|  |  |
| --- | --- |
| *voenmeh* | Министерство науки и высшего образования Российской Федерации  федеральное государственное бюджетное образовательное учреждение  высшего образования  **«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»**  **(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова»)** |
| БГТУ.СМК-Ф-4.2-К5-01 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Факультет |  | И |  | Информационные и управляющие системы |
|  |  | шифр |  | Наименование |
| Кафедра |  | И5 |  | Информационные системы и вычислительная техника |
|  |  | шифр |  | Наименование |
| Дисциплина |  | Программирование на языке высокого уровня | | |

КУРСОВАЯ РАБОТА

на тему

|  |
| --- |
| Объектно-ориентированная разработка программ с |
| графическим пользовательским интерфейсом |
| «сверху-вниз» |
| Программа по второй лабораторной работе |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | | | |  | И903Б |
| Монахов Л. | | | | | |
| Фамилия И.О. | | | | | |
| **РУКОВОДИТЕЛЬ** | | | | | |
| Першин Д. В. | |  |  | | |
| Фамилия И.О. Подпись | | | | | |
| Оценка |  | | |  |  |
| «\_\_\_\_\_» |  | | |  | 2021 г. |

**Содержание**

[**ВВЕДЕНИЕ** 3](#_Toc83477201)

[**1.ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОПРЕДЕЛЕНИЙ** 4](#_Toc83477202)

[**Приложение** 5](#_Toc83477203)

[**1 Постановка задачи** 6](#_Toc83477204)

[**2 Кратко о классах в ооп** 7](#_Toc83477205)

[**3 Описание программы** 9](#_Toc83477206)

[2.1 класс «MyString» 9](#_Toc83477207)

[2.2 Класс «StringId» 9](#_Toc83477208)

[2.3 Класс «Bite» 10](#_Toc83477209)

[2.3 Класс «SDL\_Scope» 10](#_Toc83477210)

[2.3 Класс «MainWindow» 10](#_Toc83477211)

[2.3 Класс «StateManager» 11](#_Toc83477212)

[2.3 Класс «State» 11](#_Toc83477213)

[2.3 Класс «Menu» 11](#_Toc83477214)

[2.3 Класс «InputField» 12](#_Toc83477215)

[2.3 Класс «TextMessage» 12](#_Toc83477216)

[2.3 Класс «StringChooseMenu» 12](#_Toc83477217)

[2.3 Класс «StringManager» 13](#_Toc83477218)

[**3 Результат работы программы** 15](#_Toc83477219)

[**ЗАКЛЮЧЕНИЕ** 19](#_Toc83477220)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 20](#_Toc83477221)

# **ВВЕДЕНИЕ**

Программирование занимает одну из значительных ниш в современном мире. Это область деятельности, направленная на разработку программного обеспечения с целью улучшения и облегчения человеческой жизни, промышленной деятельности, сферы услуг и технологического прогресса. Программирование позволяет создавать множество различных программ для решения задач, касающихся большого количества сфер деятельности. С помощью программирования можно налаживать эффективную работу множества процессов и создавать целостные продукты для массового потребления. Программы являются одним из таких продуктов. Создание Программы – часто довольно трудоемкая задача. Основная сложность заключается в осуществлении взаимодействия между собой различных элементов программы в реальном времени под управлением пользователя, также программист должен следить за грамотным расходованием памяти, при создании и удалении объектов в программе, с целью избежать утечек памяти и т.д. Рассмотрим пример программы на основе второй лабораторной работы, которая состоит из множества классов, логически взаимодействующих между собой, образующих иерархию.

# **1.ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОПРЕДЕЛЕНИЙ**

В пояснительной записке к курсовой работе используются следующие сокращения и обозначения.

ООП – объектно-ориентированное программирование

# **Приложение**

Исходный текст программы Исходные тексты программы и остальные файлы, требуемые для успешной компиляции прилагаются в каталоге «Курсач 2.1» в электронном виде.

# **1 Постановка задачи**

Целью курсовой работы является разработка программы с библиотекой SDL 2.0 на основе лабораторной работы № 2, задача которой состоит в создании трех классов – один базовый, два класса - наследника.

На первом этапе необходимо ознакомиться с поставленной задачей и обговорить с преподавателем особенности реализации.

Задача включает в себя следующее:

* решение задачи с помощью объектно-ориентированного программирования;
* построение иерархии классов, описывающих предметную область данной задачи;
* использование языка программирования C++ (с элементами С);
* использование мультимедийной библиотеки SDL 2.0.

К разрабатываемой программе предъявлены следующие требования:

* использование наследование;
* использование мультимедийной библиотеки SDL 2.0.

На втором этапе необходимо ознакомиться с инструментами разработки. Предварительное изучение библиотеки SDL 2.0 позволит понять основные принципы работы с графикой в данной библиотеке и существенно сократить время написания программы.

На третьем этапе необходимо приступить к описанию логик программы. Программа будет реализована с помощью словаря map, библиотеки SDL 2.0, классов.

На четвертом этапе необходимо добавить пользовательский интерфейс, включающий меню и удобные средства для управления программой.

# **2 Кратко о классах в ооп**

В данной работе активно используются классы поэтому нельзя кратко о них не упомянуть.

Класс– в ООП, представляет собой шаблон для создания объектов, обеспечивающий начальные значения состояний таких как инициализация полей-переменных и реализация

Наследование – это механизм создания нового класса на основе уже существующего. При этом к существующему классу могут быть добавлены новые элементы (данные и функции), либо существующие функции могут быть изменены. Основное назначение механизма наследования — повторное использование кодов, так как большинство используемых типов данных являются вариантами друг друга, и писать для каждого свой класс нецелесообразно [1]. Схематичное изображение примера наследования для лучшего понимания можно увидеть на рисунке 1.

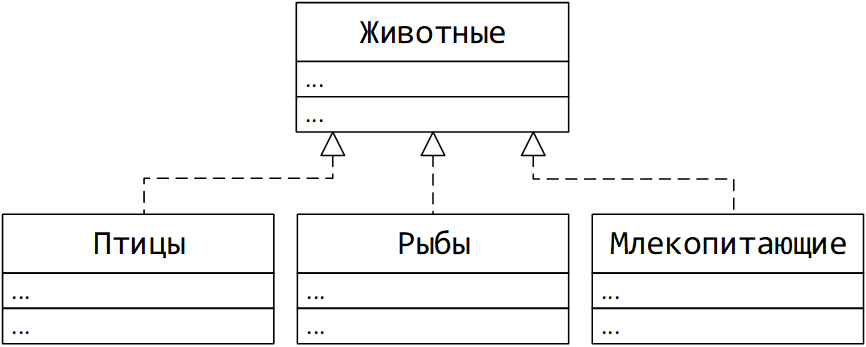


Рисунок 1 – Схематичное изображение наследования

Инкапсуляция – размещение в одном компоненте данных и методов, которые с ними работают. Также может означать скрытие внутренней реализации от других компонентов, что разграничивает доступ к различным компонентам программы.

Полиморфизм – способность объекта реагировать на запрос сообразно своему типу, при этом одно и то же имя метода может использоваться для различных классов объектов. Как правило, полиморфизм возникает, когда существует иерархия классов, и они связаны наследованием.

# **3 Описание программы**

В данной работе активно используются классы поэтому нельзя кратко о них не упомянуть.

За что отвечают классы

* классы для работы с SDL 2.0;
* класс для работы с классами “MyString”, “IdClass”, “BiteClass”;
* классы для работы с интерфейсом.

2.1 класс «MyString»

Предназначен для создание обычной строки, работы с ней и является базовым классом для “IdClass” и “BiteClass”.

Содержит поля:

* поле counter1 типа int, описывает количество вызовов конструктора копирования;
* поле str типа char\*, описывает блок динамически выделенной памяти для размещения символов строки;
* поле length типа int, описывает длину строки;

Содержит методы:

* метод virtual типа int без параметров и возвращаемым типом STRINGTYPE, предназначен для определения типа объекта;
* метод Getlength() типа int без параметров и возвращаемым типом int, предназначен для возвращения длины строки;

2.2 Класс «StringId»

Предназначен для создания строки-идентификатора, работы с ней и является наследником класса MyString.

Содержит поля:

* поле counter1 типа int, описывает количество вызовов конструктора копирования;

Содержит методы:

* метод virtual типа int без параметров и возвращаемым типом IDTYPE, предназначен для определения типа объекта;
* метод uppercase() типа void без параметров предназначен для перевода строки в верхний регистр;
* метод CheckIdentifier() типа int без параметров и возвращаемом типом int предназначен для определения строки, является ли она идентификатором или нет;

2.3 Класс «Bite»

Предназначен для создания битовой строки, работы с ней и является наследником класса MyString.

Содержит поля:

* поле counter типа int, описывает количество вызовов конструктора копирования;

Содержит методы:

* метод virtual типа int без параметров и возвращаемым типом BITETYPE, предназначен для определения типа объекта;
* метод revers() типа void без параметров предназначен для перевода битовой в строки в дополнительный код;
* метод Check\_Bite() типа int без параметров и возвращаемом типом int предназначен для определения строки, является ли она битовой или нет;

2.3 Класс «SDL\_Scope»

Предназначен для создания SDL окна и проверки на корректность работы.

2.3 Класс «MainWindow»

Предназначен для создания вида окна.

Содержит поля:

* поле \_window типа SDL\_Window\*, описывает шаблон окна;
* поле \_renderer типа SDL\_Renderer \*, описывает визуализацию;
* поле \_w типа int, описывает ширины окна.
* поле \_h типа int, описывает высоту окна.

Содержит методы:

* метод init() типа void с параметрами char title, int w, int h, предназначен для определения типа объекта;
* метод deinit() типа void без параметров предназначен для уничтожения окна;
* метод fillBackground типа void без параметров предназначен для задание цвета фона;
* метод drawText() типа SDL\_Rect без параметров предназначен для задание цвета текста и рисования текста в окне;

2.3 Класс «StateManager»

Предназначен для реализации словаря и навигации в меню.

Содержит поля:

* поле states типа map, описывает словарь;

Содержит методы:

* метод exec типа void с параметром startStateName, предназначен для навигации в меню, какой пункт меню следует за каким;

2.3 Класс «State»

Предназначен для запоминания откуда мы пришли в данный пункт меню.

Содержит поля:

* поле from типа string, запоминает откуда мы пришли;

Содержит методы:

* метод setForm типа void без параметров, предназначен для запоминания откуда мы пришли в данный пункт меню;

2.3 Класс «Menu»

Предназначен для отрисовки меню и взаимодействия с ним.

Содержит поля:

* поле items типа vector, отвечает за хранение текста;
* поле wayback типа vector, за обратный путь после выбора пункта меню;

Содержит методы:

* метод exec типа void без параметров, предназначен для выбора пункта меню и вывод следующего пункта в зависимости от выбора настоящего пункта меню пользователем;

2.3 Класс «InputField»

Предназначен для ввода.

Содержит поля:

* поле in типа string, отвечает за cохранения строки, введеной пользователем;
* поле message типа string, отвечает за вывод сопровождающего сообщения, которое объясняет что надо ввести пользователю;
* поле wayback типа string, отвечает за обратный путь, который будет после ввода корректной строки.

Содержит методы:

* метод exec типа void без параметров, предназначен для ввода пользователем строки;

2.3 Класс «TextMessage»

Предназначен для вывода сообщения.

Содержит поля:

* поле message типа string, отвечает за вывод сообщения, которое объясняет что надо ввести пользователю;
* поле wayback типа string, отвечает за обратный путь, который будет после ввода корректной строки.
* поле \*getMessage типа string, отвечает за хранения строки, которую выводи

Содержит методы:

* метод exec типа void без параметров, предназначен для вывода после работы со строкой;

2.3 Класс «StringChooseMenu»

Предназначен для выбора строки с которой работать.

Содержит поля:

* поле type типа int, отвечает за тип строки;
* поле wayback типа string, отвечает за обратный путь, который будет после ввода корректной строки.

Содержит методы:

* метод exec типа void без параметров, предназначен для выбора строки определенного типа (IDTYPE, BITETYPE, STRINGTYPE);

2.3 Класс «StringManager»

Предназначен для работы с классами лабораторной работы №2 (MyString, stringed, Bite).

Содержит поля:

* поле maxSize типа int, отвечает за количество строк;
* поле message типа string, отвечает за сообщение, которое будет выводиться;
* поле strings типа vector, отвечает за хранения строк.
* поле currentString типа MyString, отвечает за работу со строкой, с которой собираются сейчас взаимодействовать;
* поле errorMessage типа string, отвечает за хранение и вывод сообщения об ошибке;
* поле supString типа string, отвечает за хранения строки, которую пользователь вводи для работы с изначальной строкой. К примеру, сложить строки, для этого пользователь вводи строку, которую хочет сложить с изначальной.

Содержит методы:

* метод setSupString типа bool с параметром str, предназначен для записи строки supString;
* метод getBiteAdding типа string, предназначен сложение битовых строк;
* метод getBiteCompare типа string, предназначен для сравнения битовых строк на идентичность;
* метод ChangeString типа string, предназначен для замены строки;
* метод getIdAdding типа string, предназначен для сложения строк-идентификаторов;
* метод getUperrCase, предназначен для перевода строки в верхний регистр;
* метод creatBiteString, предназначен для создания битовой строки;
* метод creatIdString, предназначен для создания строки-идентификатора;
* метод creatString, предназначен для создания обычной строки;
* метод setCurrentString, предназначен для выбора строки с которой работать;
* метод setSize, предназначен для установки количества начальных строк;
* метод getLength, предназначен для вывода длины строки;

# **3 Результат работы программы**

При запуске программы. На рисунке 1 видим начальное меню.

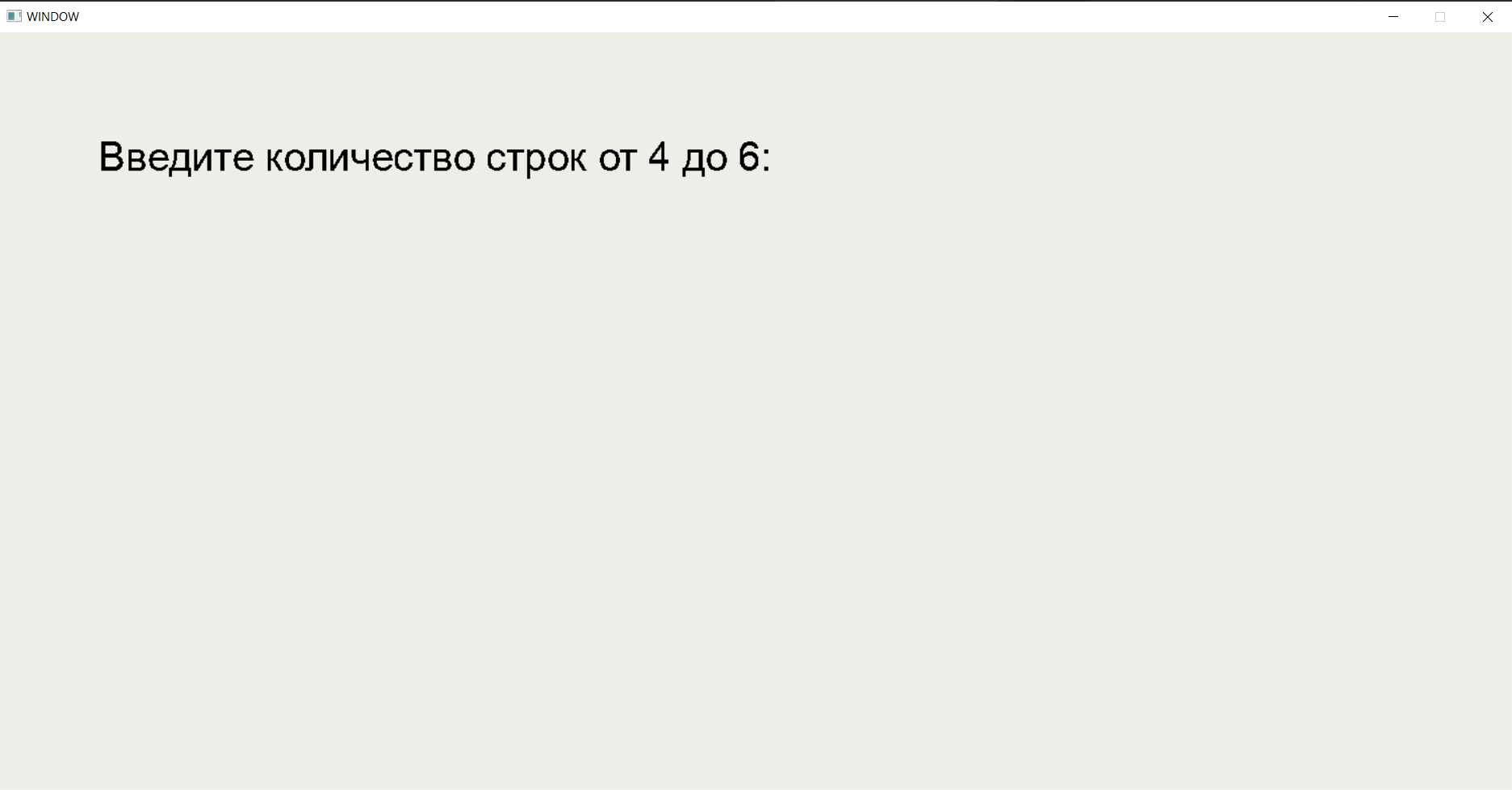


Рисунок 1

После ввода количества начальных строк переходим во главное меню, где выбираем что делать.

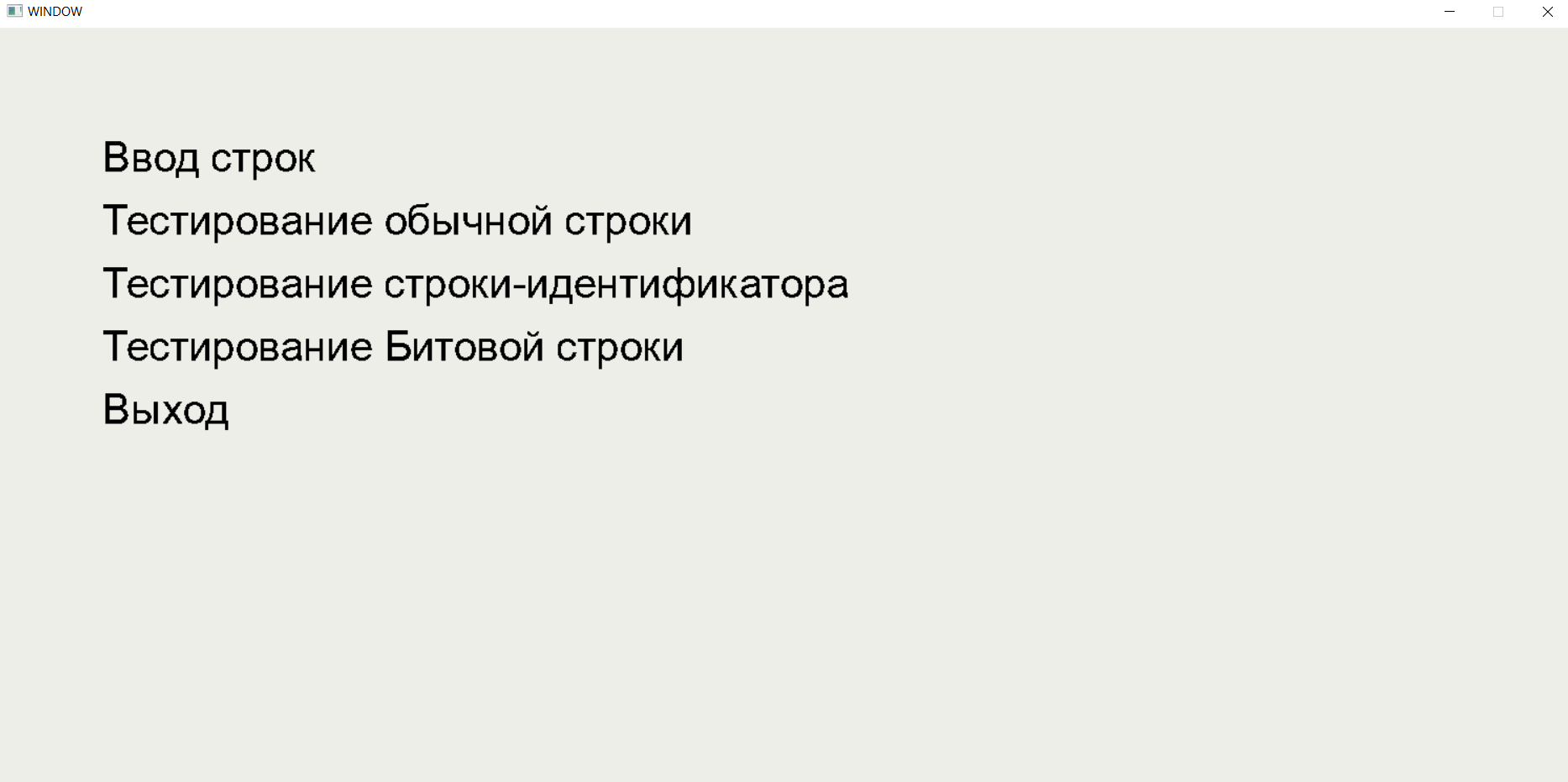


Рисунок 2

После выбора пункта “Ввод строк” в главном меню переходим в меню с выбором типа строки для ввода, показано на картинке 3.

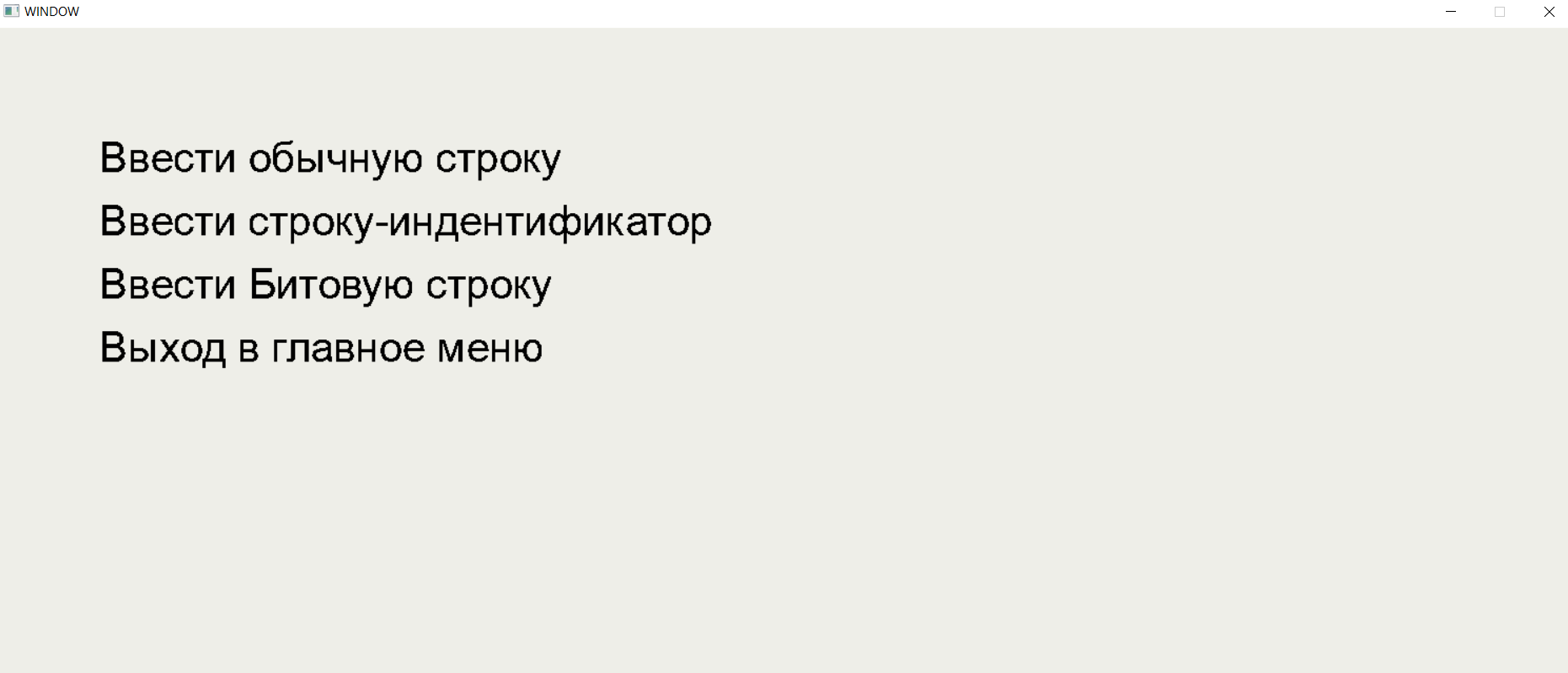


Рисунок 3

После выбора любого пункта меню с выбором типа строки вводим строку данного типа. Показано на рисунке 4

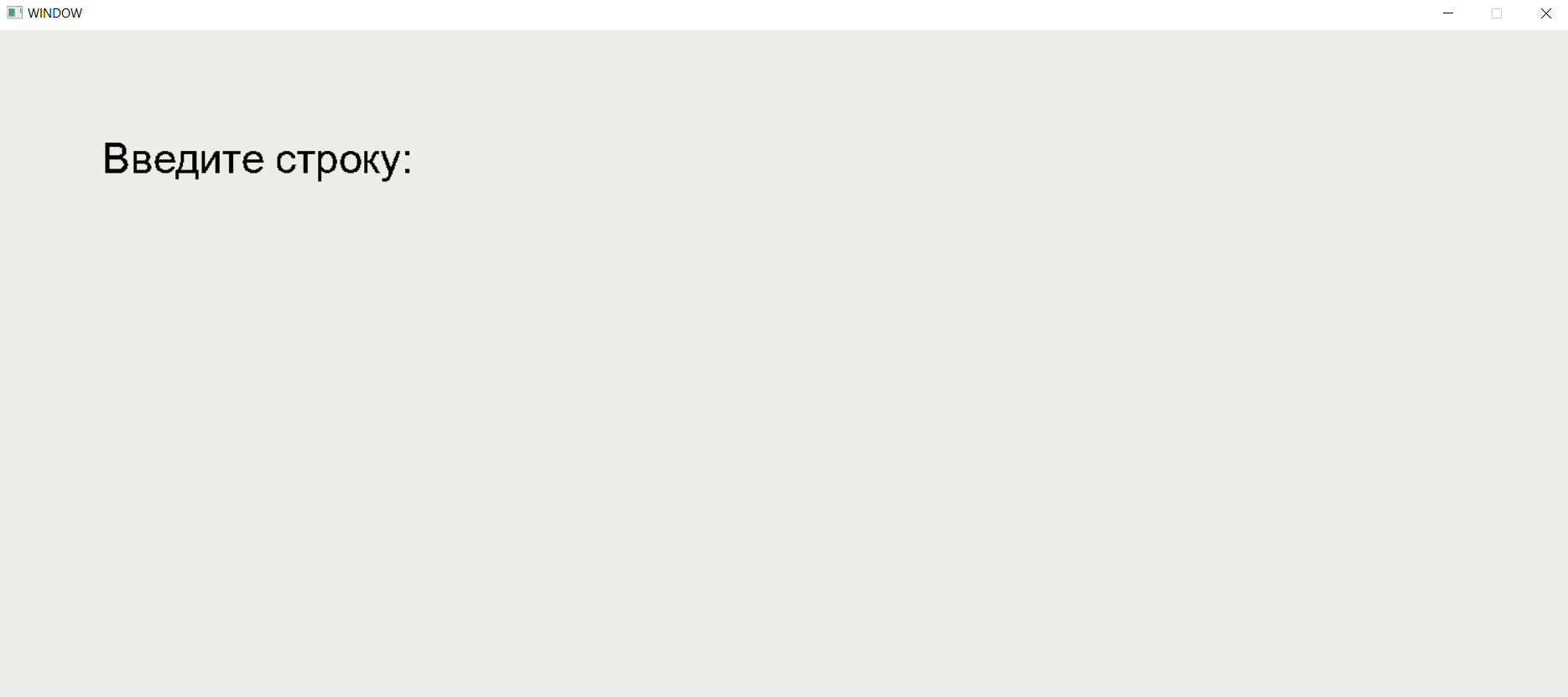


Рисунок 4

После ввода возвращаемся в меню с выбором типа строки, можно нажать на пункт “Выйти в главное меню”. Далее можно выбрать любой пункт меню для взаимодействия со строкой.

Показано на рисунке 5

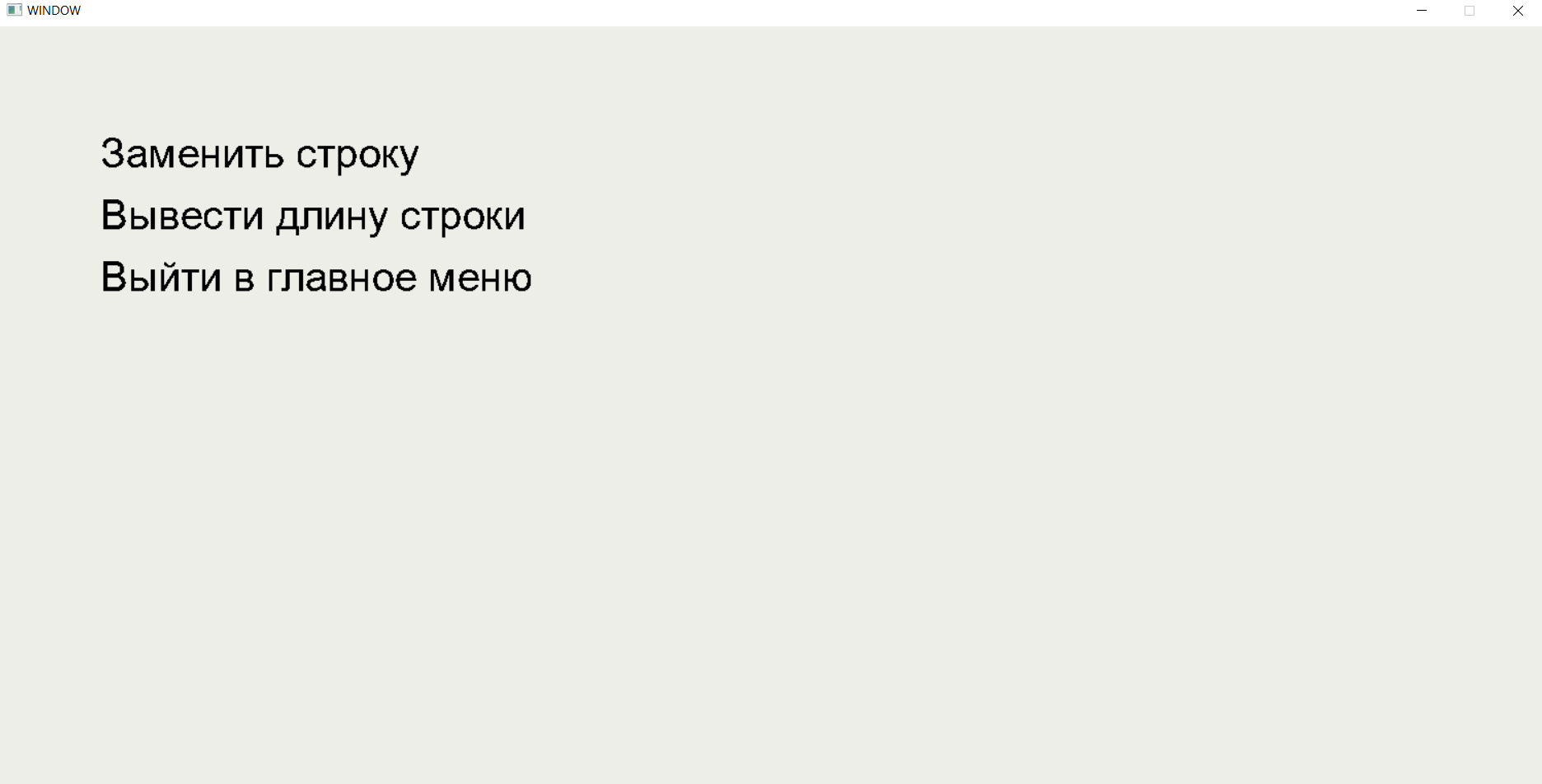


Рисунок 5

После выбора определенного пункта для взаимодействия со строкой предлагают выбрать строку для взаимодействия. Выводятся строки только данного типа строки, который вы выбрали. Показано на рисунке 6.

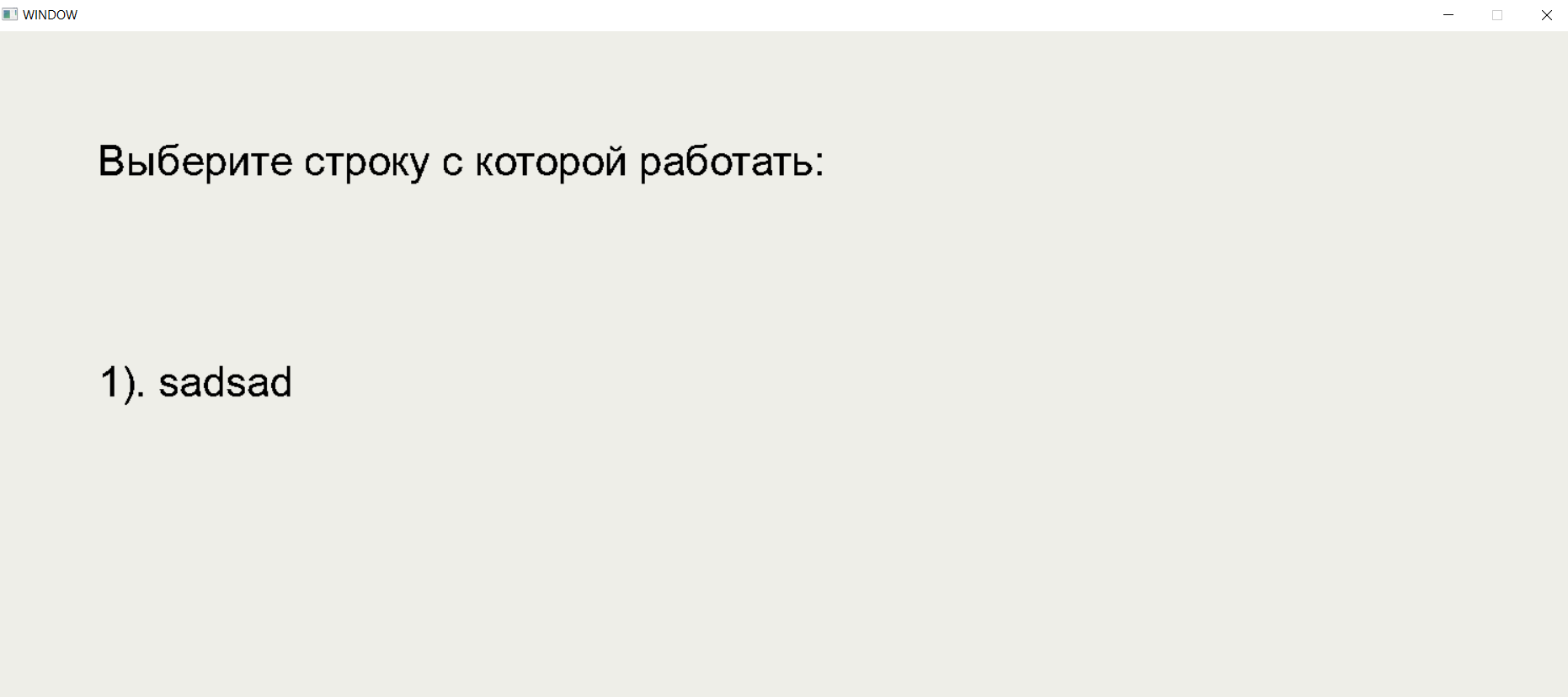


Рисунок 6

Далее работаем со строкой и получаем результат. На этом примере показана обычная строка с пунктом меню “Заменить строку”. Показано на рисунке 7.

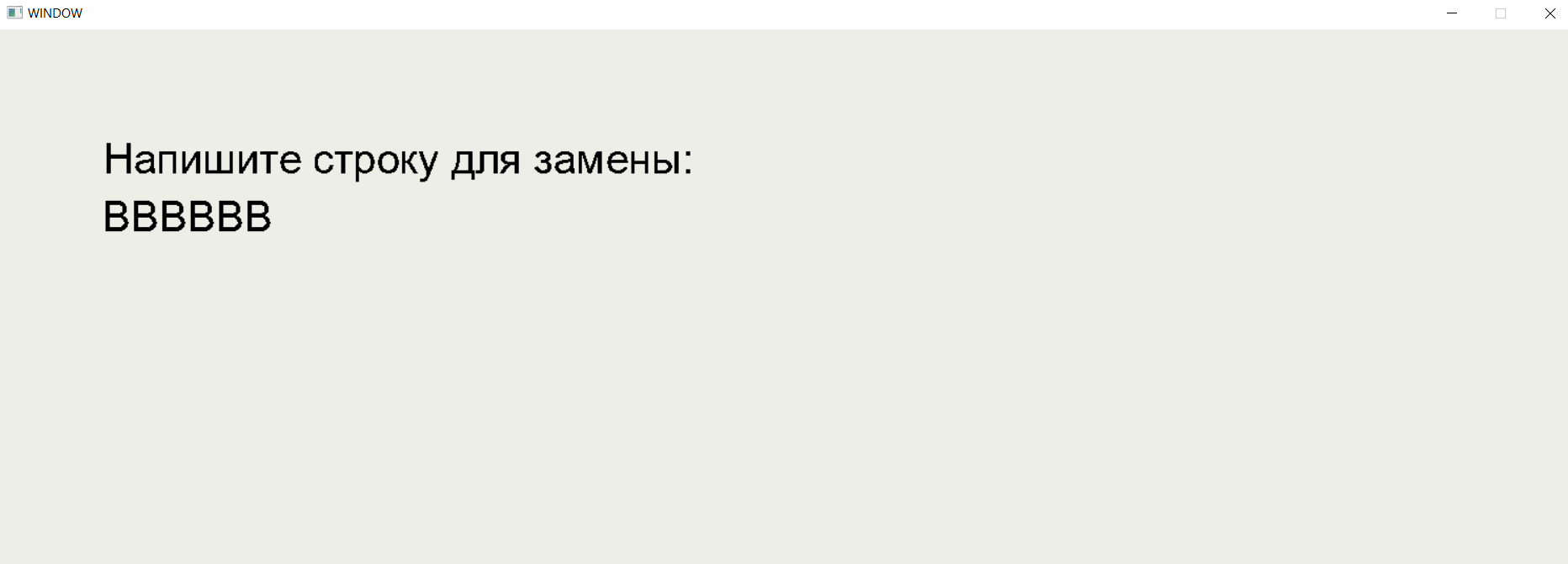


Рисунок 7

Далее получаем результат. Показано на рисунке 8.

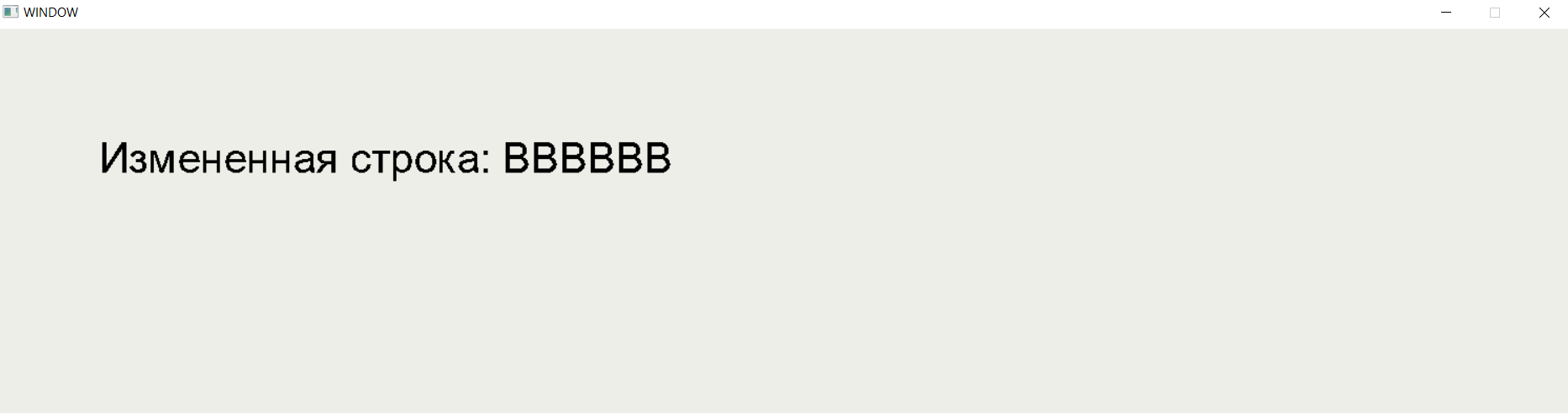


Рисунок 8.

# **ЗАКЛЮЧЕНИЕ**

В конечном итоге была выполнена цель курсовой работы – программа на основе лабораторной работы №2. В следствии разработки программы была построена и обоснована иерархия классов. Программа работает согласно поставленной задаче, требования к выполнению работы соблюдены. В программу можно внедрить новые возможности: добавить новые пункты меню. Так же можно сменить цветовое оформление при необходимости.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. 1. «Язык программирования C++. Лекции и упражнения» С. Прата – 6-ое издание; – М.: Вильямс, 2015. – 1248 с. 2. SDL Wiki [Электронный ресурс]
2. Документация по библиотеке SDL 2.0 URL: <https://wiki.libsdl.org/FrontPage> (Дата обращения: 08.06.2021). 3. Ravesli [Электронный ресурс]
3. Уроки программирования на языке С++ URL: <https://ravesli.com/uroki-cpp/> (Дата обращения: 10.06.2021). 4. Гущин, А.Н.
4. Применение библиотеки SDL для разработки программ на языке С: учебное пособие / БГТУ – СПб., 2014. – 118 с. 5. SDL Library Documentation. URL: www.libsdl.org/release/SDL-1.2.15/docs/html/index.html (дата обращения: 09.03.2019). 6. Страуструп, Б.
5. Язык программирования С++ / Б. Страуструп: пер. с англ. под ред. Н.Н. Мартынова. М..: Бином, 2011. 1135 с.