Министерство науки и образования РФ

Федеральное государственное бюджетное учреждение

высшего образования

**«Тверской государственный технический университет»**

(ТвГТУ)

Кафедра программного обеспечения

**Отчет по лабораторной работе №3**

по дисциплине: «Теория алгоритмов»

|  |
| --- |
| Выполнил:  студент группы  Б.ПИН.РИС-22.06  Мельников М.Н. |
| Проверила:  старший преподаватель  кафедры ПО  Корнеева Е.И. |

Тверь 2024

Описание задачи.

Реализовать приложение “Сервис по доставке продуктов” (Вариант 17) с GUI.

**Требования:**

* Приложение должно быть написано с применением ООП парадигмы
* Приложение должно быть небольшим.
* Визуальная часть должна содержать таблицу по теме варианта, поля ввода новых данных, график активности, логирование активности, кнопки и лейблы (labels, текст)
* Приложение должно быть с меню и выходом.
* Исключительные ситуации должны обрабатываться с использованием собственных исключений

Репозиторий.

[TheoryAlgLabs/Labs/Lab3 at main · Zawiaha/TheoryAlgLabs](https://github.com/Zawiaha/TheoryAlgLabs/tree/main/Labs/Lab2)

UML – диаграммы.



Проделанная работа.

В ходе выполнения лабораторной было реализовано 3 класса.

Класс Product хранит информацию о продукте

* Название
* Количество
* Цена
* Время доставки
* Адрес доставки

Класс DeliveryService предоставляет функционал для управления таблицей заказов.

* Добавление заказа
* Удалить заказ из таблицы
* Логирование активности

Класс DeliveryApp предоставляет интерфейс программы с использование библиотеки tkinter

Вывод

Данная лабораторная работа посвящена разработке системы управления заказами для доставки продуктов. Такая система позволяет упорядочить и автоматизировать процесс обработки заказов, что является важной задачей для предприятий, занимающихся доставкой продуктов питания.

Использованная литература.

<https://docs.python.org/> - документация по Python

Код

# Product.py

class Product:

    def \_\_init\_\_(self, name, quantity, price, deliveryTime, address):

        self.name = name

        self.quantity = quantity

        self.price = price

        self.deliveryTime = deliveryTime

        self.address = address

# DeliveryService

import logging

from product import Product

logging.basicConfig(filename='deliveryService.log', level=logging.INFO, format='%(asctime)s - %(message)s')

class DeliveryService:

    def \_\_init\_\_(self):

        self.productList = []

    def addProduct(self, product):

        if not isinstance(product, Product):

            raise ValueError("Invalid product")

        self.productList.append(product)

        logging.info(f"Product added: {product.name}, Quantity: {product.quantity}, Price: {product.price}, Delivery Time: {product.deliveryTime}, Address: {product.address}")

    def getProducts(self):

        return self.productList

    def removeProduct(self, index):

        if 0 <= index < len(self.productList):

            removedProduct = self.productList.pop(index)

            logging.info(f"Product removed: {removedProduct.name}")

        else:

            raise IndexError("Invalid index")

# DeliveryApp.py

import tkinter as tk

from tkinter import messagebox, ttk

from deliveryService import DeliveryService

from product import Product

class DeliveryApp:

    def \_\_init\_\_(self, root):

        self.deliveryService = DeliveryService()

        self.root = root

        self.root.title("Сервис по доставке продуктов")

        # Меню

        menuBar = tk.Menu(root)

        fileMenu = tk.Menu(menuBar, tearoff=0)

        fileMenu.add\_command(label="Выход", command=root.quit)

        menuBar.add\_cascade(label="Файл", menu=fileMenu)

        root.config(menu=menuBar)

        # Поля ввода

        self.nameLabel = tk.Label(root, text="Название продукта:")

        self.nameLabel.pack()

        self.nameEntry = tk.Entry(root)

        self.nameEntry.pack()

        self.quantityLabel = tk.Label(root, text="Количество:")

        self.quantityLabel.pack()

        self.quantityEntry = tk.Entry(root)

        self.quantityEntry.pack()

        self.priceLabel = tk.Label(root, text="Цена:")

        self.priceLabel.pack()

        self.priceEntry = tk.Entry(root)

        self.priceEntry.pack()

        self.deliveryTimeLabel = tk.Label(root, text="Время доставки:")

        self.deliveryTimeLabel.pack()

        self.deliveryTimeEntry = tk.Entry(root)

        self.deliveryTimeEntry.pack()

        self.addressLabel = tk.Label(root, text="Адрес доставки:")

        self.addressLabel.pack()

        self.addressEntry = tk.Entry(root)

        self.addressEntry.pack()

        self.addButton = tk.Button(root, text="Добавить продукт", command=self.addProduct)

        self.addButton.pack()

        self.removeButton = tk.Button(root, text="Удалить продукт", command=self.removeProduct)

        self.removeButton.pack()

        # Таблица для отображения продуктов

        self.tree = ttk.Treeview(root, columns=('Name', 'Quantity', 'Price', 'Delivery Time', 'Address'), show='headings')

        self.tree.heading('Name', text='Название')

        self.tree.heading('Quantity', text='Количество')

        self.tree.heading('Price', text='Цена')

        self.tree.heading('Delivery Time', text='Время доставки')

        self.tree.heading('Address', text='Адрес')

        self.tree.pack()

    def addProduct(self):

        name = self.nameEntry.get()

        quantity = self.quantityEntry.get()

        price = self.priceEntry.get()

        deliveryTime = self.deliveryTimeEntry.get()

        address = self.addressEntry.get()

        if not name or not quantity or not price or not deliveryTime or not address:

            messagebox.showerror("Ошибка", "Все поля должны быть заполнены")

            return

        try:

            quantity = int(quantity)

            price = float(price)

            product = Product(name, quantity, price, deliveryTime, address)

            self.deliveryService.addProduct(product)

            self.updateTable()

            messagebox.showinfo("Успех", "Продукт добавлен успешно")

        except ValueError as e:

            messagebox.showerror("Ошибка", str(e))

    def removeProduct(self):

        selectedItem = self.tree.selection()

        if not selectedItem:

            messagebox.showerror("Ошибка", "Выберите продукт для удаления")

            return

        index = self.tree.index(selectedItem[0])

        try:

            self.deliveryService.removeProduct(index)

            self.updateTable()

            messagebox.showinfo("Успех", "Продукт удален успешно")

        except IndexError as e:

            messagebox.showerror("Ошибка", str(e))

    def updateTable(self):

        for row in self.tree.get\_children():

            self.tree.delete(row)

        for product in self.deliveryService.getProducts():

            self.tree.insert('', 'end', values=(product.name, product.quantity, product.price, product.deliveryTime, product.address))

# main.py

import tkinter as tk

from gui import DeliveryApp

if \_\_name\_\_ == "\_\_main\_\_":

    root = tk.Tk()

    app = DeliveryApp(root)

    root.mainloop()