Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ

УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра радиоэлектронных технологий и экологического мониторинга (РЭТЭМ)

Отчет по практической работе №3 по дисциплине «Информатика

и программирование»

Выполнили:

Студенты гр. 234-3

\_\_\_\_\_\_\_\_\_\_\_/С.М. Мелконян

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2024 г.

Проверил:

Руководитель

\_\_\_\_\_\_\_\_\_\_\_\_/К.Н. Афонкин «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г.

2024

СОДЕРЖАНИЕ

ВВДЕНИЕ…………………………………………………………………...3

1. Ход работы………………………………………………………………2

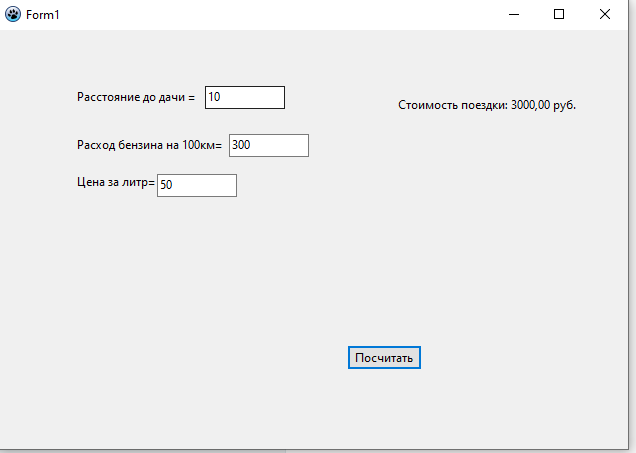
Задание 1. Создание интерфейса для ввода данных…………………...3

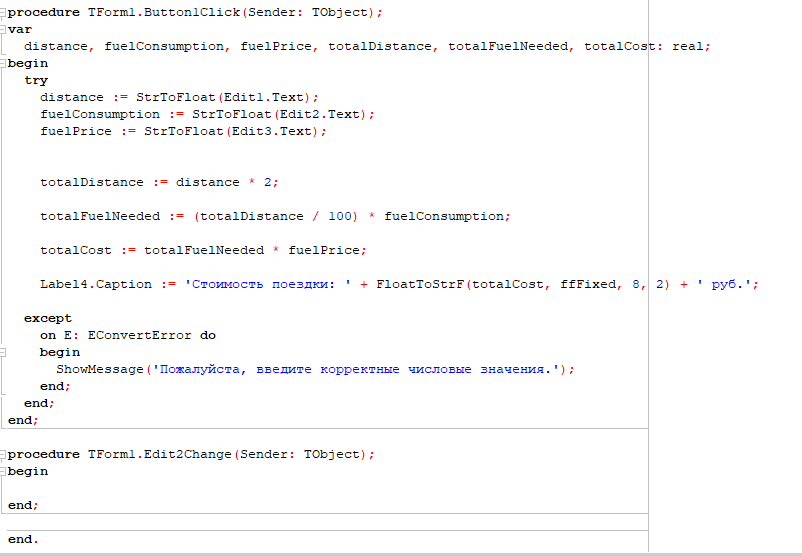
Задание 2. Написание кода……………………………………………...4

Задание 3. Вывод работы функции…………………….……………....5

ЗАКЛЮЧЕНИЕ………………………………………………………….…6

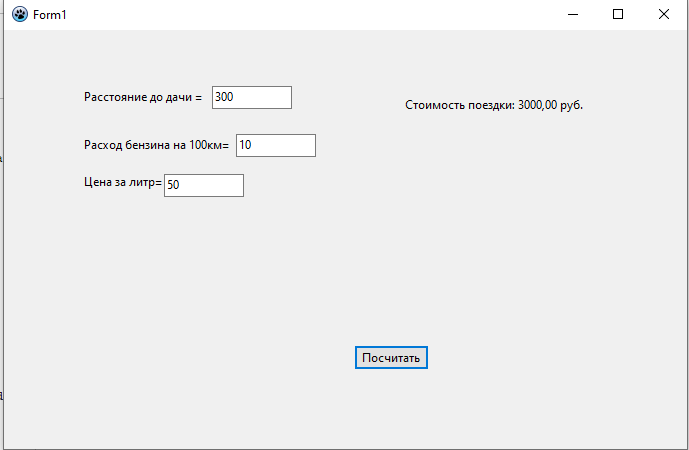
**Создание интерфейс для ввода данных пользователю**

Для расчета стоимости поездки, я создал интерфейс для пользователя  


**Написание кода**  


Создаем переменные для подсчета стоимости поездки и благодаря формулам высчитываем итоговую стоимость поездки

**Вывод работы функции**



При вводе всех значений пользователь получает итоговую стоимость поездки

ЗАКЛЮЧЕНИЕ

В лабораторной работе №3, я создал форму для подсчета затрат на дорогу. Благодаря введенным данным со стороны пользователя, я смог посчитать стоимость затрат на дорогу.   
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
 Вопросы

  1. Определение объектно-ориентированного программирования (ООП) и его достоинства и недостатки  
Объектно-ориентированное программирование (ООП) — это методология разработки программного обеспечения, основанная на использовании объектов, которые представляют собой экземпляры классов. ООП объединяет данные и функции, взаимодействующие с этими данными, в единые структуры, называемые объектами, позволяя разработчикам моделировать сложные системы более естественным способом.  
  
Достоинства ООП:  
- Инкапсуляция: Позволяет скрыть внутренние детали реализации и предоставлять пользователям только интерфейс для взаимодействия с объектами.  
- Наследование: Позволяет создавать новые классы на основе уже существующих, что облегчает повторное использование кода.  
- Полиморфизм: Позволяет использовать один и тот же интерфейс для различных типов объектов, что облегчает расширение и модификацию программы.  
- Модульность: Объекты и классы позволяют структурировать код, делая его более удобным для разработки, тестирования и поддержки.  
  
Недостатки ООП:  
- Сложность: Концепции ООП могут быть сложны для понимания, особенно для новичков.  
- Накладные расходы: ООП может ввести дополнительную сложность из-за создания объектов, что может влиять на производительность.  
- \*Проектирование: Требует более тщательного проектирования системы, чтобы правильно определить классы, их свойства и методы.  
  
 2. Классы, объекты, интерфейсы  
- Класс: Шаблон для создания объектов, который определяет свойства (данные) и методы (функции) объектов. Например, класс "Автомобиль" может иметь свойства "цвет" и "модель" и метод "поездка".  
- Объект: Конкретный экземпляр класса, который использует свойства и методы, заданные классом. Например, "Мой\_автомобиль" — это объект класса "Автомобиль".  
- Интерфейс: Абстрактный тип, который определяет набор методов без их реализации. Классы, реализующие интерфейс, обязаны предоставлять реализацию этих методов. Интерфейсы позволяют достигать полиморфизма.  
  
 3. Принципы инкапсуляции, полиморфизма, наследования  
- Инкапсуляция: Соблюдение доступа к данным объекта и их защита от воздействия извне, что позволяет предотвратить некорректное использование данных и предоставляет четкий интерфейс для взаимодействия с объектом.  
- Полиморфизм: Способность объектов разных классов обрабатывать единый интерфейс, что позволяет использовать одну и ту же функцию для вызова различных реализаций в зависимости от того, к какому классу принадлежат объекты.  
- Наследование: Создание новых классов на основе уже существующих, что позволяет использовать уже написанный код и вносить к нему изменения или расширения.  
  
 4. Принципы технологии ООП  
Основаниями технологии ООП являются:  
- Инкапсуляция  
- Наследование  
- Полиморфизм  
- Абстракция: Упрощение представления объектов путем выделения наиболее важных характеристик и скрытия несущественной информации.  
  
 5. Визуальное проектирование интерфейса  
Визуальное проектирование интерфейса — это процесс создания графического интерфейса пользователя (GUI) с помощью инструментов и технологий, позволяющих разработчикам быстро и удобно размещать элементы управления на форме, без необходимости ручного написания кода. Этот процесс упрощает создание привлекательных и удобных интерфейсов.  
  
 6. Основные элементы пользовательского интерфейса Lazarus  
- Кнопки (Button)  
- Текстовые поля (Edit)  
- Метки (Label)  
- Списки (ListBox)  
- Комбинированные списки (ComboBox)  
- Чекбоксы (CheckBox)  
- Переключатели (RadioButton)  
- Панели (Panel)  
- Группы (GroupBox)

7. Функции палитры компонентов в Lazarus  
Палитра компонентов предоставляет доступ к различным графическим и невизуальным компонентам, которые можно перетаскивать на форму. Это упрощает создание пользовательских интерфейсов, позволяя разработчикам выбирать необходимые элементы для проекта, управлять их свойствами и размещать их на экране.  
  
 8. Инспектор объектов  
Инспектор объектов — это инструмент, который позволяет разработчику модифицировать свойства выбранного компонента на форме. Он предоставляет удобный интерфейс для изменения атрибутов таких как размеры, цвета, текст и другие параметры компонента.  
  
 9. Основные файлы проекта Lazarus  
- .lpi (Lazarus Project File): Файл, который содержит информацию о конфигурации проекта, включая все файлы, необходимые для сборки проекта.  
- .lpr (Lazarus Project Run File): Файл, содержащий основной код программы, который начинается с описания самой программы.  
- .lfm (Lazarus Form File): Файл, содержащий описание формы и использованных в ней компонентов.  
- .ppu (Pascal Unit): Файлы с компилированными единицами, которые могут использоваться другими модулями.  
  
 10. Линейный алгоритм  
Линейный алгоритм — это последовательность операций, которая выполняется одна за другой, без каких-либо циклов или разветвлений. В линейном алгоритме каждый оператор выполняется строго по порядку, от первого к последнему. Он состоит из следующих операторов: - Объявления переменных (если нужно) - Присваивания значений переменным  
- Ввод данных  
- Выполнение различных вычислений  
- Вывод данных