****

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА И ГОСУДАРСТВЕННОЙ СЛУЖБЫ ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

КОЛЛЕДЖ МНОГОУРОВНЕВОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

ОТЧЕТ

По учебной практике УП.01.01 Разработка программных модулей

специальность 09.02.07 Информационные системы и программирование

Выполнили студенты:

Ковяров А. Н.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

Садовская А. Р.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

Гусятинер Л. Б.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(оценка)

Москва, 2023

**Содержание**

[**Раздел 1. Проектирование консольного приложения** 1](#_Toc153971401)

[1.1 Общая информация 1](#_Toc153971402)

[1.2 Изучение входной и выходной документации 3](#_Toc153971403)

[1.3 Разработка требований к проекту. 5](#_Toc153971404)

[1.4 Разработка сценария проекта 6](#_Toc153971405)

[1.5 Построение диаграммы классов 7](#_Toc153971406)

[**Раздел 2. Разработка консольного приложения** 8](#_Toc153971407)

[2.1 Разработка главного модуля 8](#_Toc153971408)

[2.2 Разработка входящих модулей 10](#_Toc153971409)

[2.3 Тестирование и отладка 13](#_Toc153971410)

[2.4 Дневник 14](#_Toc153971411)

[**Использованные источники** 16](#_Toc153971412)

[**Приложение 1. Листинг класса Order** 17](#_Toc153971413)

[**Приложение 2. Листинг класса End Invoice** 17](#_Toc153971414)

[**Приложение 3. Листинг класса Delivery** 17](#_Toc153971415)

[**Приложение 4. Листинг класса Check** 18](#_Toc153971416)

**Раздел 1. Проектирование консольного приложения**

## 1.1 Общая информация

Проект представляет собой консольное приложение на языке Python.

***Описание бизнес-процесса***

«Отправка груза»

Заказчик вносит имя, фамилию, адрес и груз, данные записываются в накладную заказа, из накладной заказа данные заносятся в накладную доставки, из накладной доставки рассчитывается стоимость заказа и доставки. Заказчик оплачивает груз, доставка забирает груз и развозит получателям, заказчик получает груз и затем чек об оплате, доставка в накладной отмечается полученной.

***Структура базы данных***

Таблица 1. "Order":

-id

-Customer's name

-Address

-Issue date

-Cargo

-Cargo quantity

-Cargo cost

Таблица 2. "Delivery":

-id

-id order

-delivery date

-courier name

-order status

-Price

Таблица 3. "End invoice":

-id

-Customer's name

-Address

-Issue date

-Cargo

-Cargo quantity

-Cost

-Order status

Таблица 4. "Check":

-Id order

-Customer's name

-Address

-Cargo

-Price

***Запросы к базе данных:***

1. Получить данные о клиенте и его заказах

2. Получение данных какие заказы оплачены

В данном проекте мы работали с файлами и консолью, автоматизировали процесс документации.

## 1.2 Изучение входной и выходной документации

*Запрос: Получить данные о клиенте и его заказах*

Входной файл 1

«Заказ»

Id Фио клиента адрес Дата создания заказа Груз Количество груза Стоимость груза

1; Е.Д Томин; ул.преображенского; 20.09.2023; кофемашина; 1 3200 руб

2; Г.В Зайцев; ул.ватутитна; 20.09.2023; зимняя резина; 4 9000 руб

3; А.Ф Решетников; ул.достоевского; 29.09.2023; носки; 8 350 руб

Входной файл 2.

«Доставка»

Id id заказа дата получения фио курьера цена доставки Статус заказа

1; 1; 21.09.2023; А.А Киров; 400 руб; оплачено

2; 2; 21.09.2023; И.А мечников; 400 руб; не оплачено

3; 3; 30.09.2023; В.В Ростиков; 400 руб; оплачено

Выходной файл 1.

«Накладная»

id фио клиента адрес дата доставки груз количество груза стоимость статус

1. Е.Д Томин. ул.преображенского. 21.09.2023. 1 3600 руб. оплачено

2.Г.В Зайцев. ул.ватутитна. 21.09.2023. 4 36400 руб. не оплачено

3. А.Ф Решетников. ул.достоевского. 30.09.2023. 8 3200 руб. оплачено

*Запрос: Получение данных какие заказы оплачены*

Входной файл 1

«Заказ»

Id Фио клиента адрес Дата создания заказа Груз Количество груза Стоимость груза

1; Е.Д Томин; ул.преображенского; 20.09.2023; кофемашина; 1 3200 руб

2; Г.В Зайцев; ул.ватутитна; 20.09.2023; зимняя резина; 4 9000 руб

3; А.Ф Решетников; ул.достоевского; 29.09.2023; носки; 8 350 руб

Входной файл 2.

«Доставка»

Id id заказа дата получения фио курьера цена доставки Статус заказа

1; 1; 21.09.2023; А.А Киров; 400 руб; оплачено

2; 2; 21.09.2023; И.А мечников; 400 руб; не оплачено

3; 3; 30.09.2023; В.В Ростиков; 400 руб; оплачено

Выходной файл

«Оплачено»

1. Е.Д Томин. ул.преображенского. 21.09.2023. 1. 3600 руб. оплачено

3. А.Ф Решетников. ул.достоевского. 30.09.2023. 8. 3200 руб. оплачено

Выходной файл 3.

Неоплачено.txt

2. Г.В Зайцев. ул.ватутитна. 21.09.2023. 4. 36400 руб. не оплачено

## 1.3 Разработка требований к проекту.

Построение диаграммы использования.

В таблицах «Заказ», «Доставка», «Накладная» не должно быть двух записей с одинаковым идентификатором.

Поля таблиц должны корректно ссылаться на другие.

Диаграмма использования приведена на Рисунке 1.

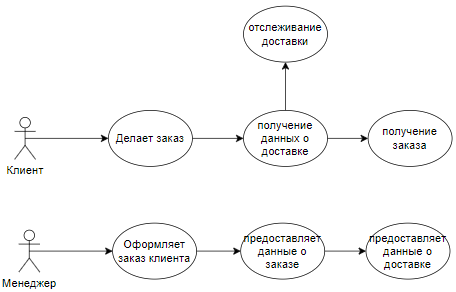


Рисунок 1. Диаграмма использования

## 1.4 Разработка сценария проекта

В данном разделе приведен сценарий использования программы пользователем (Рисунок 2).

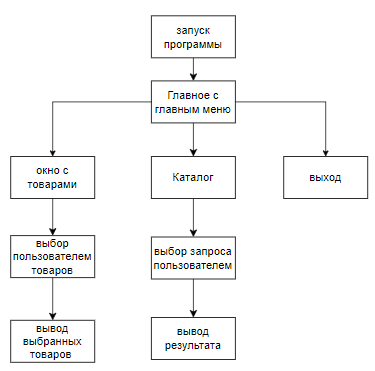


Рисунок 2. Сценарий проекта

## 1.5 Построение диаграммы классов

В данном разделе находится диаграмма классов приложения.(Рисунок 3)

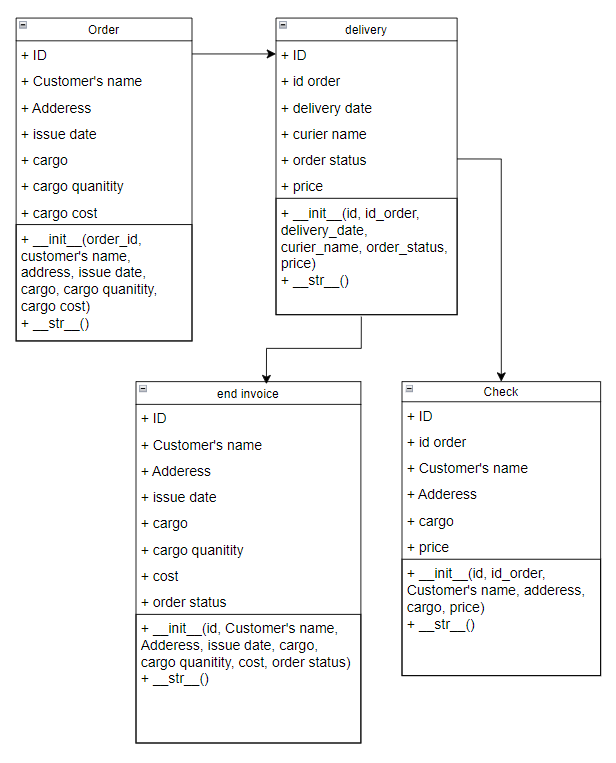


Рисунок 3. Диаграмма классов

**Раздел 2. Разработка консольного приложения**

## 2.1 Разработка главного модуля

Главный модуль состоит из класса Process. Он выполняется при запуске программы

Листинг 1. Главный модуль Process

from check import Check

from order import Order

from delivery import Delivery

from end\_invoice import End\_Invoice

# Определение класса 'Process' для управления списком различных объектов логистики

class Process:

def \_\_init\_\_(self):

# Инициализация пустого списка для хранения объектов логистики

self.lst = []

def \_\_str\_\_(self):

return f"{self.lst}"

# Метод для добавления объекта логистики в список

def add(self, thing):

self.lst.append(thing)

# Создание экземпляра класса Process для хранения объектов логистики

logistic\_process = Process()

# Чтение данных из файла "order.txt" и создание объектов Order

with open("order.txt", "r", encoding="utf-8") as forder:

# Итерация по каждой строке в файле

for line in forder:

# Извлечение данных из строки и преобразование их в соответствующие типы данных

id, customer\_name, address, issue\_date, cargo, cargo\_quantity, cargo\_cost = line.split(';')

id = int(id)

cargo\_quantity = int(cargo\_quantity)

# Создание объекта Order и добавление его в список logistic\_process

logistic\_process.add(Order(id, customer\_name, address, issue\_date, cargo, cargo\_quantity, cargo\_cost))

# Чтение данных из файла "delivery.txt" и создание объектов Delivery

with open("delivery.txt", "r", encoding="utf-8") as fdelivery:

# Итерация по каждой строке в файле

for line in fdelivery:

# Извлечение данных из строки и преобразование их в соответствующие типы данных

id, id\_order, delivery\_date, courier\_name, price, order\_status = line.split(';')

id = int(id)

id\_order = int(id\_order)

price = int(price)

# Создание объекта Delivery и добавление его в список logistic\_process

logistic\_process.add(Delivery(id, id\_order, delivery\_date, courier\_name, price, order\_status))

# Итерация по списку объектов логистики и вывод их строковых представлений

for thing in logistic\_process.lst:

print(str(thing), end='')

## 2.2 Разработка входящих модулей

Всего разработано 4 модуля:

* Класс «Delivery»
* Класс «End\_Invoice»
* Класс «Order»
* Класс «Check»

Взаимодействие между модулями можно наглядно увидеть по диаграммам классов из раздела 1.5.

Описание модулей:

Класс «Order» – представляет собой модель заказа. Он содержит следующие атрибуты:

- id (int): идентификатор клиента.

- customer\_name (str): имя клиента.

- address (str): адрес клиента.

- issue\_date (str): дата создания заказа.

- cargo (int): груз.

- cargo\_quantity(int): количество груза.

- cargo\_cost(int): цена.

Класс «Delivery» – представляет модель доставки. Он имеет следующие атрибуты:

- id (int): идентификатор доставки.

- id\_order (int): идентификатор заказа.

- delivery\_date (str): дата достваки.

- courier\_name (str): имя курьера.

- price(int): цена доставки.

- order\_status(str): статус заказа.

Класс «End\_Invoice» – моделирует накладную. Он включает в себя следующие атрибуты:

- id (int): идентификатор накладной.

- customers\_name (str): имя заказчика.

- address (str): адрес клиента.

- cargo (str): груз.

- cargo\_quantity (float): колличество груза.

- cost (float): сумма заказа.

- order\_status (float): статус заказа.

Класс «Check» – моделирует накладную. Он включает в себя следующие атрибуты:

- id\_order (int): идентификатор заказа.

- customers\_name (str): имя заказчика.

- address (str): адрес клиента.

- cargo (str): груз.

- price (int): общая цена.

Рассмотрим функцию для получения информации о менеджере клиента на основе идентификатора клиента и её блок-схему:

Листинг 2. Функция для получения информации о заказе:

# Чтение данных из файла "order.txt" и создание объектов Order

with open("order.txt", "r", encoding="utf-8") as forder:

# Итерация по каждой строке в файле

for line in forder:

# Извлечение данных из строки и преобразование их в соответствующие типы данных

id, customer\_name, address, issue\_date, cargo, cargo\_quantity, cargo\_cost = line.split(';')

id = int(id)

cargo\_quantity = int(cargo\_quantity)

# Создание объекта Order и добавление его в список logistic\_process

logistic\_process.add(Order(id, customer\_name, address, issue\_date, cargo, cargo\_quantity, cargo\_cost))

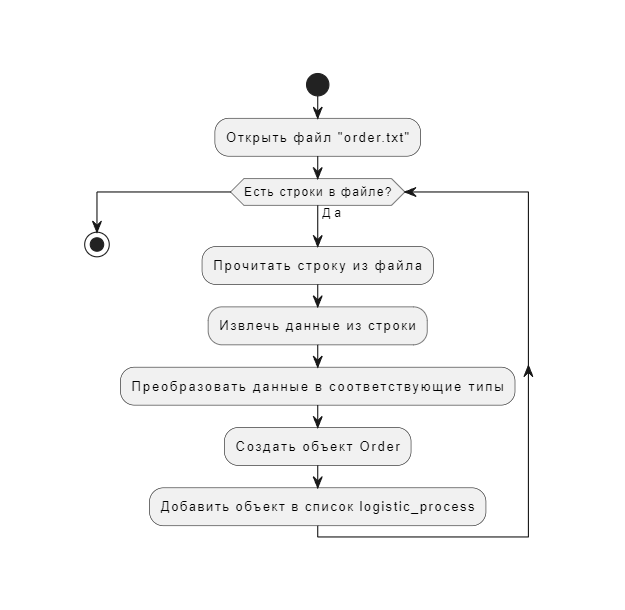


Рисунок 4. Блок-схема функции «logistic process»

## 2.3 Тестирование и отладка

В ходе написания проекта были протестированы входные данные

Входной файл orders.txt

1; Е.Д Томин; ул.преображенского; 20.09.2023; кофемашина; 1; 3200

2; Г.В Зайцев; ул.ватутитна; 20.09.2023; зимняя резина; 4; 9000

3; А.Ф Решетников; ул.достоевского; 29.09.2023; носки; 8; 350

Выходной файл check\_invoice.txt

1. 1. 21.09.2023. А.А Киров. 400. оплачено

2. 2. 21.09.2023. И.А мечников. 400. не оплачено

3. 3. 30.09.2023. В.В Ростиков. 400. Оплачено

При проверке кода были исправлены найденные ошибки, в результате при запуске программы ошибок не было (Рисунок 6):

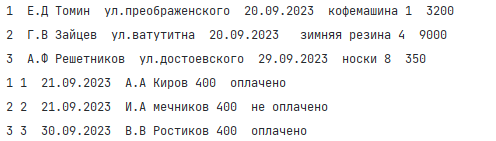


Рисунок 6. Успешная сборка

## 2.4 Дневник

|  |  |  |
| --- | --- | --- |
| Дата | Содержание работ | Отметка о выполнении |
| 01.09 | Вводный инструктаж. Формирование алгоритма решения вычислительной задачи. Формирование алгоритма решения задачи по обработке данных |  |
| 04.09 | Изучение средств визуализации алгоритмов |  |
| 09.09 | Оформление математического алгоритма при помощи программных средств |  |
| 11.09 | Разработка простой спецификации учебной задачи: Введение. Общее описание |  |
| 16.09 | Разработка простой спецификации учебной задачи: Функции системы |  |
| 22.09 | Разработка простой спецификации учебной задачи: Требования к данным. Требования к внешним интерфейсам. Атрибуты качества |  |
| 29.09 | Оформление спецификации программного продукта. Оформление введения и раздела «Общее описание». Оформление раздела «Функции системы». Оформление раздела «Требования к внешним данным» |  |
| 04.10 | Оформление раздела «Требования к внешним интерфейсам». Оформление раздела «Атрибуты качества» |  |
| 11.10 | Разработка классов главного модуля. Разработка методов классов и функций главного модуля |  |
| 18.10 | Разработка классов подсистемы вычислений. Разработка классов подсистемы связи с внешними данными |  |
| 25.10 | Разработка классов подсистемы отчетов. Разработать тестовые наборы |  |
| 01.11 | Провести тестирование главного модуля. Провести тестирование модулей |  |
| 08.11 | Провести тестирование программного продукта. По результатам тестирования провести рефакторинг программного продукта |  |
| 20.12 | Промежуточная аттестация: дифференцированный зачет |  |

## Использованные источники

1. Сайт языка Python  
   [www.xxx.yyy](http://www.xxx.yyy)
2. Сайт среды разработки PyCharm

<https://www.jetbrains.com/ru-ru/pycharm/>

1. Моделирование бизнес-процессов

https://www.businessstudio.ru/products/business\_studio/notations/

**Приложение 1. Листинг класса Order**

class Order:  
 def \_\_init\_\_(self, id, customer\_name, address, issue\_date, cargo, cargo\_quantity, cargo\_cost):  
 self.id = id  
 self.customer\_name = customer\_name  
 self.address = address  
 self.issue\_date = issue\_date  
 self.cargo = cargo  
 self.cargo\_quantity = cargo\_quantity  
 self.cargo\_cost = cargo\_cost  
  
 def \_\_str\_\_(self):  
 return f"{self.id} {self.customer\_name} {self.address} {self.issue\_date} {self.cargo} {self.cargo\_quantity} {self.cargo\_cost}"

**Приложение 2. Листинг класса End Invoice**

class End\_Invoice:  
 def \_\_init\_\_(self, id, customers\_name, address, issue\_date, cargo, cargo\_quantity, cost, order\_status):  
 self.id = id  
 self.customers\_name = customers\_name  
 self.address = address  
 self.issue\_date = issue\_date  
 self.cargo = cargo  
 self.cargo\_quantity = cargo\_quantity  
 self.cost = cost  
 self.order\_status = order\_status

**Приложение 3. Листинг класса Delivery**

class Delivery:  
 def \_\_init\_\_(self, id, id\_order, delivery\_date, courier\_name, price, order\_status):  
 self.id = id  
 self.id\_order = id\_order  
 self.delivery\_date = delivery\_date  
 self.courier\_name = courier\_name  
 self.price = price  
 self.order\_status = order\_status  
  
  
 def \_\_str\_\_(self):  
 return f"{self.id} {self.id\_order} {self.delivery\_date} {self.courier\_name} {self.price} {self.order\_status}"

**Приложение 4. Листинг класса Check**

class Check:  
 def \_\_init\_\_(self,id\_order, customers\_name, address, cargo, price):  
 self.id\_order = id\_order  
 self.customers\_name = customers\_name  
 self.address = address  
 self.cargo = cargo  
 self.price = price

from check import Check  
from order import Order  
from delivery import Delivery  
from end\_invoice import End\_Invoice  
  
# Определение класса 'Process' для управления списком различных объектов логистики  
class Process:  
 def \_\_init\_\_(self):  
 # Инициализация пустого списка для хранения объектов логистики  
 self.lst = []  
  
 def \_\_str\_\_(self):  
 return f"{self.lst}"  
  
 # Метод для добавления объекта логистики в список  
 def add(self, thing):  
 self.lst.append(thing)  
  
# Создание экземпляра класса Process для хранения объектов логистики  
logistic\_process = Process()  
  
# Чтение данных из файла "order.txt" и создание объектов Order  
with open("order.txt", "r", encoding="utf-8") as forder:  
 # Итерация по каждой строке в файле  
 for line in forder:  
 # Извлечение данных из строки и преобразование их в соответствующие типы данных  
 id, customer\_name, address, issue\_date, cargo, cargo\_quantity, cargo\_cost = line.split(';')  
 id = int(id)  
 cargo\_quantity = int(cargo\_quantity)  
  
 # Создание объекта Order и добавление его в список logistic\_process  
 logistic\_process.add(Order(id, customer\_name, address, issue\_date, cargo, cargo\_quantity, cargo\_cost))  
  
# Чтение данных из файла "delivery.txt" и создание объектов Delivery  
with open("delivery.txt", "r", encoding="utf-8") as fdelivery:  
 # Итерация по каждой строке в файле  
 for line in fdelivery:  
 # Извлечение данных из строки и преобразование их в соответствующие типы данных  
 id, id\_order, delivery\_date, courier\_name, price, order\_status = line.split(';')  
 id = int(id)  
 id\_order = int(id\_order)  
 price = int(price)  
  
 # Создание объекта Delivery и добавление его в список logistic\_process  
 logistic\_process.add(Delivery(id, id\_order, delivery\_date, courier\_name, price, order\_status))  
  
# Итерация по списку объектов логистики и вывод их строковых представлений  
for thing in logistic\_process.lst:  
 print(str(thing), end='')