**Введение**

На курсовое проектирование была поставлена задача, разработка программы на тему: «Динамические структуры данных: «стек, очередь»

Цель курсового проекта заключается в обучении и закреплении знаний по стекам и очередям.

Данное приложение рассчитано на учащихся, которые только начинают изучать базовые принципы программирования.

Приведем краткое описание разделов пояснительной записки.

Первый раздел носит название «Анализ задачи». В нем можно ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи, определение ее организационно-экономической сущности. Также в этом разделе вы сможете узнать о том, как данная задача решается в настоящее время. Все входные и выходные данные тоже будут описаны в первом разделе. В подразделе «Инструменты разработки» будет рассмотрена среда, в которой создается данный курсовой проект. Здесь также будут установлены минимальные и оптимальные требования к аппаратным характеристикам, обеспечивающим функционирование поставленной задачи.

В разделе «Проектирование задачи» будут рассмотрены основные аспекты разработки программного продукта. Здесь можно будет узнать об организации данных в контексте среды разработки. В данном разделе будет четко описан пользовательский интерфейс, составлены алгоритмы процесса обработки информации, описана разработка системы справочной информации.

«Реализация задачи» – это третий раздел пояснительной записки, в котором описываются все элементы и объекты, которые будут использованы при реализации данного приложения. В этом разделе будут четко описаны функции пользователя и их структура. Здесь можно будет найти таблицу, в которой будет представлена полный список файлов используемых в данном проекте.

Четвертый раздел – «Тестирование». В нем будет тестирование данной программы, будет оттестирована каждая операция, которая выполняется программой.

В разделе «Применение» будет описано назначение, область применения, среда функционирования курсовой программы. Также в нем будет описано использование справочной системы.

«Заключение» будет содержать краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств, описание степени автоматизации процессов на различных этапах разработки.

В «Списке используемых источников» будет приведен список используемой литературы.

В приложениях к пояснительной записке будет приведен листинг программы с необходимыми комментариями.

Схема работы системы будет представлена в графической части.

1. **Анализ задачи**
   1. **Постановка задачи**

Темой данного курсового проекта является «Списковые структуры данных: «Стек, очередь».

Списковыми структурами данных называются структуры, которые основаны на списках и могут изменять свой размер в процессе работы программы

Списковые структуры данных дают возможность для более удобной и эффективной работы с данными, так как в отличии от статических структур, при работе с которыми зачастую выделяется слишком мало или же слишком много памяти, их размерность можно изменить в процессе работы программы .

Стек и очередь являются списковыми структурами данных, а если конкретней то – линейными списками.

Линейный список – это последовательность из нуля или более элементов, главной особенностью которого является такое относительное расположение узлов, будто они образуют одну линию.

Стек – линейный список, в котором добавление и исключение элементов производится на одном его конце. Стек легко себе представить в виде стопки книг. Ведь книгу, как и элемент в стеке, можно добавить только сверху, да и забрать можно лишь то, что лежит сверху.

Очередь – линейный список, в котором добавление элементов производится на одном его конце, а исключение (и, как правильно, доступ к ним) производится на другом его конце.

Доступ к элементам в стеках и очередях организован особым образом, что позволяет их использовать например в реализации различных алгоритмов.

Возможность изменения размера в списковых структурах по мере работы программы делает их универсальными, эффективными и подходящими для множества задач.

Над списковыми структурами можно выполнять различные операции:

* Создание структуры: инициализация новой структуры с выбранным типом данных;
* Добавление элементов: добавление нового элемента в «вершину» стека или в «хвост» очереди;
* Удаление элементов: удаление элемента с «вершины» стека или «головы» очереди;
* Доступ к элементам: вывод элементов стека или очереди с «вершины» или «головы» соответственно;
* Поиск элементов: поиск элемента по значению и возврат значения True или False;
* Изменение размера: изменение размера стека или очереди при необходимости.

Изучение принципа работы стека и очереди может понадобиться всем начинающим программистам. Программный продукт «Списковые структуры данных «Стек, очередь»" служит для обучения и закрепления знаний по данной теме в визуально-наглядной и интерактивной среде.

Аналогов у данного программного продукта немного. В основном все материалы, доступные в интернете, предоставляют теоретические знания и визуализацию посредством статических элементов. Данный же программный продукт предоставляет интерактивную среду с наглядной визуализацией как стека и очереди, так и функций выполняемых с ними.

Программный продукт будет использоваться по мере необходимости изучения, закрепления и повторения знаний студентами. Он может быть использован как в рамках учебных занятий, так и для самостоятельного изучения и практики.

Задачи которые будут автоматизированы в этом программном продукте:

* Возможность создания стека и очереди;
* Автоматизация процесса добавления новых элементов с динамическим изменением размера стека или очереди;
* Упрощение процесса удаления элементов и соответствующая корректировка размера стека или очереди;
* Проверка на наличие элемента, выбранного пользователем, вывод элемента с «вершины» или же «головы»
* Автоматическое увеличение или уменьшение размера стека или очереди при добавлении или удалении элементов;
* Визуализация изменений, происходящие в стеке или очереди, чтобы пользователь мог наблюдать как изменяется структура в процессе работы с программой

Новая задача будет упрощать решение поставленной задачи тем что предоставит интерактивный интерфейс и функционал, который позволяет пользователю непосредственно взаимодействовать с структурами и изменениями, происходящими в них.

К входной информации можно отнести вводимые пользователем значения, такие как: размер стека или очереди, элемент который необходимо найти или добавить.

К выходной информации можно отнести стек или очередь которые будут созданы по данным пользователя, информация полученная в результате работы с программой.

Постоянной информацией в проекте будут текстовая информация, справочная информация, описание и название функций и методов.

* 1. **Инструменты разработки**

Для разработки данного проекта была выбрана среда Delphi 11, так как это очень удобная и доступная среда разработки на данный момент. На Delphi 11 возможно разрабатывать проекты различной сложности одним человеком, использующий данную среду.

Delphi – это интегрированная среда разработки (IDE) для создания программного обеспечения на языке программирования Object Pascal.

Delphi известен своей высокой производительностью и быстротой разработки. Визуальный редактор форм позволяет быстро создавать пользовательский интерфейс, а богатая библиотека компонентов предоставляет широкий спектр готовых решений для реализации различных функциональных возможностей.

Object Pascal, используемый в Delphi, сочетает в себе простоту и выразительность. Он обладает синтаксисом, который легко читается и поддерживается, что упрощает процесс разработки и обеспечивает понятность кода даже для новых членов команды. Delphi поставляется с обширным набором инструментов и библиотек, включая компоненты для работы с базами данных, сетевым взаимодействием, графикой и многим другим.

Delphi позволяет создавать различные виды программ такие как: оконные приложения, консольные приложения и приложения для работы с базами данных. Delphi предоставляет мощные средства для работы с базами данных. Среда позволяет использовать компоненты для доступа к различным СУБД, создавать сложные запросы и управлять данными в приложениях.

В Delphi сочетаются различные основные и дополнительные инструменты для упрощения разработки приложений. С помощью конструктора форм не нужно самому просчитывать где разместить элемент программы, это можно сделать с помощью визуального интерфейса. За счёт этого можно сэкономить много времени.

**Microsoft Word 2021** – это универсальное приложение для обработки текстовых документов. Оно предоставляет широкий спектр возможностей для создания, редактирования и форматирования текста. С помощью Microsoft Word легко создавать разнообразные документы, такие как письма, резюме, отчеты, статьи, презентации и многое другое. Программа обладает удобным и интуитивно понятным интерфейсом, что делает её доступной для пользователей всех уровней навыков. Благодаря возможностям автоматизации, шаблонам и интеграции с другими приложениями Microsoft Office, такими как Excel и PowerPoint, Word является незаменимым инструментом для работы с текстовой информацией в офисной среде, образовании и повседневной жизни. Microsoft Word необходим для написания документации.

**Smart Install Maker** – это программа для создания установочных пакетов для программного обеспечения под операционные системы Windows. Она обладает простым и интуитивно понятным интерфейсом, который позволяет пользователям легко создавать профессиональные установщики для своих программ.

С помощью Smart Install Maker можно создавать кастомизированные установочные пакеты, включая различные опции установки, параметры, создание ярлыков, добавление лицензионных соглашений и многое другое. Программа поддерживает автоматическое создание систем восстановления и резервного копирования, что обеспечивает дополнительную гибкость и надежность процесса установки.

**Paint.net** – это программа для редактирования изображений. Необходим для оформления некоторых элементов интерфейса. Прост в использовании и предоставляет достаточный функционал для создания необходимых проекту элементов.

**Dr.Explain** – это программа которая позволяет создавать справочные системы для использования в программе. В ней интуитивно понятный интерфейс, схожий с Microsoft Word, что позволяет быстро освоить программу и создавать справочные системы.

**Microsoft Powerpoint** 2021 – это программное обеспечение для создания и демонстрации презентаций, разработанное компанией Microsoft. PowerPoint позволяет пользователям создавать профессионально выглядящие слайды с текстом, изображениями, графикой, аудио- и видеофайлами. С помощью широкого набора инструментов форматирования и макетов легко оформить слайды в соответствии с их потребностями и предпочтениями. Программа также предоставляет функции анимации и переходов, что позволяет создавать динамичные и запоминающиеся презентации. PowerPoint интегрируется с другими приложениями Microsoft Office, что обеспечивает совместимость и возможность обмена данными между различными типами документов.

Разработка ведется на ноутбуке HP, который имеет следующие системные характеристики:

* Процессор Intel Core i5 12400H;
* Видеокарта Nvidia Geforce RTX 3050;
* SSD-накопитель на 500 гб;
* Объём ОЗУ 16 гб;
* Операционная система Windows 11 Pro.

Однако программа не очень требовательна к системным ресурсам и способна работать на любых современных компьютерах.

* 1. **Требования к приложению**

На этапе исследования предметной области был установлен целый ряд требований, которые предъявляются к разрабатываемой программе:

При моделировании форм следует учесть такой момент: программный продукт разрабатывается для студентов и начинающих программистов. По этой причине продукт должен давать возможность изучить, повторить и закрепить знания по теме.

**Требования к интерфейсу:**

Требования к интерфейсу: в связи с частым использованием программы она должна быть с приятной цветовой гаммой и понятной для пользователя. Следовательно, каждое окно должно иметь ясную визуальную иерархию своих элементов. Фрагменты текста должны располагаться на экране так, чтобы пользователя было просто и понятно принимать информацию.

Пользователь не должен испытывать какого-либо дискомфорта в плане восприятия информация, отображённой на экране. Объекты (рисунки и символы) не должны быть слишком мелкие. Все окна приложения по возможности должны помещаться на экран полностью, так как использование в процессе работы полос прокруток достаточно неудобно.

**Требования к надежности:**

специальных требований к надежности не предъявляется. Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбое операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Отказы программы вследствие некорректных действий пользователя при взаимодействии с программой через графический интерфейс не должны влиять на конечный результат.

.

1. **Проектирование задачи**
   1. **Организация данных**

Проектирование задачи – это очень важный и ответственный этап в разработке любого приложения.

Он важен по причине того, что методы управления формами пользователем основываются на высокой степени специализации каждого из компонентов.

Необходимым условием при разработке данного приложения является описание организации данных, т.е. логическая и физическая структура данных в контексте среды разработки. В разрабатываемой программе будут использоваться три вида данных.

Первым видом являются данные, которые будут введены разработчиком на этапе реализации задачи. Сюда можно отнести изображения (иконки), описания стека, очереди, списков, исходные коды методов работы с стеком и очередями, вопросы теста,

Вторым видом данных, используемых в программе, является вводимая пользователем информация. Входной информацией в разрабатываемой программе будут являться данные, которые введет пользователь. Это может количество элементов стека или очереди, элементы, которые необходимо добавить, удалить или найти. Но необязательно заполнять элементы вручную, в программе предусмотрена генерация элементов самой программой.

Третьим видом данных является результат программы – это могут быть данные полученные в результате проведения различных операций над стеком и очередями, результаты теста. Его также относят к отдельному виду, так как ни пользователь, ни разработчик его не вводят, а программа сама получает его в результате выполнения определенных действий.

Таким образом, организация данных является важной задачей при разработке данной и любой программы.

* 1. **Процессы**

Согласно всем перечисленным требованиям и указаниям, которые были рассмотрены в разделе «Анализ задачи», было определено, чем конкретно должна заниматься разрабатываемая курсовая программа. Главной ее задачей будет являться: демонстрация работы стека и очереди, функций работы с ними . Весь данный процесс будет заключаться в следующем.

Для реализации задач будут использоваться процедуры.

На главном окне базовые сведения в формате .htm и возможность визуализации будет генерироваться на основе выбранного пункта в компоненте TRadioGroup.

В окне с визуализацией выбранной структуры возможен переход ко всем доступным формам проекта. После ознакомления с функциями стека и очереди, и опционального ознакомления со справочником, можно пройти тест для проверки своих знаний и получить оценку.

* 1. **Описание внешнего пользовательского интерфейса**

Важным аспектом выполнения курсового проекта является налаживание диалога между пользователем и программой. Это во многом зависит от того, как программист спроектирует программу, какие компоненты будут включены и какие процессы будут автоматизированы. Во-первых, особое внимание следует уделить интерфейсу. Разработчик должен оформить программу таким образом, чтобы пользователь сразу понял, что от него требуется.

Для обеспечения эффективной работы пользователя нужно создать целостное приложение, охватывающее конкретную предметную область, где все компоненты будут сгруппированы по их функциональному назначению. При этом необходимо обеспечить удобный графический интерфейс. Приложение должно позволять пользователю решать задачи с минимальными усилиями по сравнению с работой с разрозненными объектами.

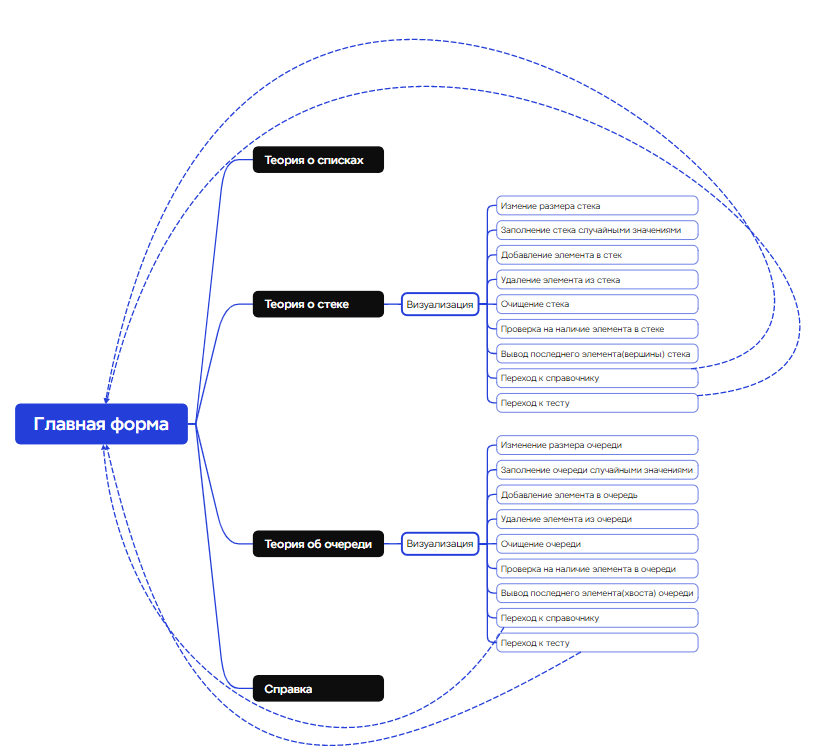
Ниже представлена система меню и организация навигации между окнами программы (Рисунок 1):

Рисунок 1 – Навигация между окнами программы

**Прототип** – это наглядная модель пользовательского интерфейса. В сущности это «черновик», созданный на основе представления разработчика о потребностях пользователя. Итоговое отображение программы может отличаться от прототипа. Ниже можно наблюдать прототипы:

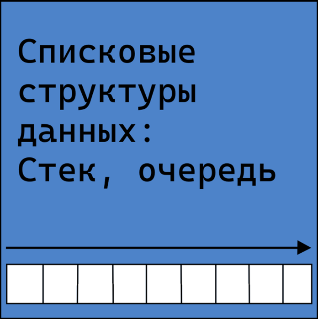
1. Логотипа (Рисунок 2);
2. Заставки (Рисунок 3);
3. Главного окна (Рисунок 4);
4. Окна визуализации (Рисунок 5);
5. Окна теста (Рисунок 6);
6. Окна-справочника, содержащего теоретическую информацию (Рисунок 7);

Рисунок 2 – Прототип логотипа

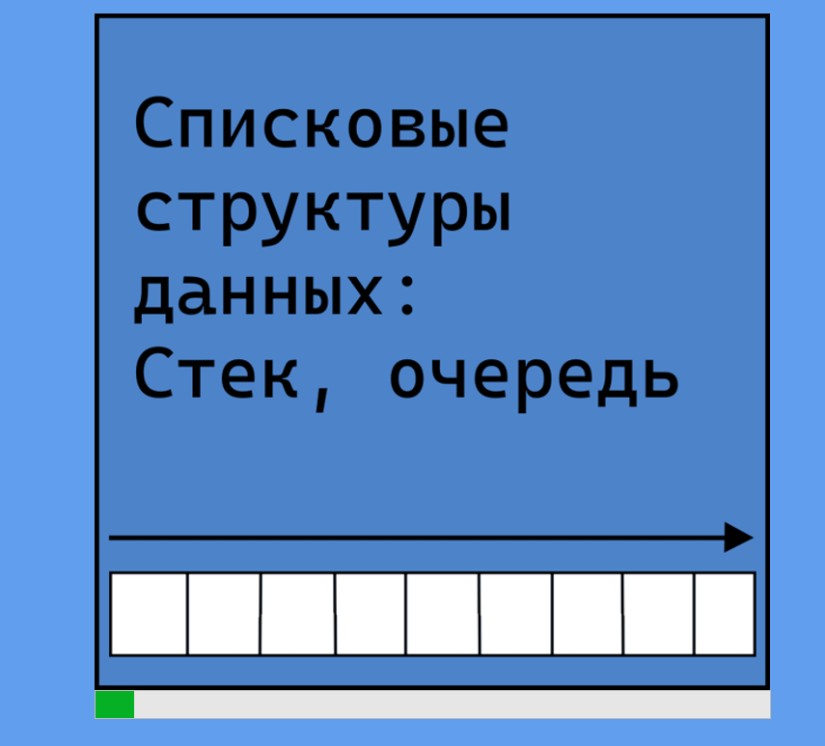
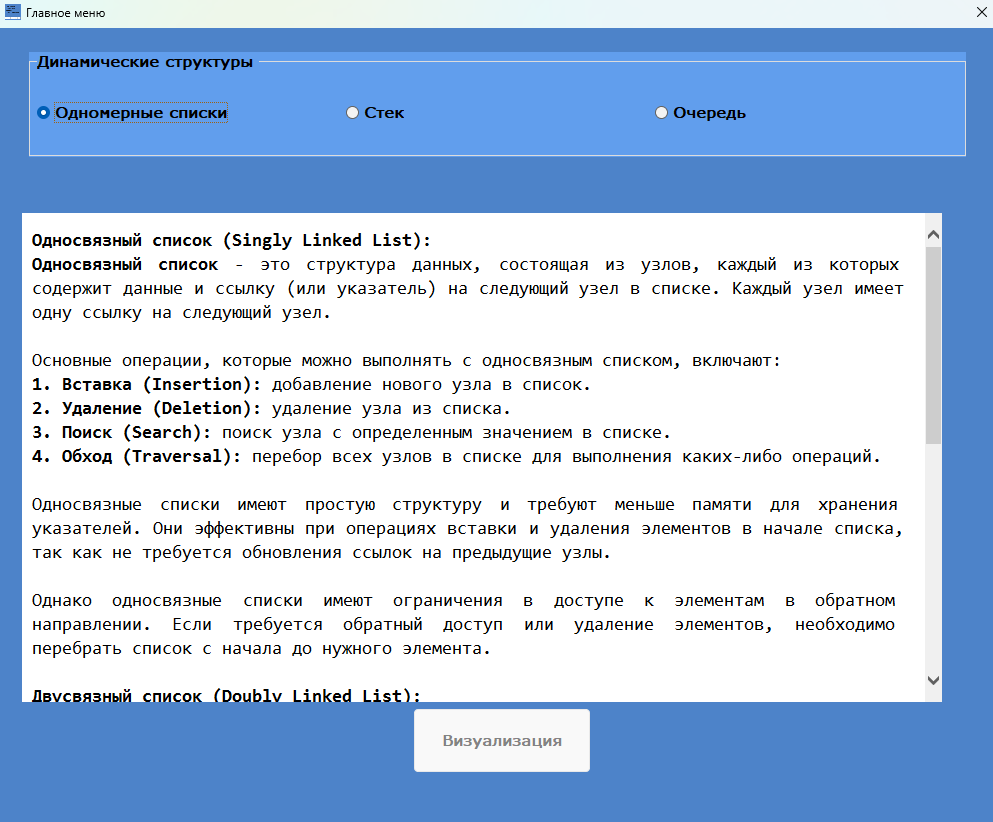


Рисунок 3 – Прототип заставки

Рисунок 4 – Прототип главного окна

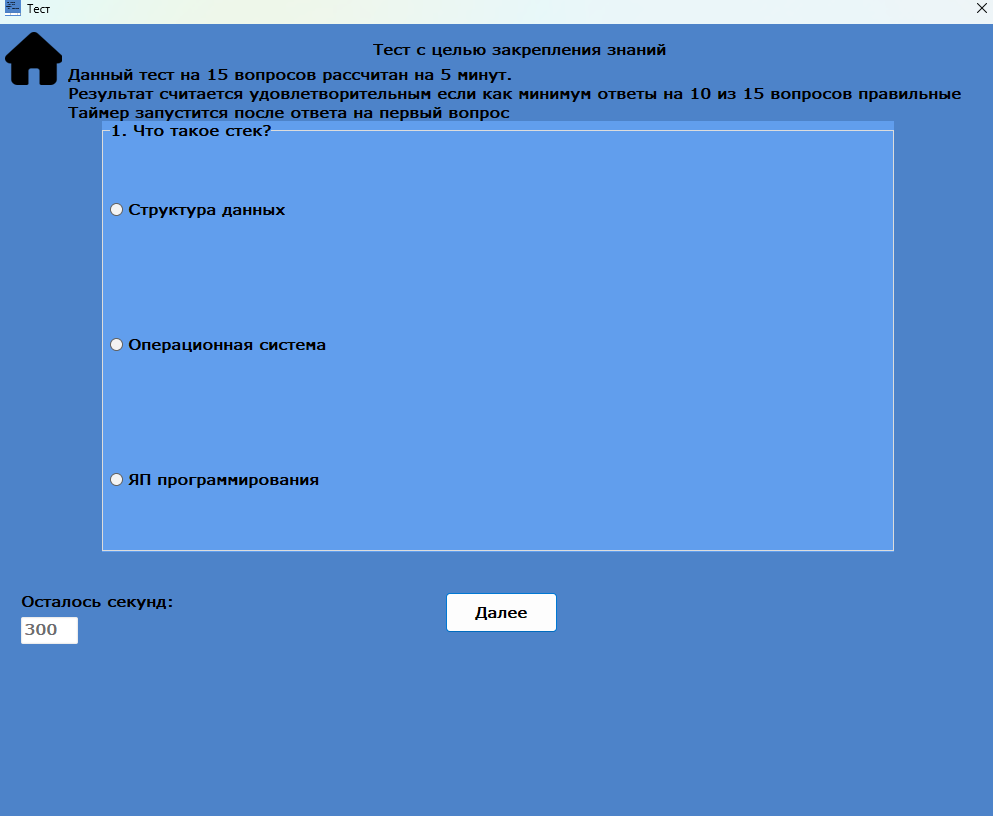
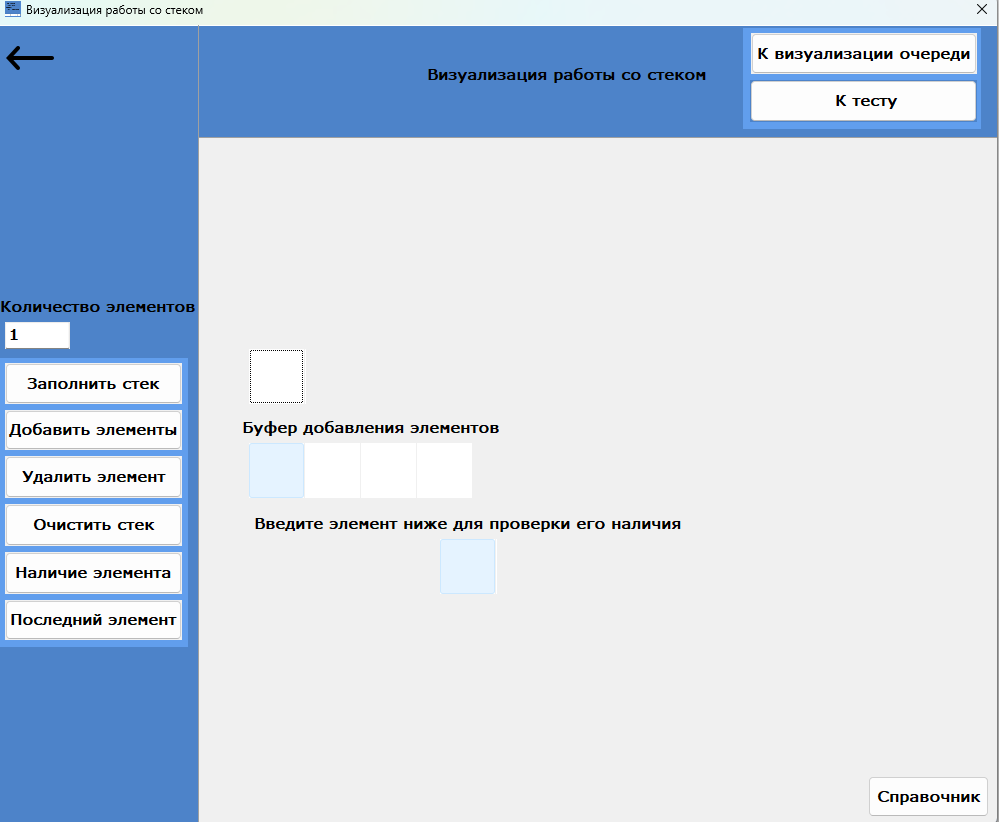
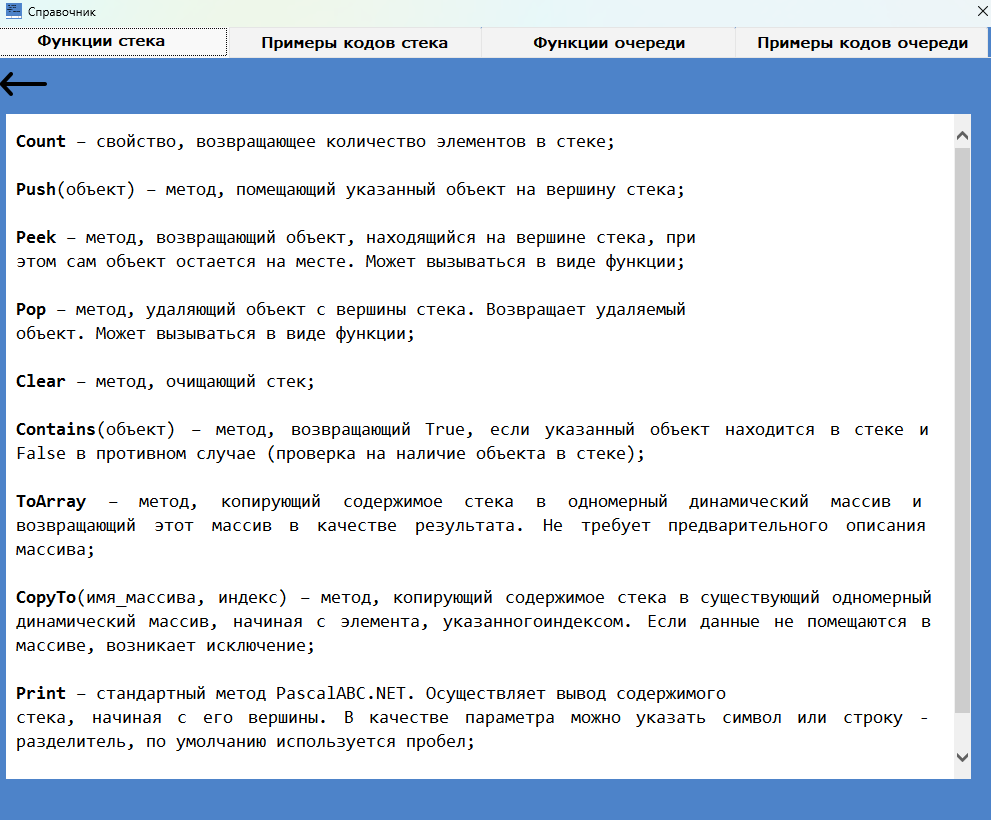
Рисунок 5 – Прототип окна визуализации

Рисунок 6 – Прототип окна теста

Рисунок 7 – Прототип окна-справочника, содержащего теоретические данные

**3.1 Реализация**

**3.1.1 Структура программы**

Данный курсовой проект содержит 6 модулей. Далее рассмотрим назначение каждого модуля:

Unit1 – является главной формой проекта

Unit2 – является формой заставкой проекта

Unit3 – является формой визуализацией стека

Unit4 – является формой визуализацией очереди

Unit5 – является формой со справкой

Unit7 – является формой с тестом

**3.1.1 Структура и описание процедур и функций пользователя**

Описание разработанных процедур находятся в таблице 1.

Таблица 1 – Процедуры и функции

|  |  |  |  |
| --- | --- | --- | --- |
| Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 1 | 2 | 3 | 4 |
| procedure TForm1.Button1Click(Sender: TObject); | Unit1 | Button1 | Кнопка выхода из программы |
| procedure TForm1.Button4Click(Sender: TObject); | Unit1 | Button4 | Кнопка перехода к выбранной форме |
| procedure TForm1.FormShow(Sender: TObject); | Unit1 | Main | Изменение цветовой гаммы компонентов и показ формы-заставки |
| procedure TForm1.N3Click(Sender: TObject); | Unit1 | MainMenu | Открытие справки |
| procedure TForm1.RadioGroup1Click(Sender: TObject); | Unit1 | RadioGroup | Выбор для вывода соответствующей информации и выбор перехода |
| procedure TForm2.FormActivate(Sender: TObject); | Unit2 | Main | Изменение цветовой гаммы |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| procedure TForm2.Timer1Timer(Sender: TObject); | Unit2 | Timer | Отвечает за изменение позиции progressbar, после её заполнения закрывает форму |
| procedure TForm3.Button1Click(Sender: TObject); | Unit3 | Button1 | Заполняет сетку случайными значениями |
| procedure TForm3.Button2Click(Sender: TObject); | Unit3 | Button2 | Удаляет элемент из буфера, перемещает его в основную сетку, сдвигает значения в буфере |
| procedure TForm3.Button3Click(Sender: TObject); | Unit3 | Button3 | Удаляет элемент из основной сетки |
| procedure TForm3.Button4Click(Sender: TObject); | Unit3 | Button4 | Очистка элементов сетки |
| procedure TForm3.Button5Click(Sender: TObject); | Unit3 | Button5 | Переход на форму с визуализацией очереди |
| procedure TForm3.Button6Click(Sender: TObject); | Unit3 | Button6 | Переход на форму со справочником |
| procedure TForm3.Button7Click(Sender: TObject); | Unit3 | Button7 | Переход на форму с тестом |
| procedure TForm3.Button8Click(Sender: TObject); | Unit3 | Button8 | Проверка на наличие элемента, введенного пользователем, в сетке |
| procedure TForm3.Button9Click(Sender: TObject); | Unit3 | Button9 | Вывод последнего элемента |
| procedure TForm3.Edit1Change(Sender: TObject); | Unit3 | Edit1 | Изменение размера основной сетки |
| procedure TForm3.FormActivate(Sender: TObject); | Unit3 | Main | Изменение цветовой гаммы |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| procedure TForm3.Image1Click(Sender: TObject); | Unit3 | Image1 | Переход на главную форму |
| procedure TForm3.StringGrid1SetEditText(Sender: TObject; ACol, ARow: Integer; const Value: string); | Unit3 | Stringgrid1 | Обозначение «Вершины» при изменении сетки |
| procedure TForm3.Timer1Timer(Sender: TObject); | Unit3 | Timer1 | Служит для правильной логики работы вывода последнего и поиска |
| procedure TForm4.Button1Click(Sender: TObject); | Unit4 | Button1 | Заполняет сетку случайными значениями |
| procedure TForm4.Button2Click(Sender: TObject); | Unit4 | Button2 | Удаляет элемент из буфера, перемещает его в основную сетку, сдвигает значения в буфере |
| procedure TForm4.Button3Click(Sender: TObject); | Unit4 | Button3 | Удаляет элемент из основной сетки |
| procedure TForm4.Button4Click(Sender: TObject); | Unit4 | Button4 | Очистка элементов сетки |
| procedure TForm4.Button5Click(Sender: TObject); | Unit4 | Button5 | Переход на форму с визуализацией очереди |
| procedure TForm4.Button5Click(Sender: TObject); | Unit4 | Button5 | Переход на форму с визуализацией очереди |
| procedure TForm4.Button6Click(Sender: TObject); | Unit4 | Button6 | Переход на форму со справочником |
| procedure TForm4.Button7Click(Sender: TObject); | Unit4 | Button7 | Переход на форму с тестом |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| procedure TForm4.Button8Click(Sender: TObject); | Unit4 | Button8 | Проверка на наличие элемента, введенного пользователем, в сетке |
| procedure TForm4.Button9Click(Sender: TObject); | Unit4 | Button9 | Вывод последнего элемента |
| procedure TForm4.Edit1Change(Sender: TObject); | Unit4 | Edit1 | Изменение размера основной сетки |
| procedure TForm4.FormActivate(Sender: TObject); | Unit4 | Main | Изменение цветовой гаммы |
| procedure TForm4.Image1Click(Sender: TObject); | Unit4 | Image1 | Переход на главную форму |
| procedure TForm4.StringGrid1SetEditText(Sender: TObject; ACol, ARow: Integer; const Value: string); | Unit4 | Stringgrid1 | Обозначение «Головы» и «Хвоста» при изменении сетки |
| procedure TForm4.Timer1Timer(Sender: TObject); | Unit4 | Timer1 | Служит для правильной логики работы вывода последнего и поиска |
| procedure TForm5.FormShow(Sender: TObject); | Unit5 | Main | Изменение цветовой гаммы, отображение информации |
| procedure TForm5.Image5Click(Sender: TObject); | Unit5 | Image5 | Переход на главную форму проекта |
| procedure TForm5.TabControl1Change(Sender: TObject); | Unit5 | TabControl1 | Служит для выбора опции с последующим выводом информаци в зависимости от выбора |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| procedure TForm7.FormCreate(Sender: TObject); | Unit7 | Main | Изменение цветовой гаммы, считывание вопросов из файла |
| procedure TForm7.Image1Click(Sender: TObject); | Unit7 | Image1 | Переход на главную форму |
| procedure TForm7.Timer1Timer(Sender: TObject); | Unit7 | Timer1 | Таймер который ведет отсчет |
| procedure TForm7.Timer2Timer(Sender: TObject); | Unit7 | Timer2 | Таймер который ограничивает время на тест и выводит результат |
| procedure TForm7.Button1Click(Sender: TObject); | Unit7 | Button1 | Засчитывает баллы при правильном ответе, очищает вопросы, загружает новые из файла, заканчивает тест |

**3.1.2 Описание использованных компонентов**

Описание использованных для разработки приложения компонентов приводится в таблице 2

Таблица 2 – использованные компоненты

|  |  |  |
| --- | --- | --- |
| Компоненты | На какой форме расположен | Назначение |
| 1 | 2 | 3 |
| TMainMenu | Form1 | Служит как меню для вызова справки |
| TButton | Form1, Form3, Form4, Form5, Form7 | Служит для вызова каких-либо процедур |
| TRadioGroup | Form1, Form5 | Служит для выбора опции и как тест |
| TProgressBar | Form2 | Служит как полоса загрузки на форме заставке |

Продолжение таблицы 2

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| TEdit | Form3, Form4, Form7 | Служит для ввода данных пользователем, ещё и как таймер |
| TStringGrid | Form3, Form4 | Служит для визуализации стека или очереди, как окно для ввода данных |
| TPanel | Form3, Form4 | Служат для визуальной составляющей проекта |
| TLabel | Form1, Form3, Form4, Form7 | Служит для вывода какой-либо информации на экран |
| TTimer | Form3, Form4, Form7 | Служит как соответственно таймер, для выполнение логики некоторых процедур |
| TWebBrowser | Form5 | Служит для подгрузки и вывода информации из файла |
| TImage | Form2, Form3, Form4, Form5, Form6 | Служат как кнопки, а так же как иконка заставки на соответствующей форме |
| TTabControl | Form5 | Служит для выбора опции при выводе теоретических сведений |

3.2 Спецификация программы

Точное название проекта и его состав приводится в таблице 3.2

Таблица 3 – Спецификация программы

|  |  |
| --- | --- |
| Имя файла | Назначение |
| 1 | 2 |
| Динамические структуры данных стек, очередь.exe | Исполняемый файл проекта, используется для запуска программы на выполнение |
| Динамические структуры данных стек, очередь.dproj | Файл проекта, связывает все файлы из которых состоит приложение. |
| Unit1.pas | Файл программного модуля главной формы |
| Unit2.pas | Файл программного модуля формы заставки |
| Unit3.pas | Файл программного модуля формы визуализации стека |
| Unit4.pas | Файл программного модуля формы визуализации очереди |
| Unit5.pas | Файл программного модуля формы справочника |
| Unit7.pas | Файл программного модуля формы теста |

Продолжение таблицы 3

|  |  |
| --- | --- |
| 1 | 2 |
| Unit1.dfm | Главная форма |
| Unit2.dfm | Заставка |
| Unit3.dfm | Форма визуализации стека |
| Unit4.dfm | Форма визуализации очереди |
| Unit5.dfm | Форма справочника |
| Unit7.dfm | Форма теста |
| Test.txt | Текстовый файл с вопросами и ответами для теста |
| Setup.exe | Файл-установщик проекта |
| Стек.htm | Файл с теорией |
| Списки.htm | Файл с теорией |
| Очередь.htm | Файл с теорией |
| Функции\_стека.htm | Файл с теорией |
| Функции\_очереди.htm | Файл с теорией |
| Примеры\_очереди.htm | Файл с теорией |
| Примеры\_стека.htm | Файл с теорией |
| Lists\_help.chm | Файл со справкой |

**4. Тестирование**

При разработке данной программы многие возникающие ошибки и недоработки были исправлены на этапе реализации проекта. После завершения этапа написания программы было проведено тщательное функциональное тестирование. Функциональное тестирование должно гарантировать работу всех элементов программы в автономным режиме.

Отчет о результатах тестирования представлен в таблице 4

Таблица 4 – Отчет о результатах тестирования

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Идентификатор теста | Тест | Ожидаемый результат | Физический результат | | Результат тестирования |
| 1 | 2 | 3 | 4 | | 5 |
| 1 | Проверка входа пользователя в приложение | Открытие заставки а после главного окна | | Открытие заставки а после главного окна | Выполнено |
| 2 | Проверка кнопок с выбором опций | Открытие соответствующего теоретического материала | | Открытие соответствующего теоретического материала | Выполнено |
| 3 | Проверка кнопки «Визуализация» | Переход на соответствующую опции форму | | Переход на соответствующую опции форму | Выполнено |
| 4 | Проверка элемента edit | Изменение размера основного сетки | | Изменение размера основного сетки | Выполнено |
| 5 | Проверка кнопки «Заполнить стек» | Заполнение сетки случайными значениями | | Заполнение сетки случайными значениями | Выполнено |
| 6 | Проверка кнопки «Добавить элемент» | Добавление элемента из буфера, увеличение сетки, сдвиг чисел в буфере | | Добавление элемента из буфера, увеличение сетки, сдвиг чисел в буфере | Выполнено |
| 7 | Проверка кнопки «Удалить элемент» | Удаление элемента в сетке, уменьшение её размера | | Удаление элемента в сетке, уменьшение её размера | Выполнено |
| 8 | Проверка кнопки «Очистить стек» | Очищение значений в сетке | | Очищение значений в сетке | Выполнено |
| 9 | Проверка кнопки «Наличие элемента» | Проверка на наличие элемента, введенного в специальной сетке на наличие в основной | | Проверка на наличие элемента, введенного в специальной сетке на наличие в основной | Выполнено |
| 10 | Проверка кнопки «Последний элемент» | Вывод последнего добавленного элемента в сетке | | Вывод последнего добавленного элемента в сетке | Выполнено |

Таблица 4 – продолжение

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 11 | Проверка кнопки «Справочник» | Переход на форму со справочником | Переход на форму со справочником | Выполнено |
| 12 | Проверка кнопки «К визуализации очереди» | Переход на форму с визуализацией очереди | Переход на форму с визуализацией очереди | Выполнено |
| 13 | Проверка кнопки «К тесту» | Переход на форму с тестом | Переход на форму с тестом | Выполнено |
| 14 | Проверка элемента edit | Изменение размера | Изменение размера | Выполнено |
| 15 | Проверка кнопки «Заполнить очередь» | Заполнение сетки случайными значениями | Заполнение сетки случайными значениями | Выполнено |
| 16 | Проверка кнопки «Добавить элемент» | Добавление элемента из буфера, увеличение основной сетки, сдвиг чисел в буфере | Добавление элемента из буфера, увеличение основной сетки, сдвиг чисел в буфере | Выполнено |
| 17 | Проверка кнопки «Удалить элемент» | Удаление элемента из сетки, изменение размера основной сетки | Удаление элемента из сетки, изменение размера основной сетки | Выполнено |
| 18 | Проверка кнопки «Очистить очередь | Очистка элементов в сетке | Очистка элементов в сетке | Выполнено |
| 19 | Проверка кнопки «Наличие элемента» | Проверка на наличие элемента, введенного в специальной сетке на наличие в основной | Проверка на наличие элемента, введенного в специальной сетке на наличие в основной | Выполнено |
| 20 | Проверка кнопки «Последний элемент» | Вывод последнего добавленного элемента в сетке | Вывод последнего добавленного элемента в сетке | Выполнено |
| 21 | Проверка кнопки «К визуализации стека» | Переход на форму с визуализацией стека | Переход на форму с визуализацией стека | Выполнено |
| 22 | Проверка кнопки «Справочник» | Переход на форму с справочником | Переход на форму с справочником | Выполнено |
| 23 | Проверка переключения вкладок на форме-справке | Подгрузка и вывод соответствующей информацией из файлов | Подгрузка и вывод соответствующей информацией из файлов | Выполнено |
| 24 | Проверка кнопки «Далее» на форме с тестом | Подгрузка опросов теста, расчет количества и процент правильных ответов, окончание теста | Подгрузка опросов теста, расчет количества и процент правильных ответов, окончание теста | Выполнено |
| 25 | Проверка срабатывания таймера на форме с тестом | Окончание теста, вывод результатов | Окончание теста, вывод результатов | Выполнено |

При разработке программного продукта было решено множество проблем, например, не осуществлялся переход к следующему изображению по нажатию стрелки или открывался доступ к последнему квесту, когда был пройден только первый, поэтому в будущем пользователь не столкнется с данными проблемами.

Элементы программы были проверены, и было установлено, что все они работают правильно и выполняют задачи, указанные в процедурах.

**5. Применение**

**5.1 Общие сведения о программном продукте**

Цель данного проекта заключается в изучении, повторении и закреплений знаний по теме «Стек, очередь»

Создаваемое приложение будет рассчитано на студентов и начинающих программистов.

Быстродействие любой программы во многом зависит от характеристик выбранного персонального компьютера: рабочей частоты процессора, объема оперативной памяти и т.д. Несмотря на все реализованные в ней задачи, она легко запускается и функционирует на любых машинах.

Тестирование проводилось на разных классах ЭВМ и работать с данной программой было комфортно.

**5.2 Инсталляция**

Для того, чтобы установить программу необходимо запустить файл Setup.exe. Появится окно установки приложения «Динамические структуры стек, очередь».

Затем достаточно следовать приведенной инструкции установки приложения.

**5.3 Выполнение программы**

**5.3.1 Запуск программы**

Данную программу можно запустить различными способами. Первым из них является запуск с помощью ярлыка на рабочем столе. Необходимо дважды щелкнуть левой кнопки мыши на ярлыке с названием «Динамические структуры: стек, очередь»

Вторым способом является запуск из каталога, в который устанавливалось приложение.

По подготовленным тестам будет осуществляться функциональное и полное тестирование программного продукта. Отчет о результатах тестирования будет представлен в 4 разделе пояснительной записки.

**5.3.2 Инструкция по работе с программой**

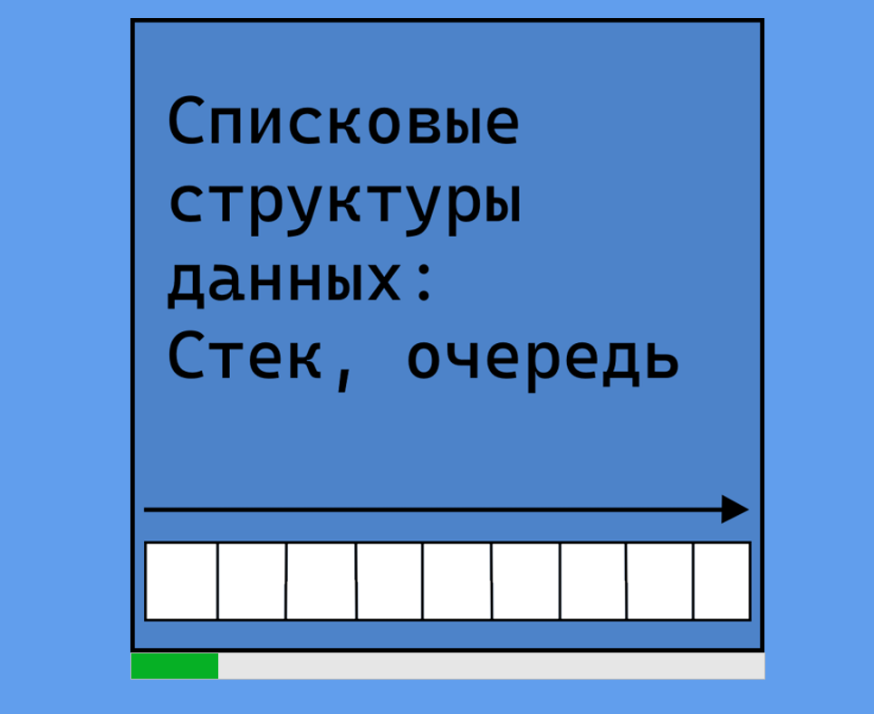
После запуска приложения на экране нас встречает заставка (Рисунок 1).

Рисунок 1 – заставка

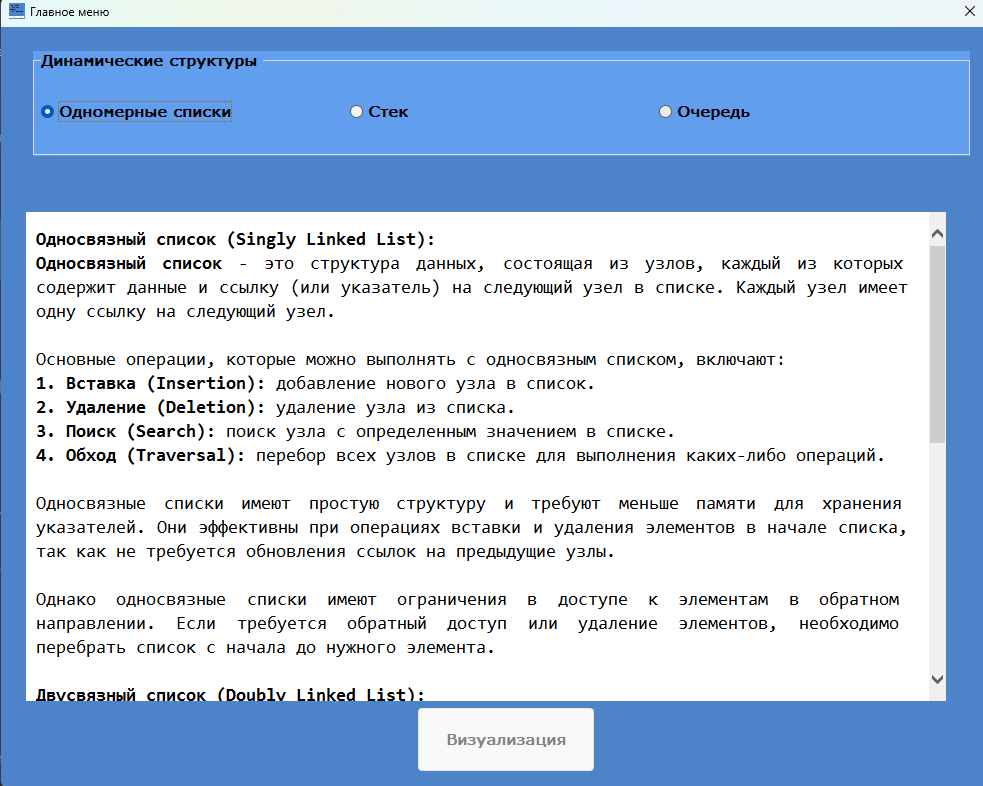
После загрузки данная форма закроется и перед пользователем появится главная форма проекта (Рисунок 2). На данной форме есть несколько элементов. По выбору одной из опций в верхней части окна будет выведена краткие теоретические сведения по выбранной теме. При выборе стека или очереди будет доступна кнопка визуализации.

Рисунок 2 ­– главная форма проекта

После выбора нужной опции и перехода к форме с визуализацией соответствующей структуры будет выведено или окно визуализации стека (Рисунок 3), или же окно визуализации очереди (Рисунок 4). Данные окна похожи по своему функционалу.

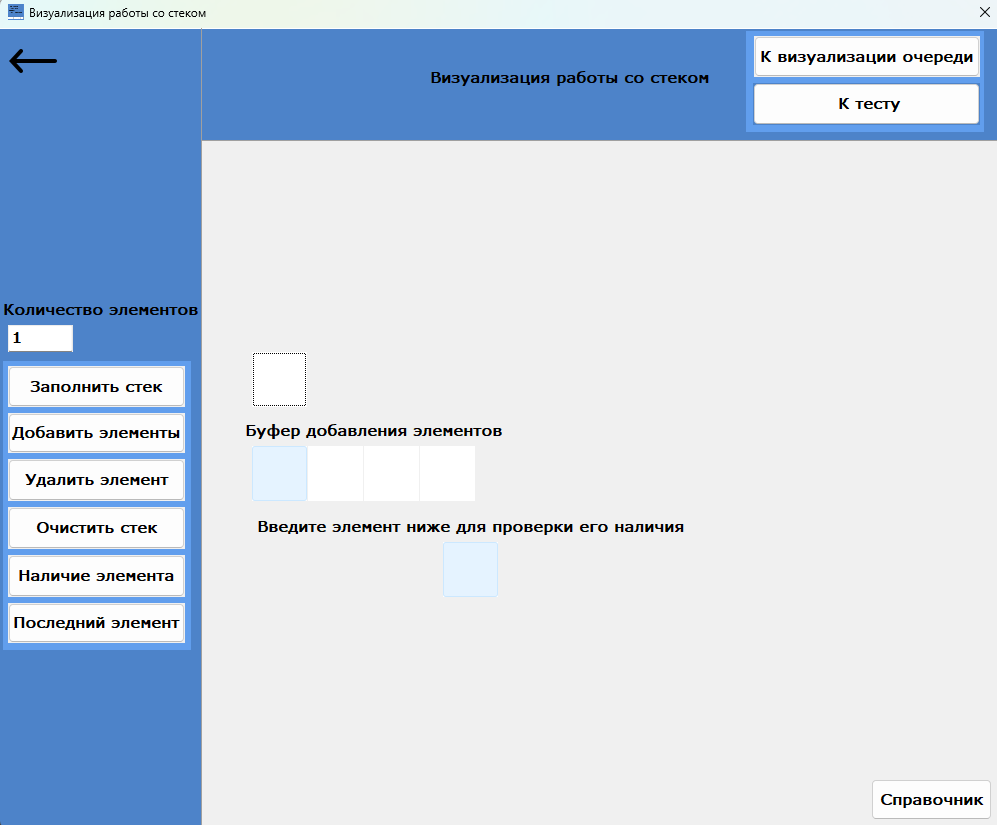
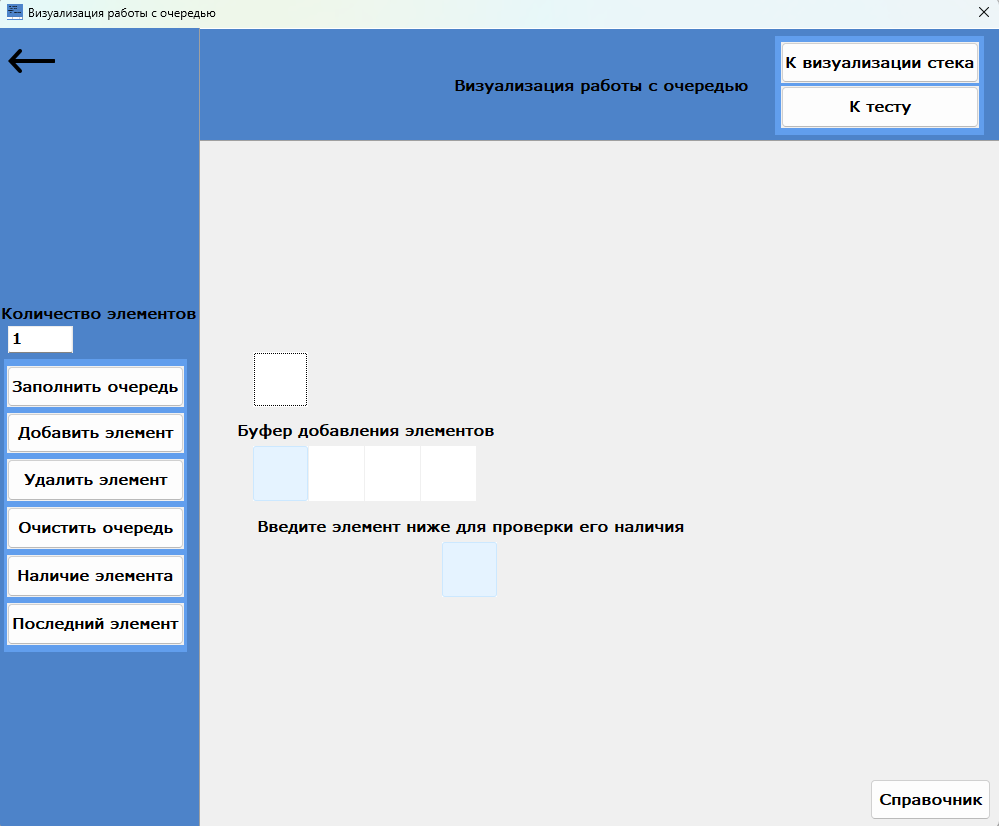
Окно содержит множество кнопок. Разберем каждую по-порядку. Поле ввода данных с подписью количество элементов – отвечает за количество клеток в основной сетке. Кнопка «Добавить элемент» - добавляет самый левый элемент из буфера в основную сетку, при необходимости увеличивает её размеры. Кнопка «Удалить элемент» удаляет элемент из сетки, изменяя её размер. Кнопка «Очистить стек/очередь» очищает все клетки основной сетки. Кнопка «Наличие элемента» проверяет есть ли элемент, введенный в специальное окно, в основной сетке. Кнопка «Последний элемент» выводит «вершину» если мы работаем со стеком или же «хвост» если работаем с очередью. Кнопка «К визуализации очереди/стека» перемещает на соответствующую форму. Кнопка «К тесту» переносит к форме с тестом для проверки своих знаний. Кнопка «Справочник» переносит к форме со справочником для ознакомления с теоретическими сведениями о функциях работы со стеком или же очередью.

Рисунок 3 – форма визуализации стека

Рисунок 4 – форма визуализации очереди

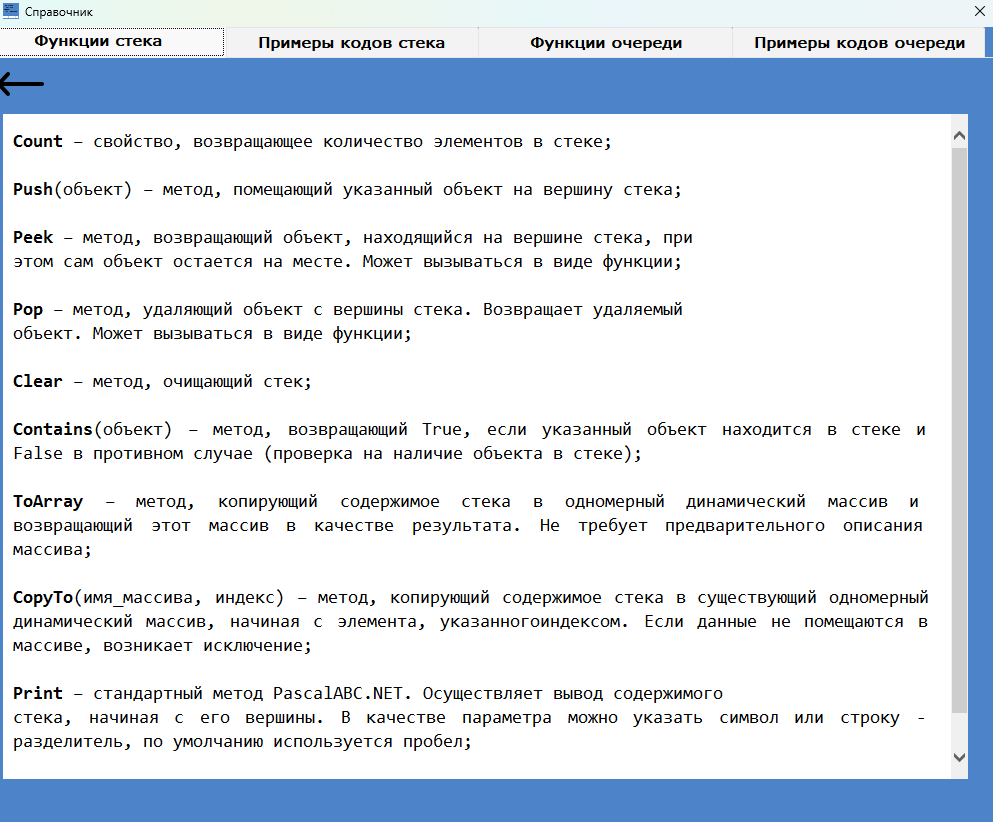
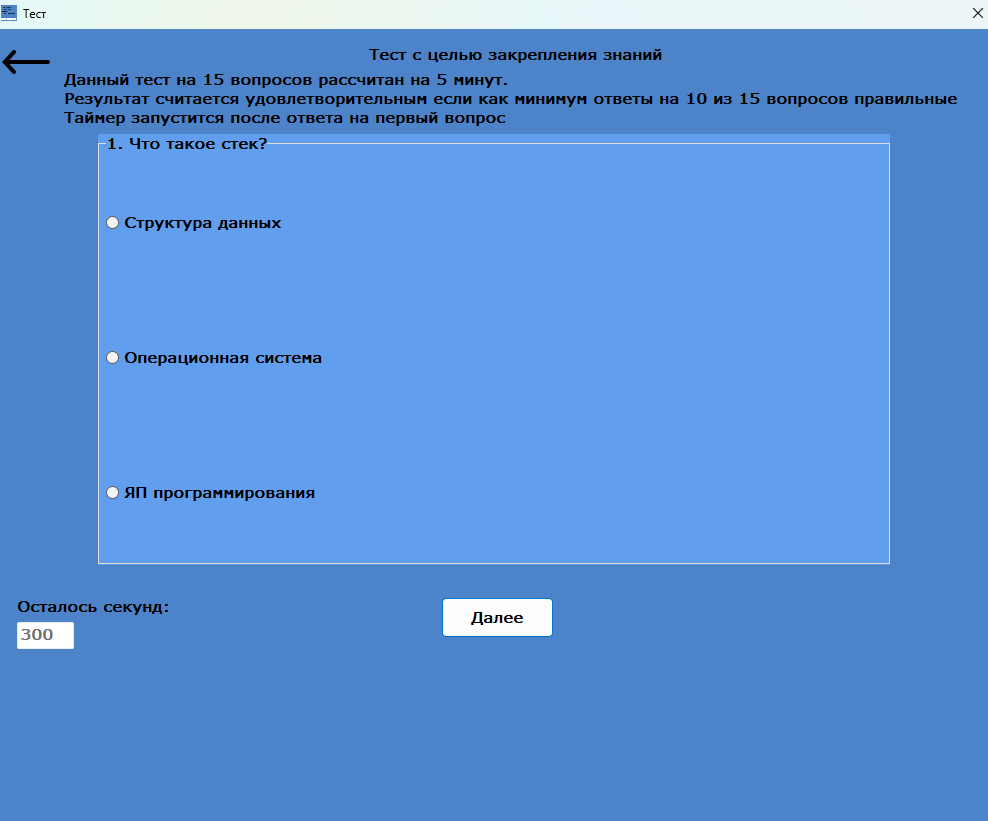
По нажатию кнопка «Справка» мы окажемся в специальной форме (Рисунок 5). В ней, выбрав нужную опцию, можно ознакомиться с теоретическими сведениями и примерами кодов по стеку или же очереди. Отсюда же возможен переход обратно на главную форму.

Рисунок 5 – форма справочника

При нажатии кнопки «К тесту» откроется форма с тестом для проверки знаний (Рисунок 6). По выбору варианта и нажатии кнопки далее таймер запустится, а ответ будет записан. После прохождения теста вам будет выведен результат (Рисунок 7).

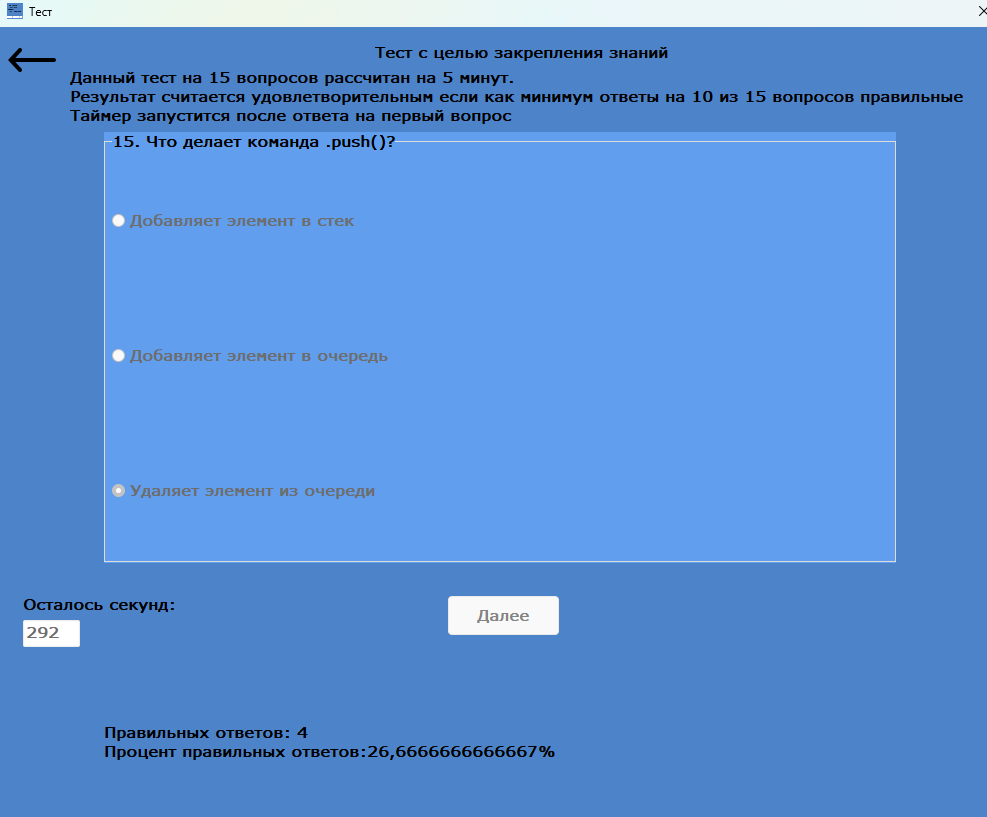
Рисунок 6 – форма теста

Рисунок 7 – форма с законченным тестом и результатом

**Заключение**

Целью данного проекта заключалось изучение, закрепление и повторение знаний по теме «Стек, очередь»

В поставленной задаче был реализован простой и понятный пользовательский интерфейс.

В ходе тестирования все исключительные ситуации были обработаны. Проект работает без сбоев и ошибок.

В процессе разработки программного продукта я научился создавать динамические компоненты на форме, проработал их взаимодействие, закрепил умение создания собственных процедур, научился обрабатывать все исключительные ситуации.

Исходя из этого, можно сделать вывод, что программа реализована успешно.

**Список использованных источников**

1. Осипов Александр Викторович PascalABC.NET: Введение в современное программирование. – Ростов-на-Дону, 2019 – 572с.
2. .Симонович С.В., Евсеев Г.А., Алексеев А.Г. Специальная информатика: Учебное пособие. - М.: АСТ-ПРЕСС: Инфорком-Пресс, 2001. - 480 с.
3. Архангельский, А. Я. Delphi 7. Справочное пособие. - Москва: Бином-Пресс, 2014. - 1024 с.

**Приложение А**

Листинг программы

nit Unit1;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.ExtCtrls,

Vcl.OleCtrls, SHDocVw, Vcl.Menus, ShellAPI;

type

TForm1 = class(TForm)

Label1: TLabel;

Button4: TButton;

RadioGroup1: TRadioGroup;

WebBrowser1: TWebBrowser;

Label2: TLabel;

Label3: TLabel;

Button1: TButton;

MainMenu1: TMainMenu;

N2: TMenuItem;

N3: TMenuItem;

procedure Button4Click(Sender: TObject);

procedure RadioGroup1Click(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure N3Click(Sender: TObject);

private

{ Private declarations }

public

f,v,c:text;

s:string;

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

uses Unit2, Unit3, unit4;

procedure TForm1.Button1Click(Sender: TObject);

begin

form1.Close;

end;

procedure TForm1.Button4Click(Sender: TObject);

begin

case radiogroup1.itemindex of

1:

begin

form3.Show;

form1.closemodal;

end;

2:

begin

form4.Show;

form1.closemodal;

end;

end;

end;

procedure TForm1.FormShow(Sender: TObject);

begin

form1.Color:=rgb(77,131,201);

radiogroup1.color:=rgb(97, 158, 237);

form2.Show;

end;

procedure TForm1.N3Click(Sender: TObject);

begin

ShellExecute(0,Pchar('Open'),Pchar(ExtractFilePath(Application.ExeName)+'lists\_help.chm'),nil,nil,SW\_SHOW);

end;

procedure TForm1.RadioGroup1Click(Sender: TObject);

begin

case radiogroup1.itemindex of

0:

begin

label2.Visible:=false;

label3.Visible:=false;

webbrowser1.Visible:=true;

button4.Enabled := false;

webbrowser1.Navigate(ExtractFilePath(Application.ExeName)+'список.htm');

end;

1:

begin

label2.Visible:=false;

label3.Visible:=false;

webbrowser1.Visible:=true;

button4.Enabled := true;

webbrowser1.Navigate(ExtractFilePath(Application.ExeName)+'стек.htm');

end;

2:

begin

label2.Visible:=false;

label3.Visible:=false;

webbrowser1.Visible:=true;

button4.Enabled := true;

webbrowser1.Navigate(ExtractFilePath(Application.ExeName)+'очередь.htm');

end;

end;

end;

end.

unit Unit2;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.ExtCtrls, Vcl.ComCtrls,

Vcl.Imaging.pngimage;

type

TForm2 = class(TForm)

ProgressBar1: TProgressBar;

Timer1: TTimer;

Label1: TLabel;

Image1: TImage;

procedure Timer1Timer(Sender: TObject);

procedure FormActivate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

uses unit1;

{$R \*.dfm}

procedure TForm2.FormActivate(Sender: TObject);

begin

form2.Color:=rgb(97, 158, 237);

end;

procedure TForm2.Timer1Timer(Sender: TObject);

begin

if progressbar1.position<progressbar1.max then

progressbar1.Position:=progressbar1.Position+10

else

begin

form1.show;

close;

timer1.Enabled:=false;

end;

end;

end.

unit Unit3;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Grids,

Vcl.Imaging.pngimage, Vcl.ExtCtrls, Vcl.Buttons;

type

TForm3 = class(TForm)

StringGrid1: TStringGrid;

Edit1: TEdit;

Button1: TButton;

Button2: TButton;

label3: TLabel;

StringGrid2: TStringGrid;

Button3: TButton;

Button4: TButton;

Button5: TButton;

Button6: TButton;

label4: TLabel;

Panel1: TPanel;

Panel2: TPanel;

Label2: TLabel;

Image1: TImage;

Button7: TButton;

Label1: TLabel;

Button8: TButton;

StringGrid3: TStringGrid;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

Timer1: TTimer;

Panel3: TPanel;

Panel4: TPanel;

Button9: TButton;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Edit1Change(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Button6Click(Sender: TObject);

procedure StringGrid1SetEditText(Sender: TObject; ACol, ARow: Integer;

const Value: string);

procedure Button7Click(Sender: TObject);

procedure FormActivate(Sender: TObject);

procedure Button8Click(Sender: TObject);

procedure Button9Click(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form3: TForm3;

implementation

{$R \*.dfm} uses math, Unit4,unit1, Unit5, Unit6,unit7;

procedure TForm3.Button1Click(Sender: TObject);

var

n,i:integer;

begin

for i:=0 to stringgrid1.ColCount-1 do

begin

randomize;

stringgrid1.Cells[i,0]:=inttostr(randomrange(-10,10));

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end;

end;

procedure TForm3.Button2Click(Sender: TObject);

var i, v, c: Integer;

begin

for i := StringGrid1.ColCount - 1 downto 0 do

begin

if (StringGrid1.Cells[i, 0] = '') and (stringgrid2.Cells[0,0] <> '') then

begin

StringGrid2.Canvas.Brush.Color := clRed;

StringGrid1.Canvas.Brush.Color := clGreen;

StringGrid2.Canvas.FillRect(StringGrid2.CellRect(0, 0));

StringGrid1.Canvas.FillRect(StringGrid1.CellRect(i, 0));

Sleep(500);

StringGrid1.Cells[i, 0] := StringGrid2.Cells[0, 0];

StringGrid2.Cells[0, 0] := '';

for v :=0 to stringgrid2.ColCount-2 do

begin

StringGrid2.Cells[v, 0] := StringGrid2.Cells[v+1, 0];

stringgrid2.Cells[v+1,0]:='';

end;

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

exit;

end;

end;

if (StringGrid1.ColCount < 12) and (StringGrid2.Cells[0, 0] <> '') then

begin

Edit1.Text := IntToStr(StringGrid1.ColCount + 1);

StringGrid1.Width := 56 \* StringGrid1.ColCount;

StringGrid2.Canvas.Brush.Color := clRed;

StringGrid1.Canvas.Brush.Color := clGreen;

StringGrid2.Canvas.FillRect(StringGrid2.CellRect(0, 0));

StringGrid1.Canvas.FillRect(StringGrid1.CellRect(i, 0));

Sleep(500);

for v := stringgrid1.ColCount-1 downto 1 do

begin

stringgrid1.Cells[v,0]:=stringgrid1.Cells[v-1,0];

stringgrid1.cells[v-1,0]:='';

end;

StringGrid1.Cells[0, 0] := StringGrid2.Cells[0, 0];

for c := 0 to stringgrid2.ColCount-2 do

begin

stringgrid2.cells[c,0]:=stringgrid2.Cells[c+1,0];

stringgrid2.Cells[c+1,0]:='';

end;

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end

else

ShowMessage('Невозможно добавить новое число');

end;

procedure TForm3.Button3Click(Sender: TObject);

var i:integer;

begin

if stringgrid1.ColCount>1 then

begin

stringgrid1.Cells[0,0]:='';

stringgrid1.canvas.brush.Color:=clred;

stringgrid1.canvas.fillrect(stringgrid1.cellrect(0,0));

sleep(500);

for i := 0 to stringgrid1.ColCount-2 do

begin

stringgrid1.Cells[i,0]:=stringgrid1.cells[i+1,0];

stringgrid1.Cells[i+1,0]:='';

end;

edit1.Text:=inttostr(stringgrid1.ColCount-1);

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end

else

showmessage('Невозможно удалить элементы');

end;

procedure TForm3.Button4Click(Sender: TObject);

var i:integer;

begin

for i:= 0 to stringgrid1.colcount-1 do

stringgrid1.Cells[i,0]:='';

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end;

procedure TForm3.Button5Click(Sender: TObject);

begin

form3.close;

form4.show;

end;

procedure TForm3.Button6Click(Sender: TObject);

begin

form3.Close;

form5.show;

end;

procedure TForm3.Button7Click(Sender: TObject);

begin

form3.Close;

form7.show;

end;

procedure TForm3.Button8Click(Sender: TObject);

var i:integer; nalichie:boolean;

begin

nalichie:=false;

if stringgrid3.Cells[0,0]<>'' then

begin

for i := 0 to stringgrid1.ColCount-1 do

begin

if stringgrid1.Cells[i,0]=stringgrid3.Cells[0,0] then

nalichie:=true;

end;

if nalichie=true then

begin

label6.visible:=true;

label6.Caption:='Элемент есть в стеке';

timer1.Enabled:=true;

end

else

begin

label6.Visible:=true;

label6.Caption:='Элемента нет в стеке';

timer1.Enabled:=true;

end;

end;

end;

procedure TForm3.Button9Click(Sender: TObject);

begin

if stringgrid1.Cells[0, 0] <> '' then

begin

label7.Visible:=true;

label7.Caption:='Последний элемент='+stringgrid1.cells[0,0];

timer1.Enabled:=true;

end

else

showmessage('Элемента не существует');

end;

procedure TForm3.Edit1Change(Sender: TObject);

var i:integer;

begin

if (strtoint(edit1.Text)>0) and (strtoint(edit1.Text)<13) then

begin

stringgrid1.colcount:=strtoint(edit1.Text);

stringgrid1.Width:=56\*strtoint(edit1.Text);

for i := strtoint(edit1.Text) to stringgrid1.colcount-1 do

stringgrid1.Cells[i,0]:='';

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end

else

begin

showmessage('Введено некорректное число');

edit1.text:='1';

end;

end;

procedure TForm3.FormActivate(Sender: TObject);

begin

panel1.Color:=RGB(77,131,201);

panel2.color:=RGB(77,131,201);

panel3.Color:=rgb(97, 158, 237);

panel4.Color:=rgb(97, 158, 237);

end;

procedure TForm3.Image1Click(Sender: TObject);

begin

form3.Close;

form1.Show;

end;

procedure TForm3.StringGrid1SetEditText(Sender: TObject; ACol, ARow: Integer;

const Value: string);

var i:integer;

pusto:boolean;

begin

pusto:=true;

for i := 0 to stringgrid1.ColCount-1 do

begin

if stringgrid1.Cells[i, 0] <> '' then

begin

pusto := false;

label4.Visible:=true;

label4.Left:=246+(55\*i);

exit;

end;

end;

if pusto=true then

label4.Visible:=false;

end;

procedure TForm3.Timer1Timer(Sender: TObject);

begin

label6.Visible:=false;

label7.Visible:=false;

timer1.Enabled:=false;

end;

end.

unit Unit4;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Grids,

Vcl.Imaging.pngimage, Vcl.ExtCtrls;

type

TForm4 = class(TForm)

StringGrid1: TStringGrid;

Edit1: TEdit;

Button1: TButton;

Button2: TButton;

Button3: TButton;

Button4: TButton;

label3: TLabel;

StringGrid2: TStringGrid;

Button5: TButton;

Button6: TButton;

Label4: TLabel;

Label5: TLabel;

Button7: TButton;

Panel1: TPanel;

Panel2: TPanel;

Label2: TLabel;

Image1: TImage;

Label1: TLabel;

Button8: TButton;

Button9: TButton;

Label6: TLabel;

StringGrid3: TStringGrid;

Label7: TLabel;

Label8: TLabel;

Panel3: TPanel;

Panel4: TPanel;

Timer1: TTimer;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Edit1Change(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Button6Click(Sender: TObject);

procedure StringGrid1SetEditText(Sender: TObject; ACol, ARow: Integer;

const Value: string);

procedure Button7Click(Sender: TObject);

procedure FormActivate(Sender: TObject);

procedure Button8Click(Sender: TObject);

procedure Button9Click(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.dfm} uses math, unit3, unit1,unit5,unit7;

procedure TForm4.Button1Click(Sender: TObject);

var

n,i:integer;

begin

for i:=0 to stringgrid1.ColCount-1 do

begin

randomize;

stringgrid1.Cells[i,0]:=inttostr(randomrange(-10,10));

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end;

end;

procedure TForm4.Button2Click(Sender: TObject);

var i,v,c:integer;

begin

for I := 0 to stringgrid1.colcount-1 do

begin

if stringgrid1.cells[I,0] = '' then

begin

stringgrid2.canvas.brush.Color:=clred;

stringgrid1.Canvas.Brush.Color:=clgreen;

stringgrid2.canvas.fillrect(stringgrid2.cellrect(0,0));

stringgrid1.Canvas.FillRect(stringgrid1.cellrect(i,0));

sleep(500);

stringgrid1.Cells[I,0] := stringgrid2.cells[0,0];

stringgrid2.Cells[0,0]:='';

for v := 0 to stringgrid1.ColCount-2 do

begin

stringgrid2.Cells[v,0]:=stringgrid2.cells[v+1,0];

stringgrid2.Cells[v+1,0]:='';

end;

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

Exit;

end;

end;

if (stringgrid1.ColCount < 12) and (stringgrid2.Cells[0,0]<>'') then

begin

edit1.Text:=inttostr(stringgrid1.ColCount+1);

stringgrid1.Width:=56\*stringgrid1.ColCount;

stringgrid2.canvas.brush.Color:=clred;

stringgrid1.Canvas.Brush.Color:=clgreen;

stringgrid2.canvas.fillrect(stringgrid2.cellrect(0,0));

stringgrid1.Canvas.FillRect(stringgrid1.cellrect(i,0));

sleep(500);

stringgrid1.Cells[stringgrid1.ColCount-1, 0] := stringgrid2.cells[0,0];

for c := 0 to stringgrid2.ColCount-1 do

begin

stringgrid2.Cells[c,0]:=stringgrid2.cells[c+1,0];

stringgrid2.Cells[c+1,0]:='';

end;

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end

else

ShowMessage('Невозможно добавить новое число');

end;

procedure TForm4.Button3Click(Sender: TObject);

var i:integer;

begin

if stringgrid1.ColCount>1 then

begin

stringgrid1.Cells[0,0]:='';

stringgrid1.canvas.brush.Color:=clred;

stringgrid1.canvas.fillrect(stringgrid1.cellrect(0,0));

sleep(500);

for i := 0 to stringgrid1.ColCount-1 do

begin

stringgrid1.Cells[i,0]:=stringgrid1.cells[i+1,0];

stringgrid1.Cells[i+1,0]:='';

end;

stringgrid1.Width:=56\*stringgrid1.colcount;

edit1.Text:=inttostr(stringgrid1.ColCount-1);

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end

else

showmessage('Невозможно удалить элементы');

end;

procedure TForm4.Button4Click(Sender: TObject);

begin

var i:integer;

begin

for i:= 0 to stringgrid1.colcount-1 do

stringgrid1.Cells[i,0]:='';

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end;

end;

procedure TForm4.Button5Click(Sender: TObject);

begin

form4.Close;

form3.Show;

end;

procedure TForm4.Button6Click(Sender: TObject);

begin

form4.Close;

form5.show;

end;

procedure TForm4.Button7Click(Sender: TObject);

begin

form4.Close;

form7.show;

end;

procedure TForm4.Button8Click(Sender: TObject);

var i:integer; nalichie:boolean;

begin

nalichie:=false;

if stringgrid3.Cells[0,0]<>'' then

begin

for i := 0 to stringgrid1.ColCount-1 do

begin

if stringgrid1.Cells[i,0]=stringgrid3.Cells[0,0] then

nalichie:=true;

end;

if nalichie=true then

begin

label7.visible:=true;

label7.Caption:='Элемент есть в стеке';

timer1.Enabled:=true;

end

else

begin

label7.Visible:=true;

label7.Caption:='Элемента нет в стеке';

timer1.Enabled:=true;

end;

end;

end;

procedure TForm4.Button9Click(Sender: TObject);

begin

if stringgrid1.Cells[stringgrid1.ColCount-1, 0] <> '' then

begin

label8.Visible:=true;

label8.Caption:='Последний элемент='+stringgrid1.cells[stringgrid1.ColCount-1,0];

timer1.Enabled:=true;

end

else

showmessage('Элемента не существует');

end;

procedure TForm4.Edit1Change(Sender: TObject);

var i:integer;

begin

if (strtoint(edit1.Text)>0) and (strtoint(edit1.Text)<13) then

begin

stringgrid1.colcount:=strtoint(edit1.Text);

stringgrid1.Width:=56\*strtoint(edit1.Text);

for i := strtoint(edit1.Text) to stringgrid1.colcount-1 do

stringgrid1.Cells[i,0]:='';

StringGrid1SetEditText(Sender, 0, 0, StringGrid1.Cells[0, 0]);

end

else

begin

showmessage('Введено некорректное число');

edit1.text:='1';

end;

end;

procedure TForm4.FormActivate(Sender: TObject);

begin

panel1.Color:=RGB(77,131,201);

panel2.color:=RGB(77,131,201);

panel3.Color:=rgb(97, 158, 237);

panel4.Color:=rgb(97, 158, 237);

end;

procedure TForm4.Image1Click(Sender: TObject);

begin

form4.close;

form1.Show;

end;

procedure TForm4.StringGrid1SetEditText(Sender: TObject; ACol, ARow: Integer;

const Value: string);

var

i: Integer;

pusto: Boolean;

begin

pusto := true;

for i := 0 to stringgrid1.ColCount - 1 do

begin

if stringgrid1.Cells[i, 0] <> '' then

begin

pusto := false;

label5.Visible:=true;

label5.Left:=254+(55\*i);

end;

end;

for i := 0 to stringgrid1.ColCount-1 do

begin

if stringgrid1.Cells[i,0]<>'' then

begin

label4.Visible:=true;

label4.Left:=254+(55\*i);

exit;

end;

end;

if pusto then

begin

label4.Visible := false;

label5.Visible := false;

end;

end;

procedure TForm4.Timer1Timer(Sender: TObject);

begin

label7.Visible:=false;

label8.Visible:=false;

timer1.Enabled:=false;

end;

end.

unit Unit5;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls,

Vcl.Imaging.pngimage, Vcl.ComCtrls, Vcl.OleCtrls, SHDocVw;

type

TForm5 = class(TForm)

Image5: TImage;

TabControl1: TTabControl;

WebBrowser1: TWebBrowser;

procedure Image5Click(Sender: TObject);

procedure TabControl1Change(Sender: TObject);

procedure FormShow(Sender: TObject);

private

{ Private declarations }

public

f:text; s:string;

end;

var

Form5: TForm5;

implementation

{$R \*.dfm} uses unit3,unit4,unit6,unit1;

procedure TForm5.FormShow(Sender: TObject);

begin

form5.Color:=rgb(77,131,201);

webbrowser1.Navigate(ExtractFilePath(Application.ExeName)+'Функции\_стека.htm');

end;

procedure TForm5.Image5Click(Sender: TObject);

begin

form5.Close;

form1.show;

end;

procedure TForm5.TabControl1Change(Sender: TObject);

begin

case tabcontrol1.tabindex of

0:

begin

webbrowser1.Navigate(ExtractFilePath(Application.ExeName)+'Функции\_стека.htm');

end;

1:

begin

webbrowser1.Navigate(ExtractFilePath(Application.ExeName)+'Примеры\_стека.htm');

end;

2:

begin

webbrowser1.Navigate(ExtractFilePath(Application.ExeName)+'Функции\_очереди.htm');

end;

3:

begin

webbrowser1.Navigate(ExtractFilePath(Application.ExeName)+'Примеры\_очереди.htm');

end;

end;

end;

end.

unit Unit7;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls,

Vcl.Imaging.pngimage;

type

TForm7 = class(TForm)

Label1: TLabel;

Label2: TLabel;

Timer1: TTimer;

Button1: TButton;

RadioGroup1: TRadioGroup;

Label3: TLabel;

Image1: TImage;

Label4: TLabel;

edit1: TEdit;

Timer2: TTimer;

procedure FormCreate(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Timer2Timer(Sender: TObject);

private

{ Private declarations }

public

s:string;

f:text;

ball:integer; nvern:integer;

end;

var

Form7: TForm7;

implementation

{$R \*.dfm} uses unit4,unit3,unit1;

procedure TForm7.FormCreate(Sender: TObject);

begin

radiogroup1.Color:=rgb(97, 158, 237);

form7.Color:=rgb(77,131,201);

label2.caption:='Данный тест на 15 вопросов рассчитан на 5 минут.'+#13#10'Результат считается удовлетворительным если как минимум ответы на 10 из 15 вопросов правильные'+#13#10+'Таймер запустится после ответа на первый вопрос';

assignfile(f,ExtractFilePath(Application.ExeName)+'test.txt');

ball:=0;

Reset(f);

Readln(f,s);

repeat

if(s[1]='-') then

begin

delete(s,1,1);

radiogroup1.caption:=s;

end

else if s[1]='\*' then

begin

delete(s,1,1);

nvern:=StrToInt(s)

end

else radiogroup1.Items.Add(s);

readln(f,s);

until (s[1]='-') or (eof(f));

end;

procedure TForm7.Image1Click(Sender: TObject);

begin

form7.Close;

form1.show;

end;

procedure TForm7.Timer1Timer(Sender: TObject);

begin

edit1.Text:=inttostr(strtoint(edit1.Text)-1);

end;

procedure TForm7.Timer2Timer(Sender: TObject);

begin

timer1.Enabled:=false;

radiogroup1.Enabled:=false;

button1.Enabled:=false;

label3.Caption := 'Правильных ответов: '+inttostr(ball)+#13#10+'Процент правильных ответов:'+floattostr(ball/15\*100)+'%';

end;

procedure TForm7.Button1Click(Sender: TObject);

begin

timer1.Enabled:=true;

timer2.Enabled:=true;

if (radiogroup1.ItemIndex > -1) and (not Eof(f)) then

begin

if radiogroup1.ItemIndex = Nvern-1 then ball:=ball+1;

radiogroup1.Items.Clear();

Repeat

if Length(s) >= 1 then

begin

if (s[1] = '-') then

begin

Delete(s, 1, 1);

radiogroup1.Caption := s;

end

else if (s[1] = '\*') then

begin

Delete(s, 1, 1);

Nvern := StrToInt(s);

end

else

begin

radiogroup1.Items.Add(s);

end;

Readln(f, s);

end;

until (Length(s) < 1) or (s[1] = '-');

end

else if Eof(f) then

begin

if radiogroup1.ItemIndex = Nvern - 1 then

begin

ball:= ball+1;

end;

radiogroup1.Enabled:=false;

label3.Caption := 'Правильных ответов: '+inttostr(ball)+#13#10+'Процент правильных ответов:'+floattostr((ball/15\*100))+'%';

CloseFile(f);

radiogroup1.Enabled:=false;

button1.Enabled := False;

timer1.Enabled:=false;

end;

end;