

Санкт-Петербургский Национальный  
Исследовательский Университет  
Информационных технологий, механики и оптики

**Лабораторная работа 1**  
**Программа для контроля собственных**  
**денежных средств**

Выполнил: Буров  
Глеб  
Группа № К3123  
Проверила: Казанова  
Полина Петровна

Санкт-Петербург  
СОДЕРЖАНИЕ

	Стр.
ЦЕЛЬ РАБОТЫ.....	3
ЗАДАЧИ .....	4
1   Ход работы.....	5
1.1   Описание этапа анализа предметной области и требований .....	5
1.2   Общий вид программы .....	5
1.3   Описание возможностей программы.....	7
ЗАКЛЮЧЕНИЕ.....	17

## ЦЕЛЬ РАБОТЫ

Целью работы является создание приложения, которое используется для контроля собственных денежных средств, с помощью языка программирования python,

## ЗАДАЧИ

В приложении необходимо реализовать следующие функции:

1. Добавление продукта в коллекцию.
2. Просмотр всего записанного в программу.
3. Просмотр покупок по дате и категории.
4. Распределение покупок по стоимости от минимальной к максимальной или наоборот.
5. Удаление требуемых записей и выход из программы

## 1 Ход работы

### 1.1 Описание этапа анализа предметной области и требований

Приложение предназначено для контроля собственных денежных средств. Для работы требовалось написать программу на языке python, которая будет удовлетворять всем требованиям. Основные функции – просмотр коллекции товаров с возможностью добавления и удаления позиций, просмотр покупок, включающий в себя просмотр по категории, дате или просмотр всех покупок, с возможностью сортировки по цене (как по возрастанию, так и по убыванию). Необходимо было реализовать сохранение всех данных. Также нужно было создать блок-схему, отписывающую работу программы (см. Рисунок 0).

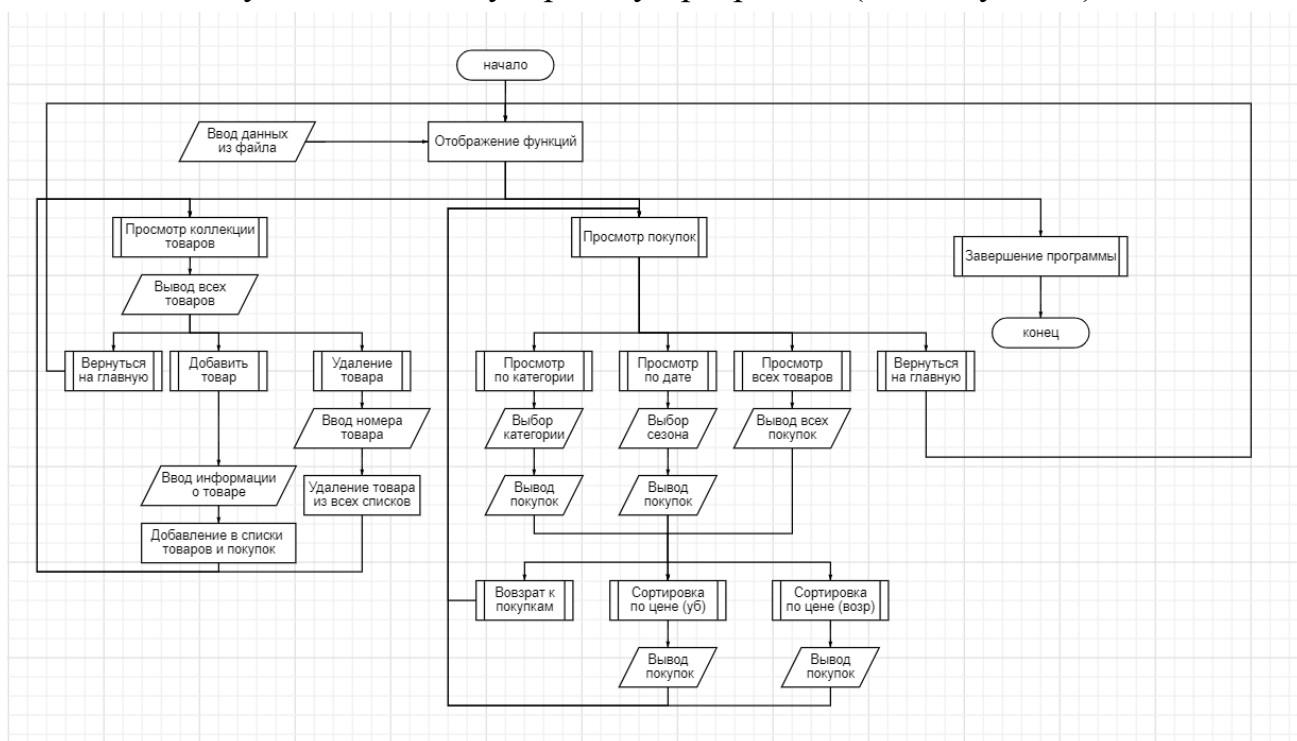
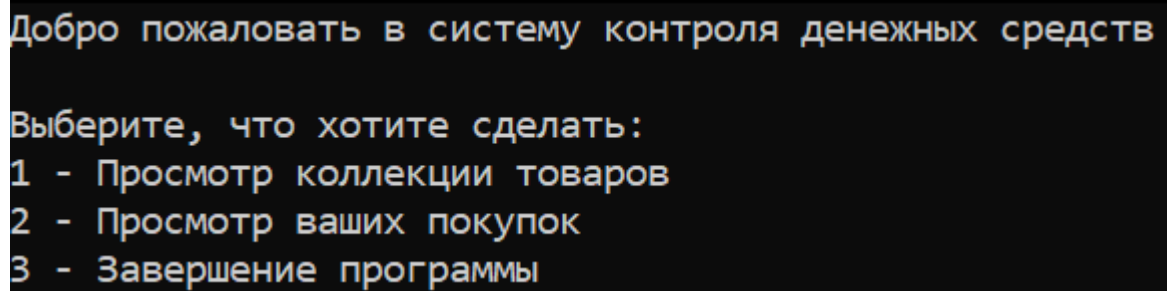


Рисунок 0 – Блок-схема

### 1.2 Общий вид программы

Работа с программой осуществляется через консоль (см. Рисунок 1).



```
Добро пожаловать в систему контроля денежных средств

Выберите, что хотите сделать:
1 - Просмотр коллекции товаров
2 - Просмотр ваших покупок
3 - Завершение программы
```

Рисунок 1– Общий вид программы

Старт программы осуществляется с помощью двух функций: start и chose. Функция start выполняется при запуске программы, она считывает с файлов products.txt и pokupki.txt данные о товарах и покупках в одноименные списки (см. Рисунок 2).

```
pokupki = []
products =[]

def start():
    f = open('pokupki.txt', 'r', encoding='utf-8')
    global pokupki
    for line in f:
        pokupki.append(line.split('|'))
    for i in pokupki:
        o = i[-1].strip('\n')
        i[-1] = o
    for i in pokupki:
        k = int(i[2])
        i[2]=k
    f.close()

    f = open('products.txt', 'r', encoding='utf-8')
    global products
    products = f.read().splitlines()
    f.close()
```

Рисунок 2 – Функция start

Функция chose же реализует вывод строк с выбором действий программы и, в случае выбора, вызывает необходимые функции (см. Рисунок 3). Реализован контроль ввода, который выводит сообщение в случае неправильного выбора.

```

def choise():
    print()
    print('Выберите, что хотите сделать:')
    print('1 - Просмотр коллекции товаров')
    print('2 - Просмотр ваших покупок')
    print('3 - Завершение программы')
    while True:
        print()
        ch = input()
        if ch == '1':
            return prosmotr_products()
        elif ch == '2':
            return prosmotr_pokupki()
        elif ch == '3':
            print('Завершение программы, до новых встреч!')
            return end()

    else:
        print('Введено некорректное действие, повторите попытку')

```

Рисунок 3 – Функция choise

### 1.3 Описание возможностей программы

Программа имеет множество функций. Для каждой из них будет описан код и сами функции.

Первая функция – “Просмотр коллекции товаров” осуществляется в Python с помощью prosmotr\_products (см. Рисунок 4).

```

def prosmotr_products():
    print()
    print('Ваши товары:')
    count = 1
    for i in products:
        print(str(count)+' ', i)
        count+=1
    print()
    print('Выберите, что хотите сделать:')
    print('1 - Вернуться на главную')
    print('2 - Добавить товар')
    print('3 - Удалить товар')
    t = 0
    while t != 1:
        print()
        ch1 = input()
        if ch1 == '1':
            return choise()
        elif ch1 == '2':
            return dobav()

        elif ch1 == '3':
            return de(products)
            prosmotr_products()
        else:
            print('Введено некорректное действие, повторите попытку')

```

Рисунок 4 – Функция prosmotr\_products

Как будет выглядеть вызов этой функции для пользователя можно увидеть на рисунке 5.

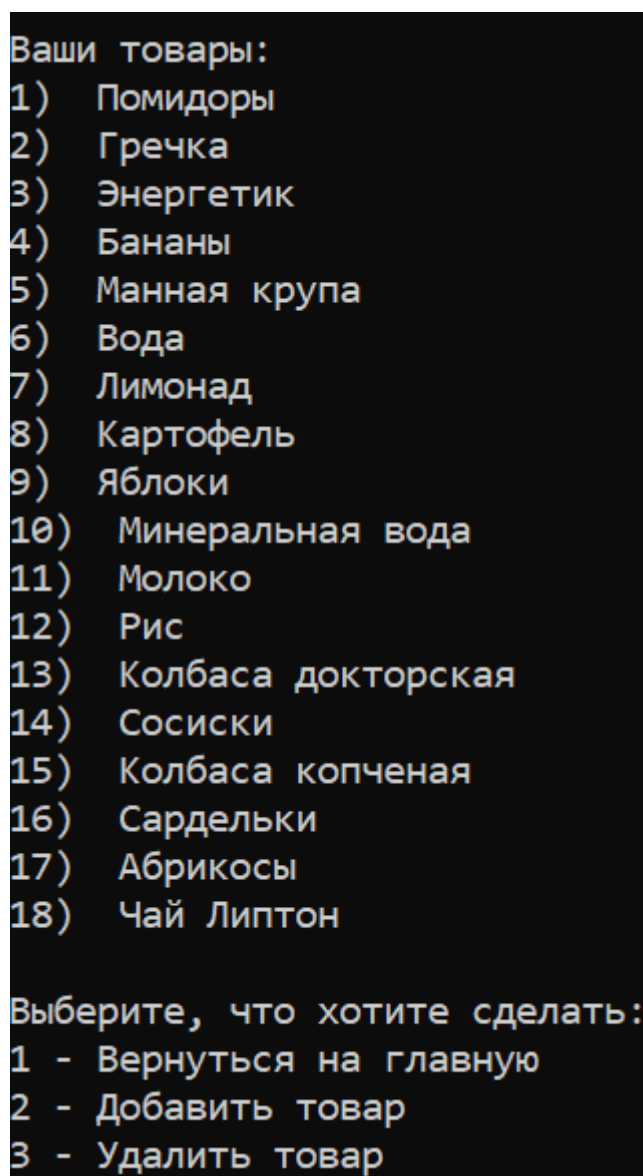


Рисунок 5 – Просмотр коллекции товаров

Пользователю предлагается выбор: вернуться на главную, добавить товар или удалить товар. Если пользователь выбрал возвращение, результатом работы функции `prosmotr_products` является вызов функции `choise`. Если пользователь выбрал “Добавить товар”, запускается функция `dobav`, которая запрашивает у пользователя данные о товаре: название, дату покупки, стоимость и категория, к которой относится продукт (см. Рисунок 6). Предусмотрен контроль при вводе названия и даты – длина названия не должна превышать 30 символов, а дата должна вводиться в указанном формате. В противном случае добавление начинается заново (см. Рисунок 7).



```

Введите продукт, который хотите добавить:

Лимонад "Барбарис"
Введите дату покупки в формате xx.xx.xxxx:
Напоминаем, что программа запоминает покупки, совершенные в 2022 году :)
12.15.2022
Неверно указана дата, попробуйте заново
Введите продукт, который хотите добавить:

Лимонад "Барбарис"
Введите дату покупки в формате xx.xx.xxxx:
Напоминаем, что программа запоминает покупки, совершенные в 2022 году :)
03.05.2022
Введите стоимость товара:
50
Выберите, к какой категории относится товар:
1 - Овощи и фрукты
2 - Крупы
3 - Напитки
4 - Мясные изделия
3
Товар добавлен!

```

Рисунок 6 – Функция добавления товара

```

def dobav():
    new_pokup = []
    global products
    while True:
        print("Введите продукт, который хотите добавить:")
        print()
        tovar = input()
        if len(tovar) > 30:
            print('Слишком длинное название товара, попробуйте еще раз!')
        else:
            if tovar in products:
                print('Товар уже есть в списке!')
                return prosmotr_products()
            else:
                new_pokup.append(tovar)
                print('Введите дату покупки в формате xx.xx.xxxx:')
                print('Напоминаем, что программа запоминает покупки, совершенные в 2022 году :)')
                data = input()
                if len(data)!=10 or int(data[:2]) > 31 or data[2] != '.' or data[5] != '.' or int(data[3:5]) == 0 or int(data[3:5]) > 12 or data[6:] != '2022':
                    print('Неверно указана дата, попробуйте заново')
                else:
                    new_pokup.append(data)
                    print('Введите стоимость товара:')
                    cost = int(input())
                    new_pokup.append(cost)
                    print('Выберите, к какой категории относится товар:')
                    print('1 - Овощи и фрукты')
                    print('2 - Крупы')
                    print('3 - Напитки')
                    print('4 - Мясные изделия')
                    category = input()
                    if category == '1':
                        new_pokup.append('Овощи и фрукты')
                    elif category == '2':
                        new_pokup.append('Крупы')
                    elif category == '3':
                        new_pokup.append('Напитки')
                    elif category == '4':
                        new_pokup.append('Мясные изделия')
                    pokupki.append(new_pokup)
                    products.append(tovar)
                    print('Товар добавлен!')
                    return prosmotr_products()

```

Рисунок 7 – Код функции dobav

Следующая функция - удаление позиции. За нее в коде отвечает de. В ней можно удалить любую позицию из списка товаров. Она же удалится и из списка покупок. При вызове высвечиваются все позиции из файла, после чего можно выбираем позицию для удаления (см. Рисунок 8, Рисунок 9).

```

4) Бананы
5) Манная крупа
6) Вода
7) Лимонад
8) Картофель
9) Яблоки
10) Минеральная вода
11) Молоко
12) Рис
13) Колбаса докторская
14) Сосиски
15) Колбаса копченая
16) Сардельки
17) Абрикосы
18) Чай Липтон
19) Лимонад "Барбарис"

Выберите, что хотите сделать:
1 - Вернуться на главную
2 - Добавить товар
3 - Удалить товар

3

Введите номер товара, который хотите удалить:
19
Данные были отредактированы

```

Рисунок 8 – Функция удаления товара

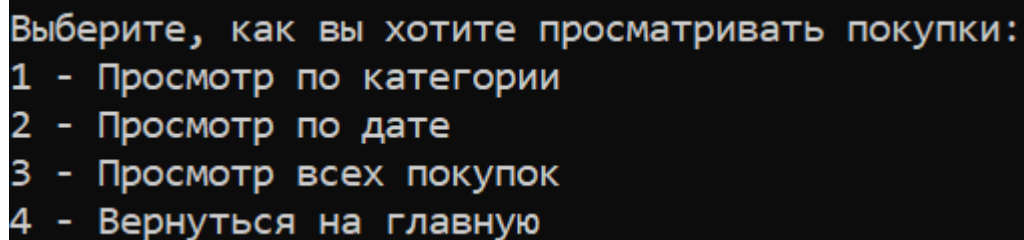
```

def de(arr):
    print()
    print("Введите номер товара, который хотите удалить:")
    n = int(input())
    count = 1
    name = arr[n-1]
    arr.pop(n-1)
    global pokupki
    for i in range(len(pokupki)):
        if name in pokupki[i]:
            pokupki.pop(i)
            break
    print("Данные были отредактированы")
    return prosmotr_products()

```

Рисунок 9 – Код функции de

За просмотр покупок отвечает функция `prosmotr_pokupki`. Если пользователь хочет увидеть свои покупки, ему предлагается выбор: просмотр по категории, просмотр по дате или просмотр всех покупок. Имеется возможность вернуться в главное меню (см. Рисунок 10). Код функции `prosmotr_pokupki` приведен на рисунке 11.



Выберите, как вы хотите просматривать покупки:  
1 - Просмотр по категории  
2 - Просмотр по дате  
3 - Просмотр всех покупок  
4 - Вернуться на главную

Рисунок 10 – Просмотр покупок

```
def prosmotr_pokupki():  
    print()  
    print('Выберите, как вы хотите просматривать покупки:')  
    print('1 - Просмотр по категории')  
    print('2 - Просмотр по дате')  
    print('3 - Просмотр всех покупок')  
    print('4 - Вернуться на главную')  
    while True:  
        print()  
        ch2 = input()  
        if ch2 == '1':  
            return category()  
  
        elif ch2 == '2':  
            return data()  
  
        elif ch2 == '3':  
            return all_pokupki()  
  
        elif ch2 == '4':  
            return chose()  
        else:  
            print('Введено некорректное действие, повторите попытку')
```

Рисунок 11 – Код функции prosmotr\_pokupki

За просмотр по категории отвечают функции `category` и `vibor_categ`. Пользователю предлагаются на выбор 4 категории: “Овощи и фрукты”, “Крупы”, “Напитки” и “Мясные изделия” (см. Рисунок 12). Функция `category` запрашивает выбор, который потом передает `vibor_categ` – функции, отвечающей за вывод (см. Рисунок. 13, Рисунок 14).

```

1
Выберите категорию:
1 - Овощи и фрукты
2 - Крупы
3 - Напитки
4 - Мясные изделия
5 - Назад к выбору

1
1) Помидоры 18.10.2022 90
2) Бананы 30.07.2022 75
3) Картофель 25.04.2022 30
4) Яблоки 01.05.2022 37
5) Абрикосы 11.12.2022 103

1 - Вернуться к остальным покупкам
2 - Сортировать по возрастанию цены
3 - Сортировать по убыванию цены

```

Рисунок 12 – Просмотр по категории

```

def category():
    global pokupki
    print()
    print('Выберите категорию:')
    print('1 - Овощи и фрукты')
    print('2 - Крупы')
    print('3 - Напитки')
    print('4 - Мясные изделия')
    print('5 - Назад к выбору')
    while True:
        print()
        ch3 = input()
        if ch3 == '1':
            return vibor_categ('Овощи и фрукты')
        elif ch3 == '2':
            return vibor_categ('Крупы')
        elif ch3 == '3':
            return vibor_categ('Напитки')
        elif ch3 == '4':
            return vibor_categ('Мясные изделия')
        elif ch3 == '5':
            return prosmotr_pokupki()
        else:
            print('Введено некорректное действие, повторите попытку')

```

Рисунок 13 - Код функции category

```

def vibor_categ(name):
    global pokupki
    arr = []
    count = 1
    for i in pokupki:
        if name in i[-1]:
            arr.append(i)
            print(str(count)+' ' ,i[0],i[1],i[2])
            count+=1
    return cost_sort(arr)

```

Рисунок 14 – код функции vibor\_categ

За просмотр по дате отвечает функция data. Пользователю предлагается на выбор 4 сезона: “Зима”, “Весна”, “Лето”, “Осень”. При выборе одного из них выводятся все покупки за определенный период (см. Рисунок 15, Рисунок 16).

```

2
Выберите время года:
1 - Зима
2 - Весна
3 - Лето
4 - Осень
5 - Назад к выбору

2
1) Картофель 25.04.2022 30
2) Яблоки 01.05.2022 37
3) Молоко 03.03.2022 92
4) Чай Липтон 03.04.2022 140

1 - Вернуться к остальным покупкам
2 - Сортировать по возрастанию цены
3 - Сортировать по убыванию цены

```

Рисунок 15 – Просмотр по дате

```

def data():
    print()
    print('Выберите время года:')
    print('1 - Зима')
    print('2 - Весна')
    print('3 - Лето')
    print('4 - Осень')
    print('5 - Назад к выбору')
    while True:
        print()
        ch3 = input()
        if ch3 == '1':
            arr = []
            count = 1
            for i in pokupki:
                if i[1][3:5] in ['01', '02', '12']:
                    arr.append(i)
                    print(str(count)+' ', i[0], i[1], i[2])
                    count+=1
            return cost_sort(arr)

        elif ch3 == '2':
            arr = []
            count = 1
            for i in pokupki:
                if i[1][3:5] in ['03', '04', '05']:
                    arr.append(i)
                    print(str(count)+' ', i[0], i[1], i[2])
                    count+=1
            return cost_sort(arr)

        elif ch3 == '3':
            arr = []
            count = 1
            for i in pokupki:
                if i[1][3:5] in ['06', '07', '08']:
                    arr.append(i)
                    print(str(count)+' ', i[0], i[1], i[2])
                    count+=1
            return cost_sort(arr)

        elif ch3 == '4':
            arr = []
            count = 1
            for i in pokupki:
                if i[1][3:5] in ['09', '10', '11']:
                    arr.append(i)
                    print(str(count)+' ', i[0], i[1], i[2])
                    count+=1
            return cost_sort(arr)

        elif ch3 == '5':
            return data()

```

Рисунок 16 – Код для функции data

Если пользователь хочет посмотреть все покупки, в этом ему поможет функция all\_pokupki, которая выводит товары с датой их покупки и стоимостью (см. Рисунок 17, Рисунок 18).

```

3
1) Помидоры 18.10.2022 90
2) Гречка 19.11.2022 50
3) Энергетик 22.06.2022 210
4) Бананы 30.07.2022 75
5) Манная крупа 03.01.2022 40
6) Вода 12.12.2022 25
7) Лимонад 14.02.2022 43
8) Картофель 25.04.2022 30
9) Яблоки 01.05.2022 37
10) Минеральная вода 16.08.2022 29
11) Молоко 03.03.2022 92
12) Рис 01.10.2022 36
13) Колбаса докторская 19.08.2022 143
14) Сосиски 17.10.2022 105
15) Колбаса копченая 24.12.2022 150
16) Сардельки 07.01.2022 111
17) Абрикосы 11.12.2022 103
18) Чай Липтон 03.04.2022 140

1 - Вернуться к остальным покупкам
2 - Сортировать по возрастанию цены
3 - Сортировать по убыванию цены

```

Рисунок 17 – Просмотр всех покупок

```

def all_pokupki():
    global pokupki
    count = 1
    for i in pokupki:
        print(str(count)+' ', i[0], i[1], i[2])
        count+=1
    return cost_sort(pokupki)

```

Рисунок 18 – Код функции all\_pokupki

Покупки можно сортировать по цене: как по возрастанию, так и по убыванию (см. Рисунок 19). Для этого существуют функции cost\_sort и cost (см. Рисунок 20). Функция cost\_sort предлагает пользователю выбор, как сортировать. Результат ввода пользователя передается функции cost, которая и выполняет сортировку.

```

1 - Вернуться к остальным покупкам
2 - Сортировать по возрастанию цены
3 - Сортировать по убыванию цены

2
1) Картофель 25.04.2022 30
2) Яблоки 01.05.2022 37
3) Бананы 30.07.2022 75
4) Помидоры 18.10.2022 90
5) Абрикосы 11.12.2022 103

1 - Вернуться к остальным покупкам
2 - Сортировать по возрастанию цены
3 - Сортировать по убыванию цены

3
1) Абрикосы 11.12.2022 103
2) Помидоры 18.10.2022 90
3) Бананы 30.07.2022 75
4) Яблоки 01.05.2022 37
5) Картофель 25.04.2022 30

1 - Вернуться к остальным покупкам
2 - Сортировать по возрастанию цены
3 - Сортировать по убыванию цены

```

Рисунок 19 – Сортировка по стоимости

```

def cost_sort(arr):
    while True:
        print()
        print('1 - Вернуться к остальным покупкам')
        print('2 - Сортировать по возрастанию цены')
        print('3 - Сортировать по убыванию цены')
        print()
        ch4 = input()
        if ch4 == '1':
            return prosmotr_pokupki()
        elif ch4 == '2':
            cost(arr, 1)
        elif ch4 == '3':
            cost(arr, -1)
        else:
            print('Введено некорректное действие, повторите попытку')

def cost(arr, sign):
    count = 1
    if sign == 1:
        arr = sorted(arr, key = lambda x: x[2])
    else:
        arr = sorted(arr, key = lambda x: x[2], reverse = True)
    for i in arr:
        print(str(count)+' ' ,i[0],i[1],i[2])
        count+=1

```

Рисунок 20 – Код функций cost\_sort и cost

Последней значимой функцией является функция Завершения программы. Представлена в коде она как функция end. Все данные из списков продуктов и покупок записываются в файлы, и программа завершается (см. Рисунок 21, Рисунок 22).

```
Выберите, что хотите сделать:
1 - Просмотр коллекции товаров
2 - Просмотр ваших покупок
3 - Завершение программы

3
Завершение программы, до новых встреч!
>>>|
```

Рисунок 21 – Завершение программы

```
def end():
    f = open('pokupki.txt', 'w', encoding='utf-8')
    for element in покупки:
        for x in element:
            if x!=element[-1]:
                f.write(str(x)+'|')
            else:
                f.write(str(x))
        f.write('\n')
    f.close()

    f = open('products.txt', 'w', encoding='utf-8')
    for element in products:
        f.write(element)
        f.write('\n')
    f.close()
```

Рисунок 22 – Код функции end



## ЗАКЛЮЧЕНИЕ

В ходе работы все функции были реализованы правильно, программа представляет собой законченное приложение. Были выполнены все поставленные задачи. Трудностей в процессе выполнения не возникло. Планируется реализовать авторизацию с помощью логина и пароля.