добавления элементов в конец списка

**Дано**: пустой односвязный циклический список.

**Получить**: односвязный циклический список с элементами.

1. Увеличить размер на 1 ;
2. создать новый элемент;
3. запросить ввод данных для элемента;
4. присвоить данные элементу;
5. присвоить указателю элемента адрес головы;
6. если головы нет, то элемент сделать головой и хвостом;
7. если голова есть, то указателю хвоста присвоить адрес элемента, после чего элемент сделать хвостом.

### 1.2.4 Алгоритм добавления элементов между другими элементами списка

**Дано**: заполненный односвязный циклический список с размером 2 и больше.

**Получить**: заполненный односвязный циклический список с размером на 1 больше.

1. Запросить номер позиции больше 1, но не больше размера списка;
2. запросить ввод данных для элемента;
3. создать 2 новых элемента: добавляемый и копию головы;
4. присвоить данные добавляемому элементу;
5. если номер позиции равен размеру списка:
   1. присвоить указателю добавляемого элемента адрес хвоста;
   2. создать переменную со значением 1;
   3. присвоить копии головы следующий за ним элемент;
   4. прибавить 1 к переменой;
   5. пока переменная не меньше позиции на 1, вернуться к пункту «c»;
   6. присвоить указателю этого элемента адрес добавляемого элемента;
6. если номер позиции равен любому другому:
   1. создать первую переменную со значением 1;
   2. присвоить копии головы следующий за ним элемент;
   3. если первая переменная меньше позиции на 1:
      1. присвоить указателю добавляемого элемента адрес копии головы;
      2. вернуть копии головы значение головы;
      3. создать вторую переменную со значением 1;
      4. присвоить копии головы следующий за ним элемент;
      5. прибавить 1 ко второй переменой;
      6. пока вторая переменная не меньше позиции на 1, вернуться к пункту «iii»;
      7. присвоить указателю копии головы адрес добавляемого элемента.
   4. прибавить 1 к первой переменой;
   5. пока первая переменная не равна позиции, вернуться к пункту «b».
7. увеличить размер на 1.

### 1.2.5 Алгоритм вывода элементов списка

**Дано**: заполненный элементами односвязный циклический список.

**Получить**: вывод на экран заполненного элементами односвязного циклического списка.

1. Создать копию головы;
2. создать переменную, которой присвоить размер списка;
3. вывести данные элемента;
4. присвоить копии головы следующий элемент списка;
5. уменьшить переменную размера на 1;
6. пока размер не равен нулю вернуться к пункту 3.

### 1.2.6 Алгоритм поиска элемента по условию

**Дано**: заполненный элементами односвязный циклический список.

**Получить**: элемент, удовлетворяющий условию.

1. Создать копию головы;
2. запросить ввод способа поиска: по расположению или по наличию данных;
3. если выбран поиск по расположению:
   1. запросить ввод номера позиции больше нуля и не больше размера списка;
   2. создать переменную со значением 1;
   3. присвоить копии головы следующий за ним элемент;
   4. прибавить 1 к переменой;
   5. пока переменная не равна позиции, вернуться к пункту «c»;
   6. вывести элемент.
4. если выбран поиск по наличию данных:
   1. запросить ввод данных;
   2. создать переменную со значением 1;
   3. если данные копии головы совпадают с введёнными данными, то вывести копию головы;
   4. присвоить копии головы следующий за ним элемент;
   5. прибавить 1 к переменой;
   6. пока переменная не равна размеру списка, вернуться к пункту «c»;
   7. если ни одного элемента с введёнными данными нет, то вывести сообщение о том, что элемент отсутствует.

### 1.2.7 Алгоритм удаления элемента из списка

**Дано**: заполненный элементами односвязный циклический список.

**Получить**: односвязный циклический список без удалённого элемента.

1. Создать копии головы и хвоста;
2. запросить ввод способа удаления: по расположению или по наличию данных;
3. если выбран поиск по расположению:
   1. запросить ввод номера позиции больше нуля и не больше размера списка:
   2. создать переменную со значением 1;
   3. присвоить копии хвоста копию головы;
   4. присвоить копии головы следующий за ним элемент;
   5. прибавить 1 к переменой;
   6. пока переменная не будет равная позиции, вернуться к пункту «b»;
   7. присвоить указателю копии хвоста адрес, следующий за копией головы;
   8. удалить копию головы;
   9. уменьшить размер списка на 1.
4. если выбран поиск по наличию данных:
   1. запросить ввод данных;
   2. создать переменную со значением 1;
   3. если данные копии головы равны введённым данным:
      1. создать новый элемент;
      2. присвоить указателю копии хвоста адрес, следующий за копией головы;
      3. новый элемент сделать копией головы;
      4. если элемент в списке один, то удалить новый элемент и закончить цикл;
      5. если элемент в списке не один и он является головой, то головой сделать следующий элемент;
      6. если элемент в списке не один и он не является головой:
         1. присвоить копии хвоста копию головы;
         2. присвоить копии головы следующий за ним элемент.
      7. удалить новый элемент.
   4. если данные копии головы равны введённым данным:
      1. присвоить копии хвоста копию головы;
      2. присвоить копии головы следующий за ним элемент.
   5. прибавить 1 к переменой;
   6. пока переменная не будет равная размеру списка, вернуться к пункту «c»
   7. уменьшить размер списка на количество удалённых.

### 1.2.8 Алгоритм удаления списка

**Дано**: заполненный элементами односвязный циклический список.

**Получить**: удалённый список.

1. Создать новый элемент;
2. новый элемент сделать элементом, следующим за головой;
3. удалить голову;
4. голову сделать новым элементом;
5. уменьшить размер списка;
6. пока размер списка не равен нулю, вернуться к пункту 1.

## Входные и выходные данные

Входные данные:

* + выбор действий;
  + выбор темы;
  + данные для элемента списка;
  + позиция элемента в списке;
  + ответ пользователя на тестовый вопрос.

Выходные данные:

* + меню действий;
  + теоретический материал темы и сегмент кода;
  + элементы списка;
  + операции списков;
  + вопросы теста и его итоги.

## Системные требования

Рекомендуемая конфигурация:

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

Операционная система: Windows 7. Среда разработки – Microsoft Visual Studio 2022.

# 2 РАБОЧИЙ ПРОЕКТ

## 2.1 Общие сведения о работе системы

Программный продукт разработан в интегрированной среде Microsoft Visual Studio 2022 на языке C++. Программа работает под управлением операционной системы Windows 7 (x64) и более поздними. Стартовый модуль для Microsoft Visual Studio 2022 – Learning.cpp.

## 2.2 Функциональное назначение программного продукта

Разработанный программный продукт предназначен для изучения темы «односвязный циклический список» и проверки степени усвоения материала пользователем. Программа имеет следующие функциональные возможности:

* + предоставление пользователю свободного перемещения по программе;
  + предоставление пользователю теоретического материала и кода;
  + наглядная демонстрация кода по теории;
  + прекращение демонстрации кода по желанию пользователя;
  + вывод содержимого списка после каждой темы;
  + ввод чит-кода для моментального доступа к тесту;
  + предоставление пользователю пройти тест из 9 вопросов;
  + проверка правильности решения тестовых вопросов;
  + демонстрация результатов теста;
  + рекомендация на основе результатов теста;
  + прекращение работы программы по желанию пользователя.

Программа имеет следующие функциональное ограничение: вводить необходимо только целые числа.

## 2.3 Инсталляция и выполнение программного продукта

Для выполнения программы необходимо:

1. Скопировать на жесткий диск компьютера папку Курсовой Проект, содержащую проект Learning List.sln и модуль Learning.cpp.
2. Запустить интегрированную среду Microsoft Visual Studio.
3. Открыть из среды файл examples.sln.
4. Убедиться, что модуль Learning.cpp назначен стартовым.
5. Запустить на выполнение модуль.

## 2.4 Описание программы

В таблице 2.1 приведено описание структуры Node, используемой в программе.

**Таблица 2.1 – Описание структуры Node**

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип** | **Назначение** |
| x | int | Поле данных |
| Next | Node\* | Указатель на следующий элемент |

В таблице 2.2 приведено описание класса List, используемого в программе.

**Таблица 2.1 – Описание класс List**

|  |  |  |  |
| --- | --- | --- | --- |
| **Поле** | **Тип** | **Назначение** | |
| Head | Node\* | Объявление первого элемента списка | |
| Tail | Node\* | Объявление последнего элемента списка | |
| size | int | Количество элементов в списке | |
| **Метод** | | | **Назначение** |
| void Add(int n, int count, int BeforeChoice, int Insta); | | | Добавление элементов в список |
| void Show(int size, List& lst, int Insta); | | | Отображение элементов списка |
| int Count(); | | | Возвращает количество элементов |
| void Find(); | | | Поиск элемента в списке |
| void DelEl(); | | | Удаление элемента в списке |
| ~List(); | | | Освобождение памяти |

В таблице 2.3 приведены функции, используемые в программе.

**Таблица 2.3 – Функции программы**

|  |  |
| --- | --- |
| **Прототип** | **Назначение** |
| void Color(int mode) | Устанавливает цвет mode |
| void Paint(wstring text, int mode) | Изменяет цвет параметра text в соответствии с mode |
| void CheckEnter(int& choice, int NumberOfChoice) | Проверяет ввод пользователя с ограничением от 0 до количества действий |
| void CheckEnter(int& choice, int NumberOfChoice, int& check) | Проверяет ввод пользователя с ограничением от 0 до количества действий, активирует чит-код |
| void CheckEnter(int& num) | Проверяет ввод пользователя |
| void Introduction() | Выводит информацию о программе |

**Продолжение таблицы 2.3**

|  |  |
| --- | --- |
| void AboutList() | Выводит теоретический материал о списке |
| void StructureOfList() | Выводит теоретический материал о структуре списка |
| void ClassOfList() | Выводит теоретический материал о классе списка |
| void AddElementToEnd(List& lst) | Выводит теоретический материал о добавлении элементов в конец списка |
| void AddElementBetweenNum(List& lst) | Выводит теоретический материал о добавлении элементов в список между другими элементами |
| void OutputList(List& lst) | Выводит теоретический материал о выводе списка |
| void LookingForElement(List& lst) | Выводит теоретический материал о поиске элемента в списке |
| void DeleteElement(List& lst) | Выводит теоретический материал об удалении элемента из списка |
| void DeleteList(List& lst) | Выводит теоретический материал об удалении списка |
| void DeleteListBeforeTheme(List& lst) | Выводит сообщение о непрохождении темы удаления списка и удаляет список |
| void ChooseThemes (int ch, int rep, int& ReturnMenu, int& ReturnThemes, List& lst, int& check) | Отвечает за активацию действий из списка тем |
| void Themes(int& ReturnMenu, int rep, List& lst, int& check) | Выводит список тем и 2 действия |
| void Test(int check, int& num, int& score, int wrong[9], int& have) | Тестирование |
| void ChooseMenu(int choice, int rep, int& ReturnMenu, List& lst, int& check) | Отвечает за активацию действий из меню |
| void Menu(int err, List& lst, int& check) | Выводит стартовое меню |

В программе имеется множество переменных. В таблице 2.4 приведены важнейшие переменные, используемые в программе.

**Таблица 2.4 – Важнейшие переменные программы**

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| check | int | При значении 9 и выше даёт доступ к тесту |
| lst | List | Показ работы со списком |
| score | int | Количество баллов теста |
| choice | int | Выбор действия |

## 2.5 Разработанные меню и интерфейсы

После запуска программы появится её титульный лист и меню. На титульном листе представлена тема учебно-демонстрационной программы, а также её автор (рис. 2.1). Среди пунктов меню представлен выбор из четырёх различных вариантов: информация о программе, выбор темы, тестирование и возможность завершения работы программы.

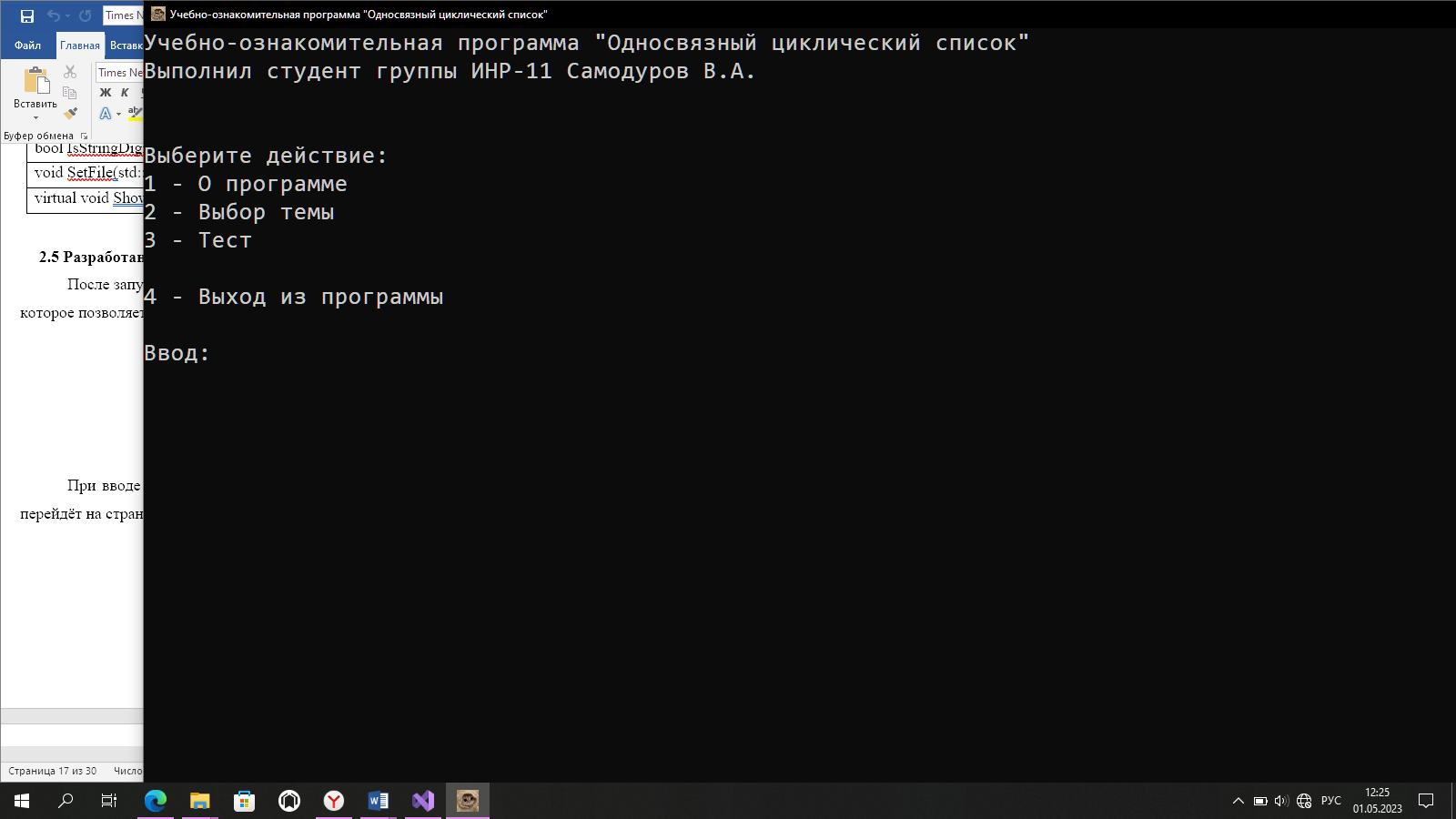


Рисунок 2.1 – Меню и титульный лист

При вводе числа 4 программа прекратит выполнение. При вводе числа 1 пользователь перейдёт на страницу информации о программе. (рисунок 2.2), при вводе числа 2 – на страницу выбора темы (рисунок 2.3), при вводе числа 3 – на страницу прохождения теста (рисунок 2.4).

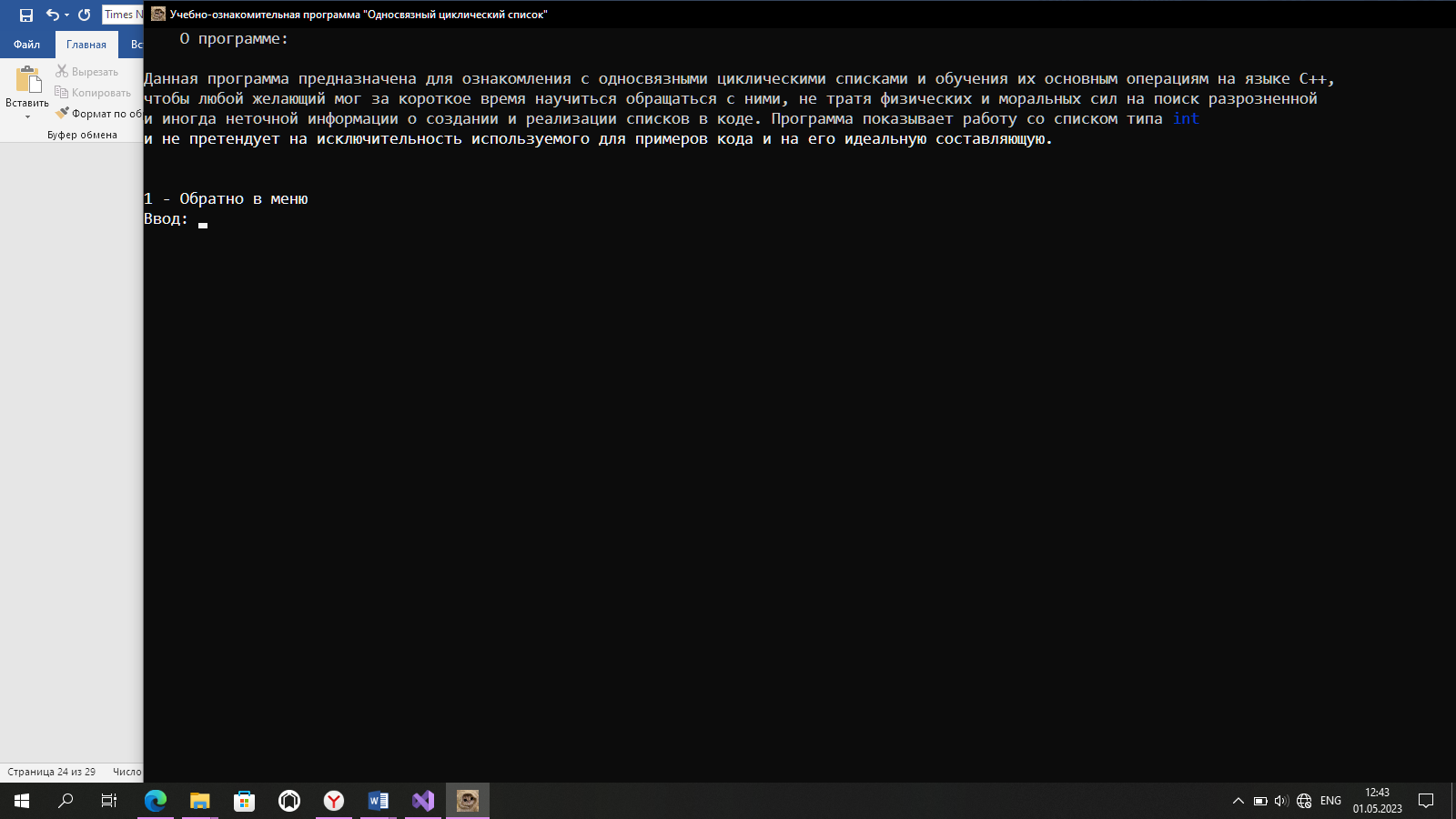


Рисунок 2.2 - Информация о программе

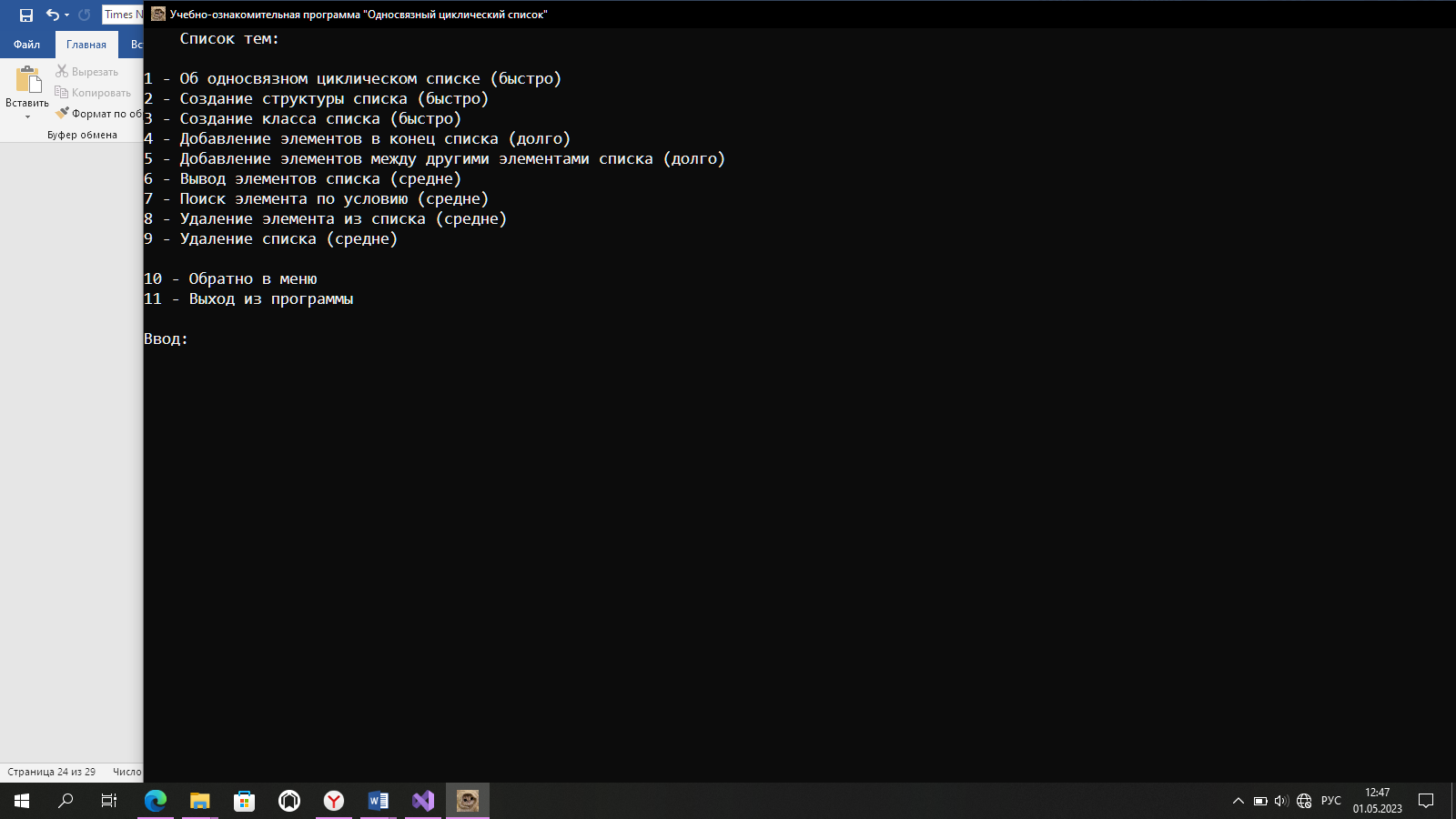


Рисунок 2.3 - Выбор тем

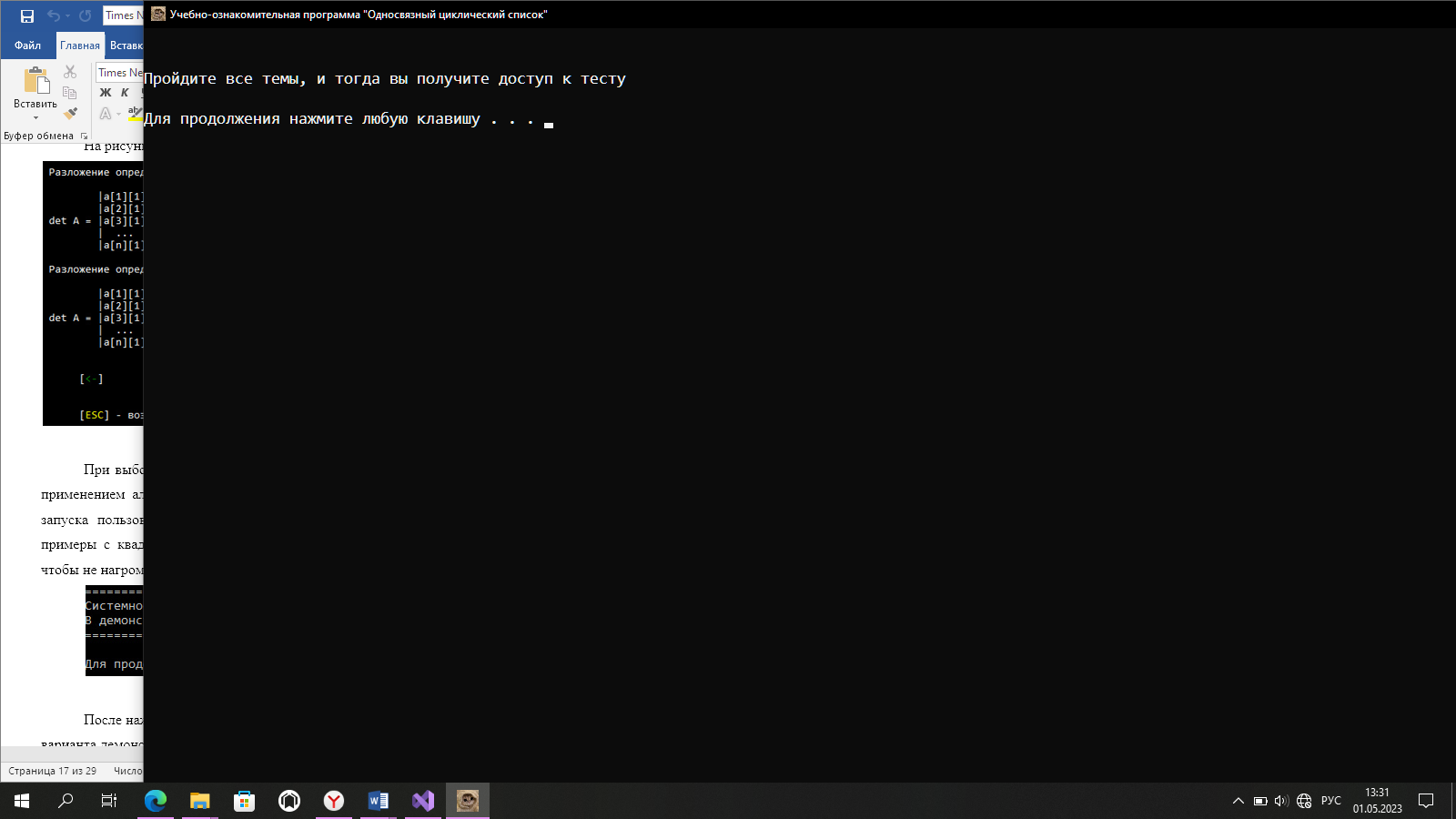


Рисунок 2.4 - Тестирование до изучения тем

На странице выбора тем (рисунок 2.3) пользователь не ограничен в выборе последовательности изучения тем, так как программа нацелена не на ускоренное обучение, а на понятное усвоение материала в рамках языка программирования «С++».

На рисунке 2.5 представлен фрагмент темы о добавлении элемента в конец списка. На рисунке 2.6 представлен фрагмент практической части этой темы.

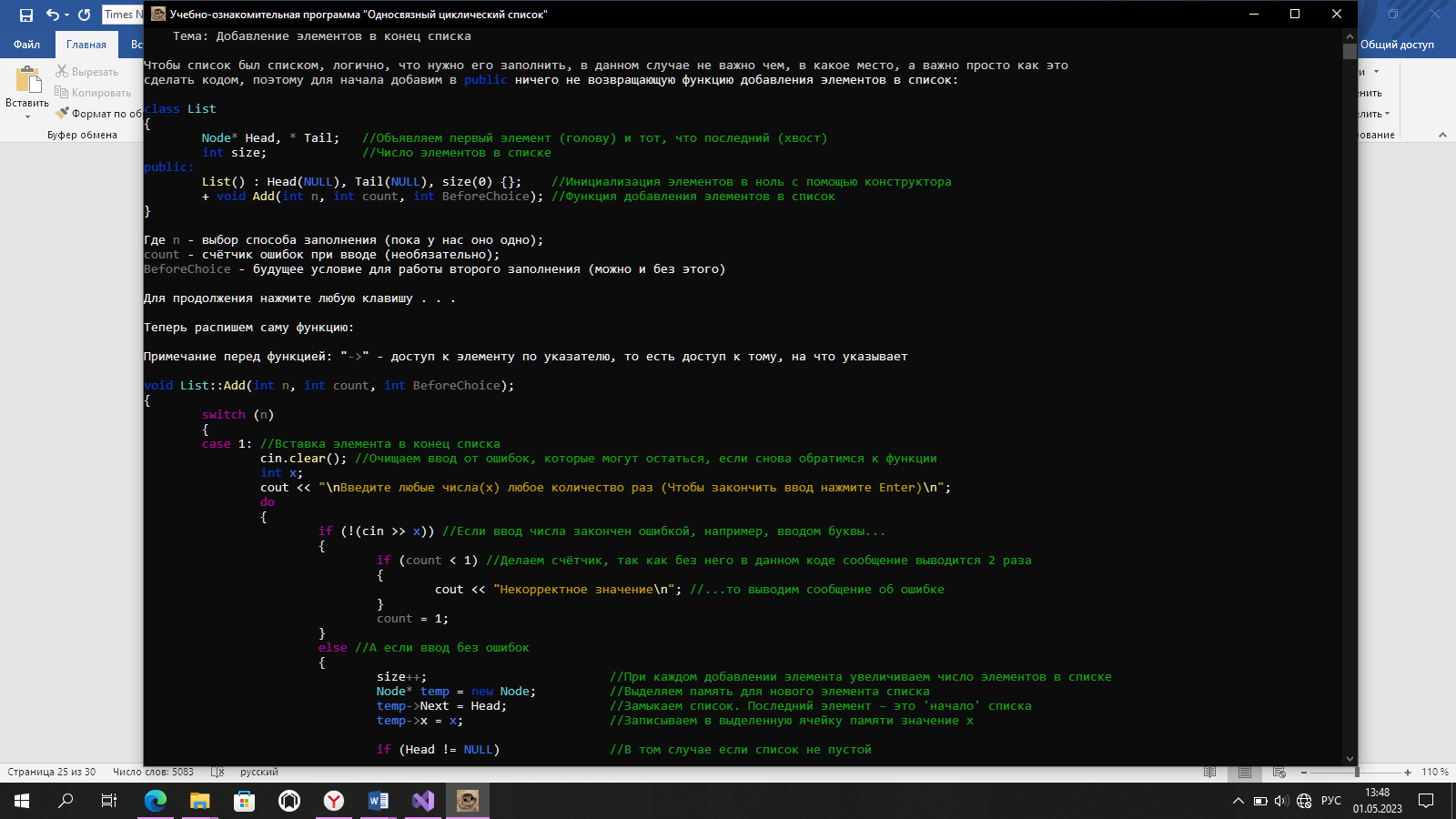


Рисунок 2.5 – Фрагмент темы о добавлении элемента в конец списка

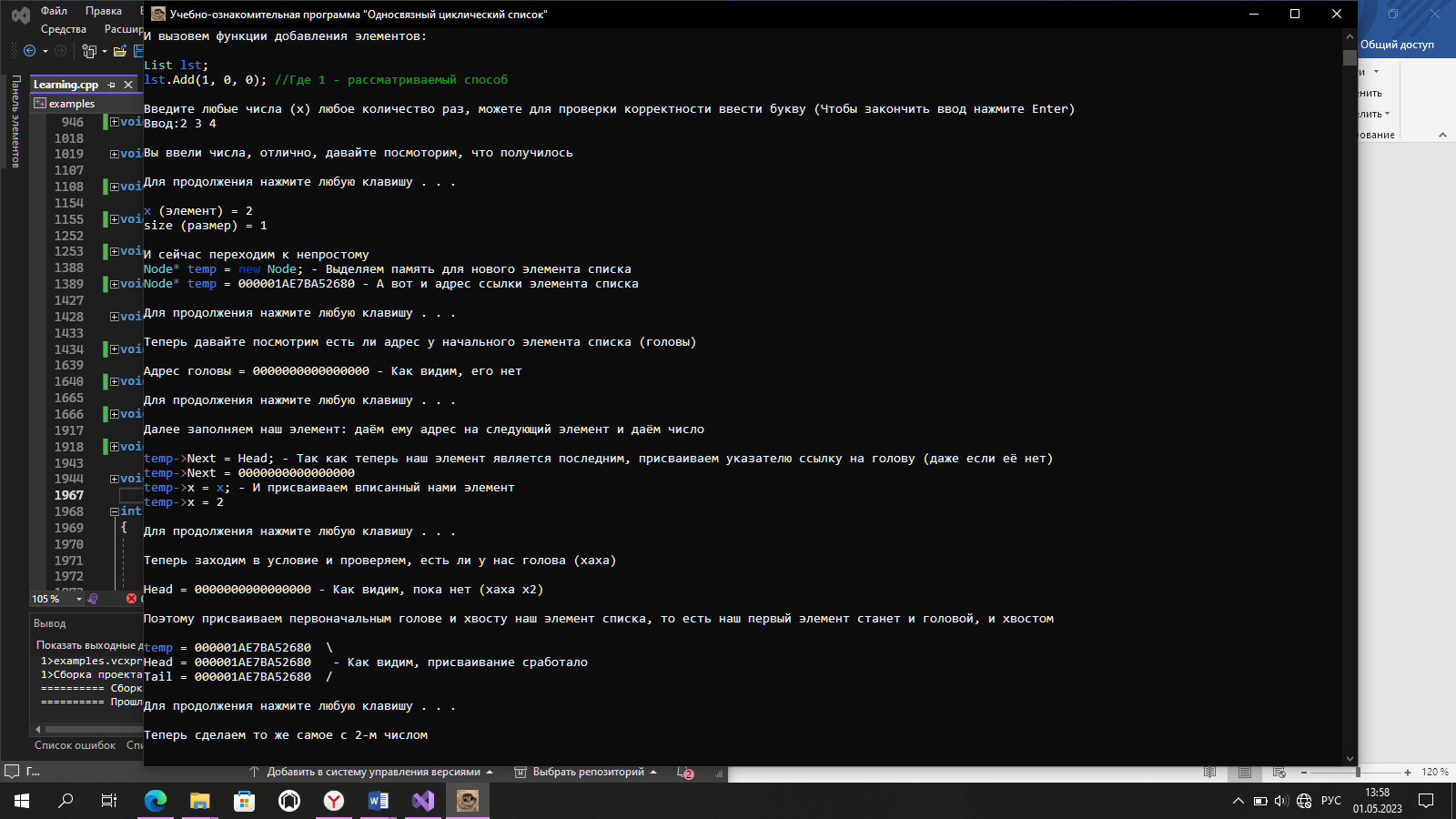


Рисунок 2.6 – Фрагмент практической части темы о добавлении элемента в конец списка

После каждой пройденной темы пользователю даётся список действий (рисунок 2.7), во время практической части некоторых тем, если список пустой, пользователю даётся выбор, связанные с заполнением списка (рисунок 2.8). После практической части таких тем пользователю также даётся выбор на повтор практической части (рисунок 2.9).

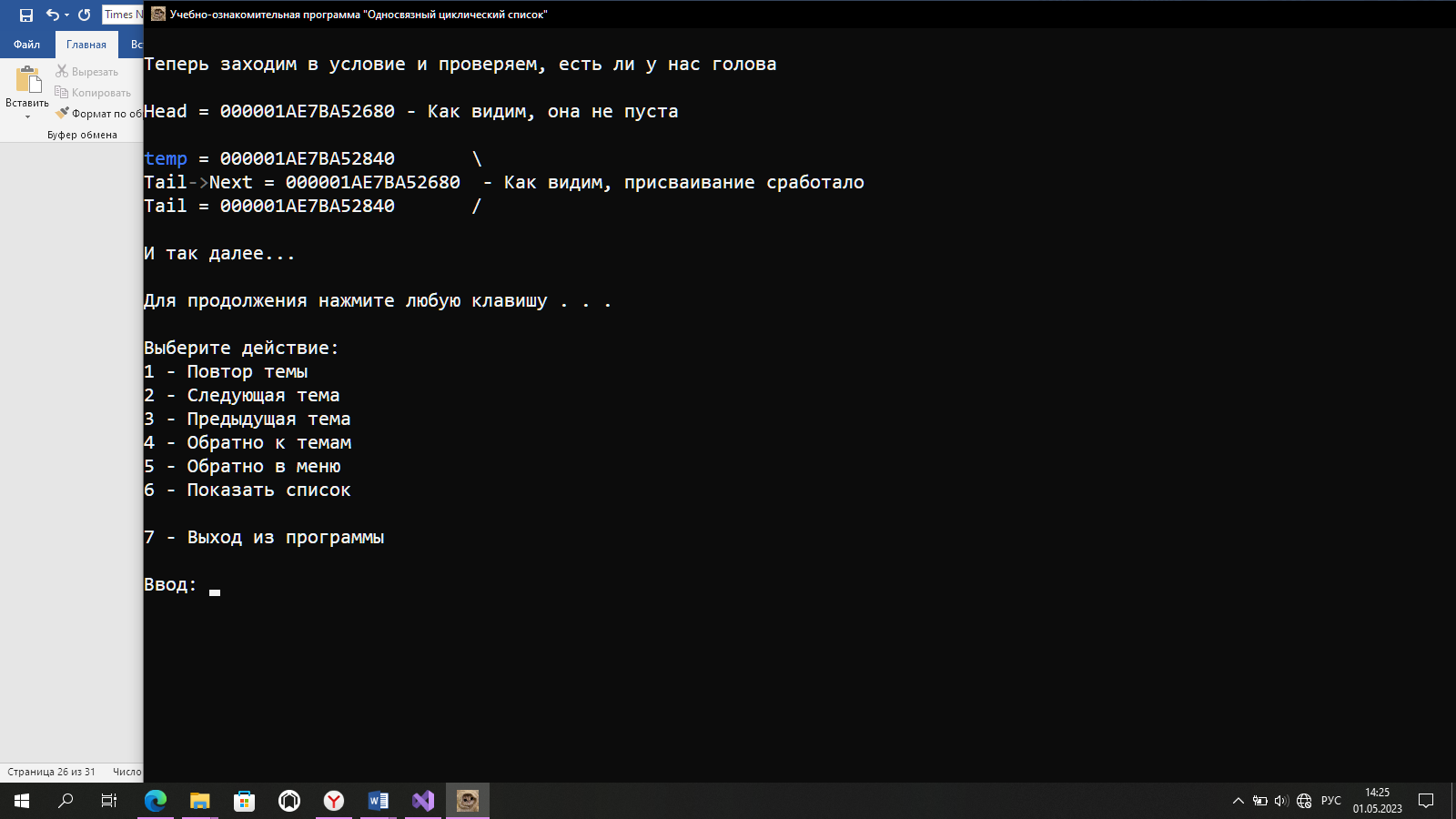


Рисунок 2.7 – Список действий после завершения темы

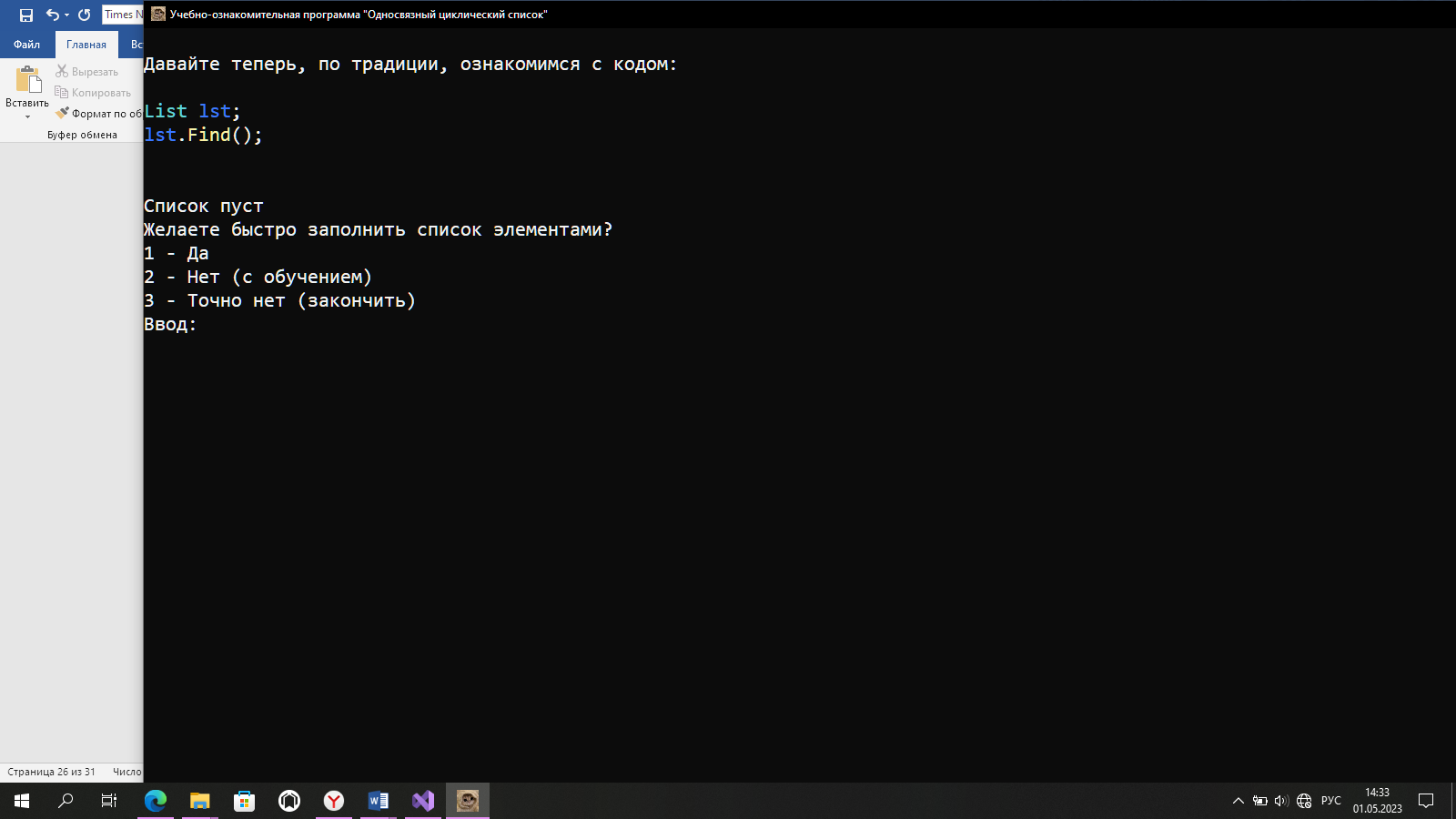


Рисунок 2.8 – Выбор заполнения списка во время практической части тем

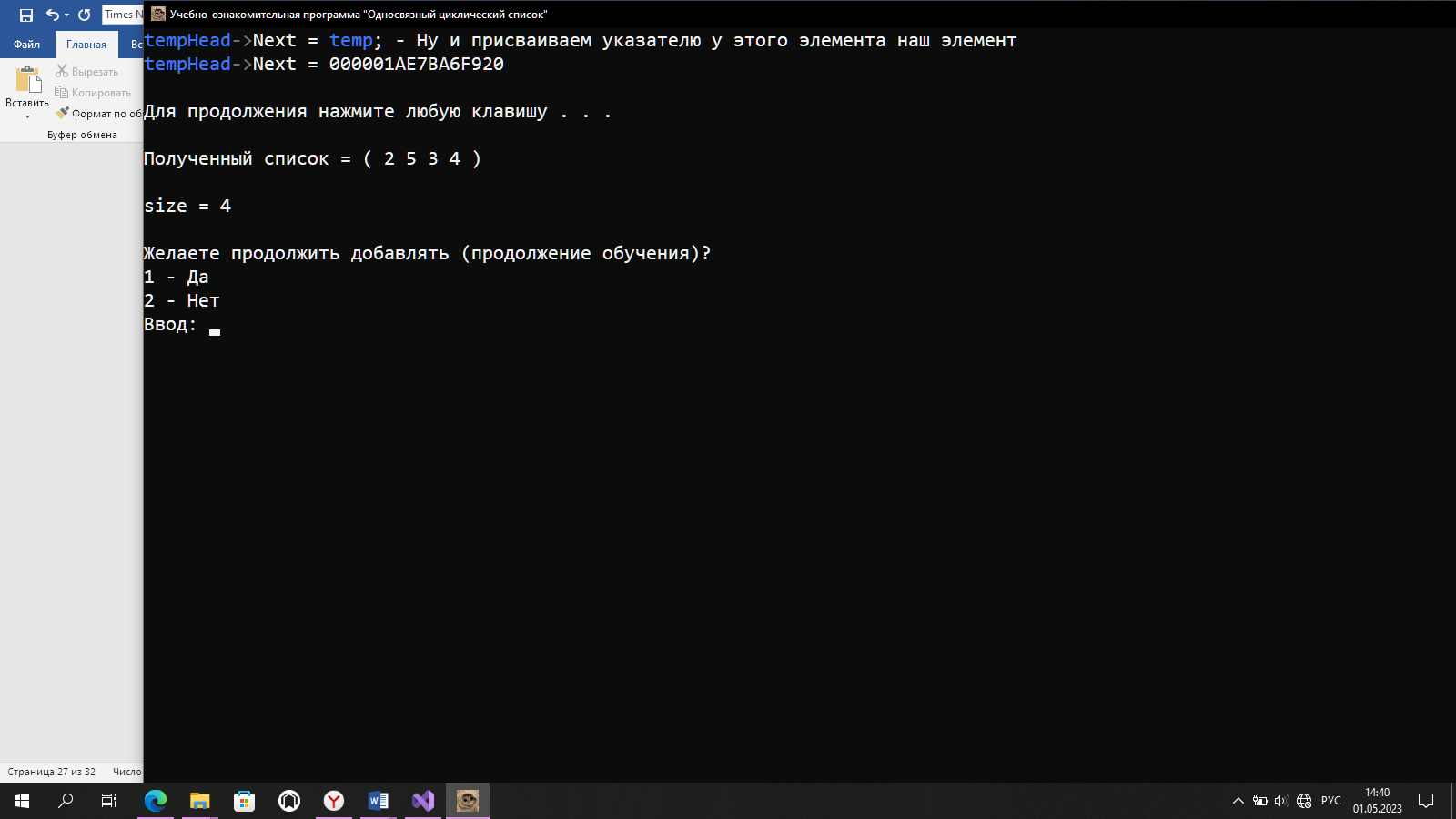


Рисунок 2.9 – Выбор продолжения обучения после практической части тем

После прохождения всех тем пользователю даётся доступ к тесту, который можно пройти в любой момент и неограниченное число раз, и чит-код к нему (рисунок 2.10).

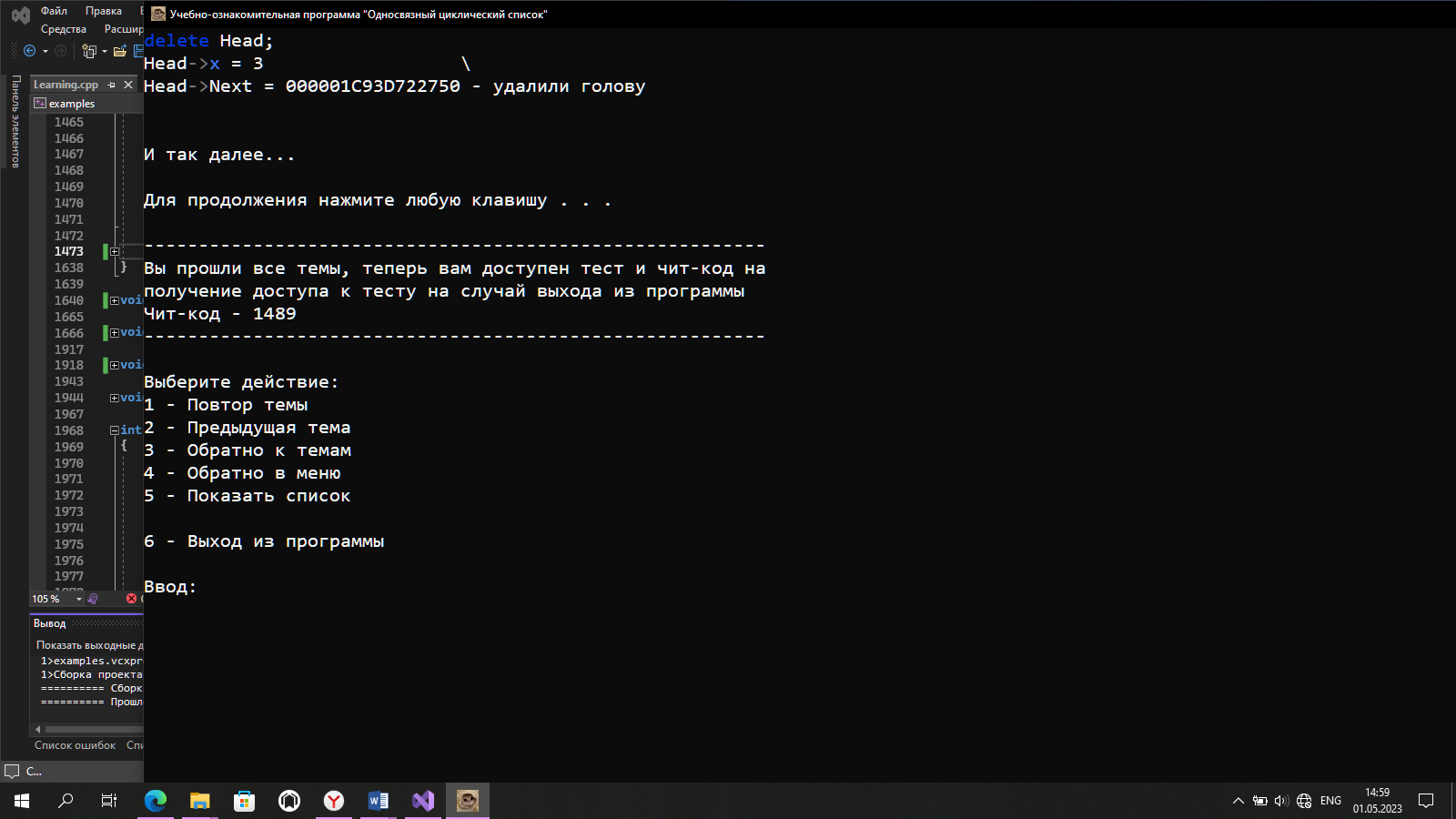


Рисунок 2.10 – Вывод сообщения о получении доступа к тесту

На данном этапе изучения пользователь может начать проходить тест без каких-либо ограничений, выбрав в меню пункт «Тест», и получить 9 вопросов с одним правильным ответом (рисунок 2.11), после прохождения которых получить результат с количеством баллов, оценкой, выводом верных и неверных вопросов и рекомендацией (рисунок 2.12).

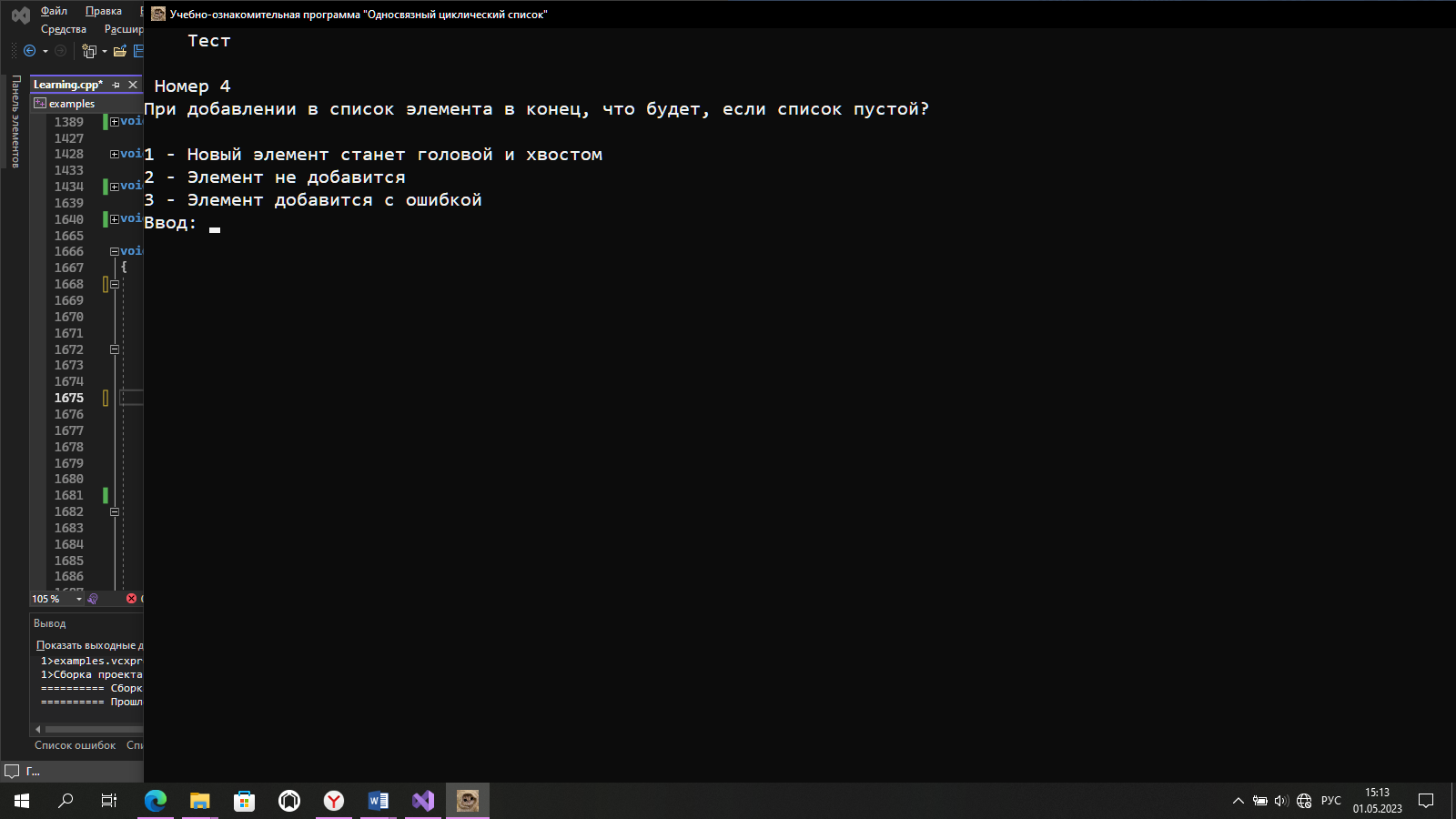


Рисунок 2.11 – Пример вопроса из теста

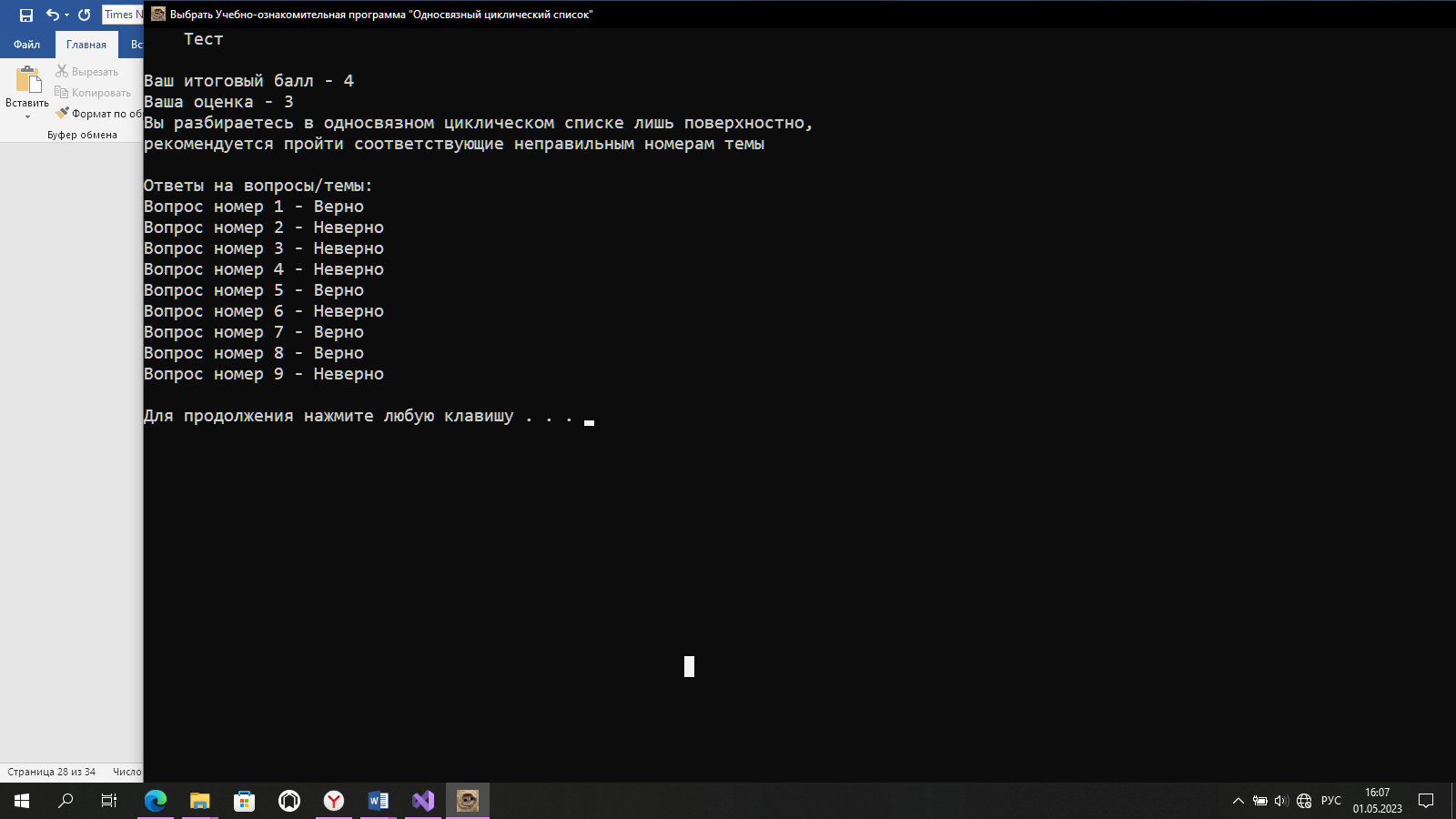


Рисунок 2.11 – Результат теста

## 2.6 Сообщение системы

В таблице 2.5 приведены сообщения системы.

**Таблица 2.5 – Сообщения системы**

|  |  |  |
| --- | --- | --- |
| **№ п\п** | **Сообщение** | **Причина возникновения** |
| 1 | size = 0 - В списке нет элементов, поэтому работа деструктора невозможна по условию самого деструктора ;) | Пользователь выбрал тему удаления списка с пустым списком |
| 2 | Пройдите все темы, и тогда вы получите доступ к тесту | Пользователь выбрал в меню тест со значением переменной check меньше 9 |
| 3 | Действие отсутствует  Введите повторно: | Пользователь ввёл число, не входящее в диапазон количества действий |
| 4 | Это не число  Введите повторно: | Пользователь ввёл не число |
| 5 | Чит-код активирован | Пользователь ввёл в меню число 1489 |

**Продолжение таблицы 2.5**

|  |  |  |
| --- | --- | --- |
| 6 | Элемент отсутствует | Пользователь ввёл данные, которые ни один элемент не имеет, или номер позиции несуществующего элемента в списке |
| 7 | Неправильная позиция | Пользователь в практической части темы «добавления элемента между элементами списка» ввёл число меньше 2 или больше размера списка |
| 8 | Ваш итоговый балл – 8-9  Ваша оценка – 5  Вы усвоили в полной мере каждую тему и можете работать самостоятельно с односвязными циклическими списками | Пользователь прошёл тест на 8 или 9 баллов |
| 9 | Ваш итоговый балл – 5-7  Ваша оценка – 4  Вам понятны большинство аспектов односвязного циклического списка, но для полной уверенности пройдите соответствующие неправильным номерам темы | Пользователь прошёл тест на 5, 6 или 7 баллов |
| 10 | Ваш итоговый балл – 3-4  Ваша оценка – 3  Вы разбираетесь в односвязном циклическом списке лишь поверхностно, рекомендуется пройти соответствующие неправильным номерам темы | Пользователь прошёл тест на 3 или 4 балла |
| 11 | Ваш итоговый балл – 0-2  Ваша оценка – 3  Вы не усвоили изученный материал, настоятельно рекомендуется пройти все темы заново | Пользователь прошёл тест на 2 балла и меньше |

В случае появления других сообщений следует обратиться к разработчику.

# 3 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

## 3.1 Проверка работоспособности тем

1. Запустить программу на выполнение. Появится меню (см. рисунок 2.1);
2. выбрать пункт 2, убедиться, что выведен список тем (см. рисунок 2.3);
3. выбрать пункт 4, убедиться, что выведен теоретический материал (см. рисунок 2.5);
4. просмотреть материал и дойти до практической части;
5. ввести через пробел «д», 3, 4, убедиться, что это предусмотрено программой, и после ввести через пробел 2, 3, 4 (см. рисунок 2.6);
6. после окончания темы убедиться в появлении списка действий (см. рисунок 2.7);
7. выбрать пункт 6 и убедиться в наличии набранных элементов;
8. выбрать пункт 2 и убедиться в появлении теоретического и практического материала темы;
9. ввести последовательно «й», 4, 1, убедиться в ошибке;
10. ввести 3, ввести 45, убедиться в продолжении практической части темы и после её окончания появления выбора продолжения (см. рисунок 2.9);
11. ввести 4, убедиться в появлении списка тем (см. рисунок 2.3);
12. выбрать пункт 9, убедиться в появлении темы;
13. после прохождения темы убедиться в появлении списка действий (см. рисунок 2.7);
14. выбрать пункт 5 и убедиться в пустом списке;
15. ввести 3, убедиться в появлении списка тем (см. рисунок 2.3);
16. выбрать пункт 8 и убедиться в появлении темы;
17. в практической части убедиться в появлении сообщения о пустом списке и выбора о заполнении (см. рисунок 2.8);
18. выбрать пункт 1 и убедиться в появлении ожидания ввода чисел, как на шаге 5;
19. ввести через пробел 5,6,7 и убедиться в появлении выбора способа удаления;
20. выбрать пункт 2, ввести 5 и убедиться в продолжении практической части;
21. после практической части убедиться в появлении выбора в повторении практической части:
22. выбрать пункт 2, убедиться в появлении списка действий (см. рисунок 2.7);
23. выбрать пункт 3, убедиться в появлении темы поиска;
24. в практической части убедиться в появлении выбора способа поиска;
25. выбрать пункт 1, ввести 1, убедиться в продолжении практической части;
26. после практической части убедиться в появлении выбора в повторении практической части;
27. выбрать пункт 2, убедиться в появлении списка действий (см. рисунок 2.7);
28. выбрать пункт 3, убедиться в появлении темы вывода списка;
29. после вывода списка убедиться в появлении списка действий, после чего выбрать пункт 7.

## 3.1 Проверка работоспособности теста

1. Запустить программу на выполнение. Появится меню (см. рисунок 2.1);
2. выбрать пункт 3, убедиться в появлении сообщения о прохождении всех тем и в возвращении в меню после сообщения;
3. ввести 1489, убедиться в появлении сообщения об активации чит-кода;
4. выбрать пункт 3, убедиться в появлении первого вопроса теста;
5. ввести последовательно 2, 1, 1, 2, 2, 3, 2, 1, 2;
6. убедиться в появлении результатов как на рисунке 2.11 и в возвращении обратно в меню после паузы;

# ЗАКЛЮЧЕНИЕ

В результате курсового проектирования разработана учебно-ознакомительная программа "Односвязный циклический список". Программа позволяет обучиться односвязным циклическим спискам на языке программирования «С++» благодаря грамотно написанному теоретическому и практическому материалам и свободному перемещению между темами неограниченное количество раз. Такое перемещение позволяет просмотреть множество исходов и нюансов в односвязном циклическом списке без каких-либо ошибок и аварийных выходов из программы. После прохождения всех тем программа предоставляет доступ к тесту, который можно пройти неограниченное количество раз, имея возможность пройти темы снова, и в результате которого, в зависимости от оценки, даются рекомендации.

Программа отвечает поставленным требованиям и может быть использована для обучения студентов высших учебных заведений.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Е.В. Симонова. Структуры данных. Часть I. Линейные динамические структуры. – Самара.: Издательство СГАУ, 2006. 101 с.
2. Мамлеева А.Р., Евсина Е.М., Кравченкова Е.П. Требования к оформлению пояснительной записки при выполнении отчётов по различным видам практик. – Астрахань.: Астраханский государственный технический университет, 2022. 50 с.
3. Связный список. Википедия – [Электронный ресурс] режим доступа: <https://ru.wikipedia.org/wiki/Связный_список#:~:text=Свя́зный%20спи́сок%20—%20базовая%20динамическая,явно%20задаётся%20его%20внутренними%20связями> (25.01.2023).
4. С++ для начинающих. Кольцевой Однонаправленный список FIFO – [Электронный ресурс] режим доступа: <https://ciplusplussnachala.wordpress.com/2012/06/21/с-для-начинающих-кольцевой-однонапра/> (25.01.2023).

# ПРИЛОЖЕНИЕ 1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ

**на выполнение курсового проекта**

по дисциплине «Основы алгоритмизации и программирования**»**

Направление 090301 – Информатика и вычислительная техника

Исполнитель: обучающийся гр. ДИНРб11 **Самодуров В.А.**

Тема: **Учебно-ознакомительная программа «Односвязный циклический список»**

1. **Назначение, цели и задачи разработки**

**Цель разработки** – понятное и **малозатратное** обучение односвязным циклическим спискам

**Назначение разработки:**

* повышение качества знаний студентов;
* снижение затрат сил и времени у студентов.

**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.

1. **Характер разработки:** прикладная квалификационная работа.
2. **Основания для разработки**
   * Учебный план направления 09.03.01 «Информатика и вычислительная техника» 2022 года набора.
   * Рабочая программа дисциплины «Информатика и вычислительная техника».
   * Распоряжение по кафедре АСОИУ №\_\_\_\_ от «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2022 г.
3. **Плановые сроки выполнения –** весенний семестр 2023 учебного года:

Начало «13» февраля 2023 г.

Окончание «7» июня 2023 г.

1. **Требования к проектируемой системе**

**5.1 Требования к функциональным характеристикам**

Проектируемая система должна обеспечивать выполнение следующих основных функций:

* + предоставлять пользователю визуализацию процесса любой операции;
  + запрашивать у пользователя входные данные;
  + прекращать визуализацию по желанию пользователя.

Система должна предусматривать функциональные ограничения:

* + входные данные должны быть целыми числами;
  + программа не может использоваться в качестве электронной книги.

**5.3 Требования к эксплуатационным характеристикам**

Программа не должна аварийно завершаться при любых действиях пользователя.

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

**5.4 Требования к программному обеспечению:**

Среда разработки – Microsoft Visual Studio (С++) (Версия 17.4.1).

Операционная система: Windows 10 (x64), версия 21H2, сборка ОС 19044.2486.

**5.5 Требования к аппаратному обеспечению:**

Рекомендуемая конфигурация:

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

1. **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.

1. **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.

1. **Порядок контроля и приемки**

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

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 практики.

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

1. **Литература**
2. Оператор косвенного обращения: «\*». Microsoft Learn – [Электронный ресурс] режим доступа: <https://learn.microsoft.com/ru-ru/cpp/cpp/indirection-operator-star?view=msvc-170> (24.01.2023)
3. Операторы доступа к членам: «.» и «->». Microsoft Learn – [Электронный ресурс] режим доступа: <https://learn.microsoft.com/ru-ru/cpp/cpp/member-access-operators-dot-and?view=msvc-170> (24.01.2023)
4. Struct (C++). Microsoft Learn – [Электронный ресурс] режим доступа: <https://learn.microsoft.com/ru-ru/cpp/cpp/struct-cpp?view=msvc-170> (24.01.2023)
5. Деструкторы (C++). Microsoft Learn – [Электронный ресурс] режим доступа: <https://learn.microsoft.com/ru-ru/cpp/cpp/destructors-cpp?view=msvc-170> (25.01.2023)
6. Как закончить цикл. С++. Киберфорум – [Электронный ресурс] режим доступа: <https://www.cyberforum.ru/cpp-beginners/thread2245272.html> (25.01.2023)

# ПРИЛОЖЕНИЕ 2 БАЗА ВОПРОСОВ ДЛЯ ТЕСТИРОВАНИЯ

1. У списка может быть неограниченное число элементов, какие из них главные?
   1. Первый;
   2. голова и хвост;
   3. у кого информации больше.
2. Из каких параметров состоит структура списка?
   1. из данных;
   2. из встроенных параметров;
   3. из данных и указателей.
3. Что нужно сделать с начальными элементами списка в классе, чтобы начать работу с ним?
   1. назвать;
   2. объявить и проинициализировать в ноль;
   3. присвоить к имеющимся.
4. При добавлении в список элемента в конец, что будет, если список пустой?
   1. новый элемент станет головой и хвостом;
   2. элемент не добавится;
   3. элемент добавится с ошибкой.
5. Что произойдёт, если при добавлении в список элемента между другими элементами указать первую позицию?
   1. первый элемент сдвинется вправо, а добавленный станет головой;
   2. код будет ругаться;
   3. добавленный элемент вставиться после головы.
6. Вывод списка сразу осуществим?
   1. нет, нужно написать функцию, возвращающую размер;
   2. да;
   3. нет, нужно обеспечить целостность данных.
7. Как происходит поиск элемента?
   1. вбивается запрос и сразу выдаётся элемент, как у матрицы;
   2. вбивается запрос и выполняется перебор элементов, начиная с головы;
   3. вбивается запрос и выполняется метод Лагранжа.
8. Сколько нужно указателей, чтобы выполнить удаление элемента по данным без ошибок?
   1. 3;
   2. 1;
   3. 2.
9. Являются ли элементы списка динамическими объектами и что нужно делать с динамическими объектами после завершения работы с ними?
   1. нет, удалять их;
   2. да, ничего не делать;
   3. да, удалять их.