**Оформление отчета.**

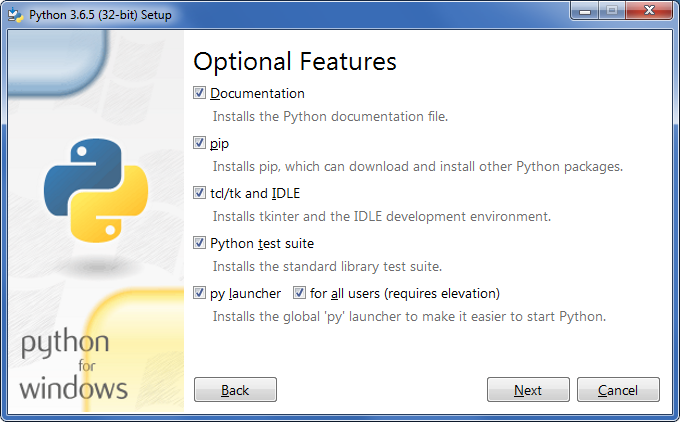
1. Разработка программных модулей в среде Python

**1.1 Подготовка и настройка среды. Начало работы.**

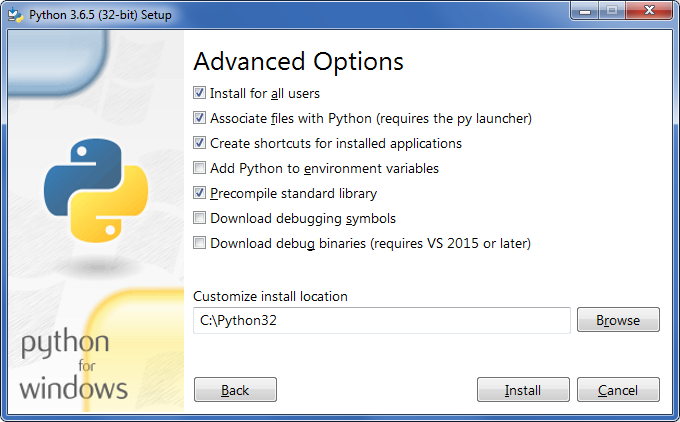
**1.1.1 Установки интерпретатора Python**

Все релизы для доступных операционных систем перечислены на [странице](https://www.python.org/downloads/?roistat_visit=12724225) официального сайта [python.org](https://www.python.org/). Последняя сборка Python на данный момент — 3.7.4. Неважно, какую конкретно версию Python вы скачаете, — главное, чтобы её номер начинался с цифры 3.

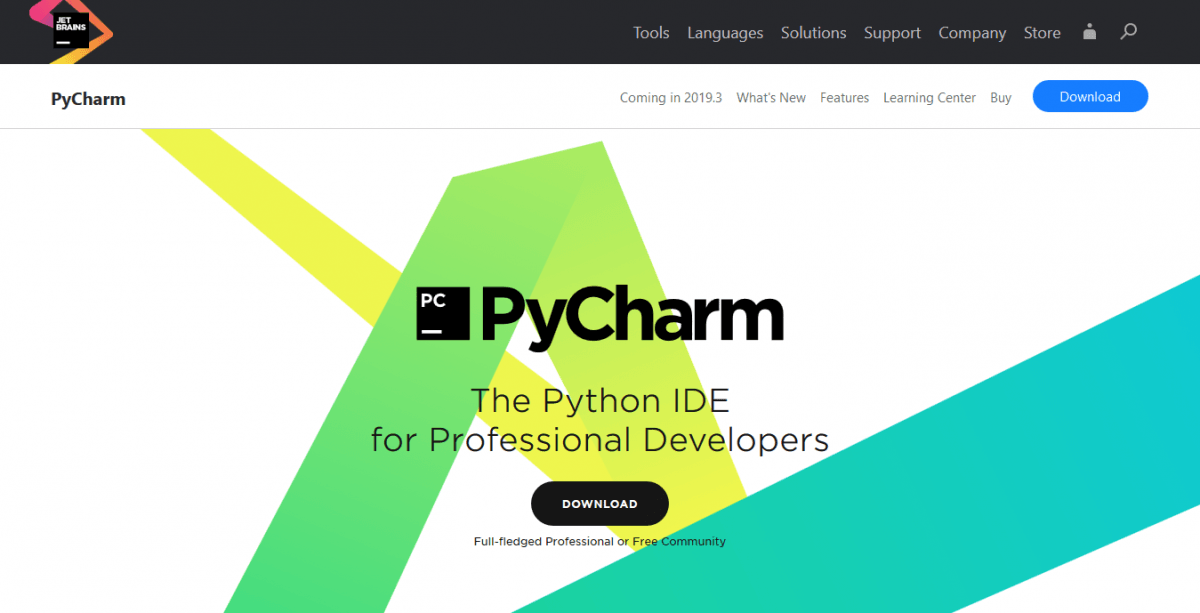
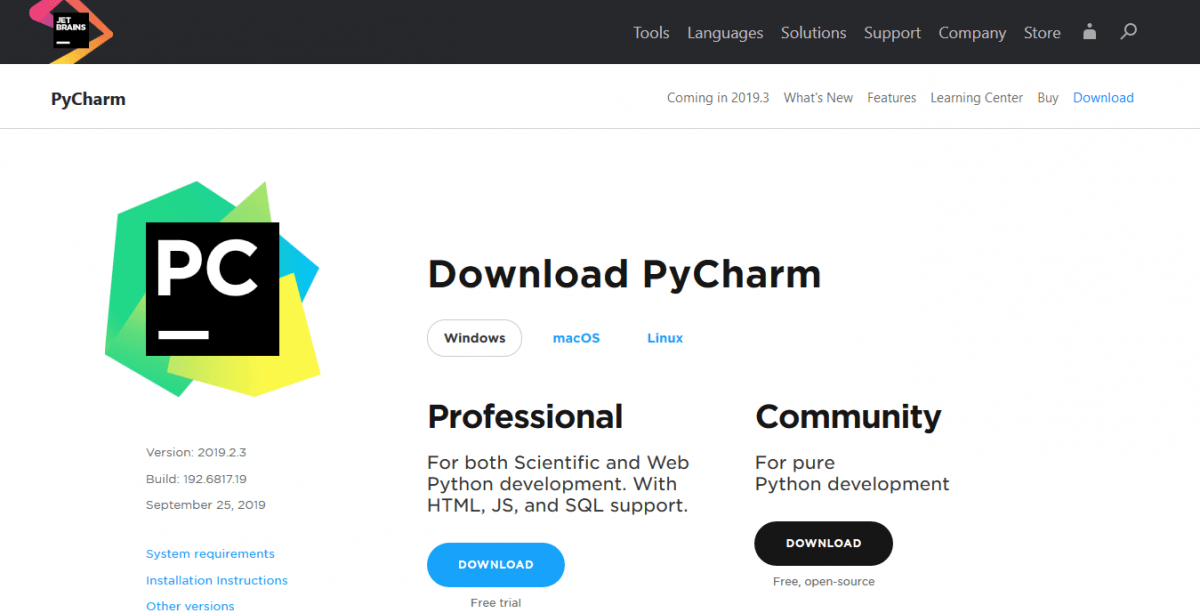
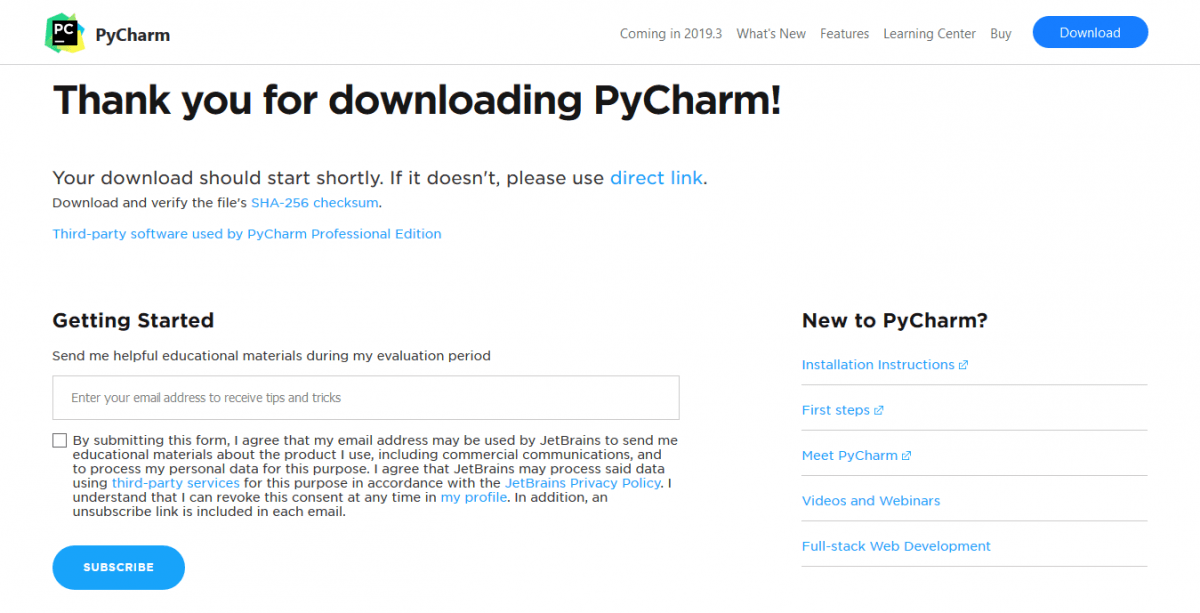
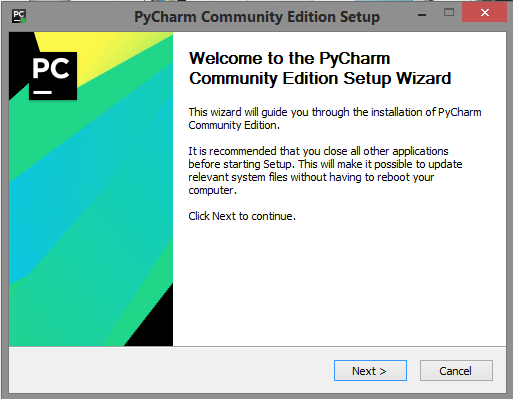
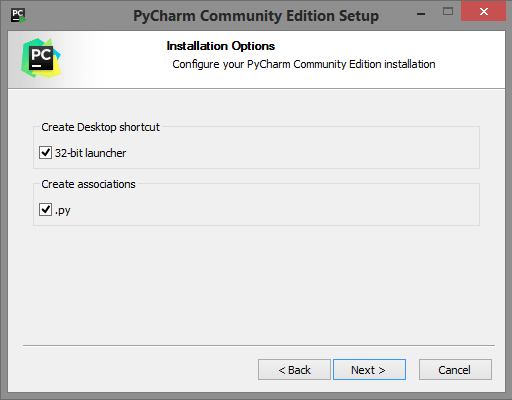
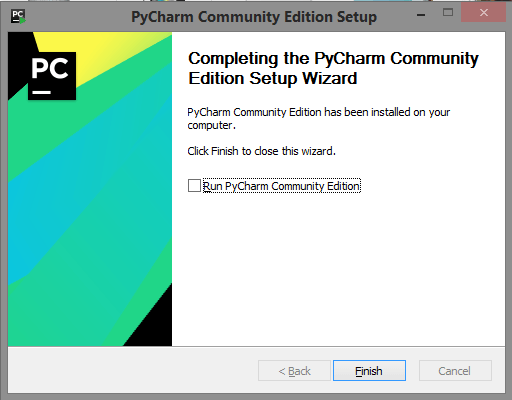
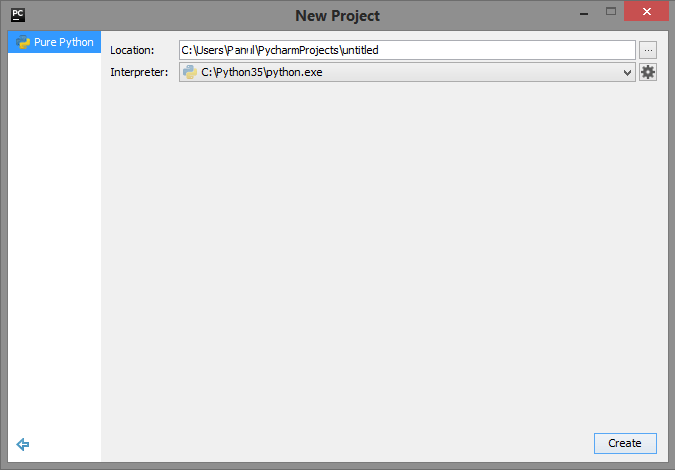
Далее запускаем скачанный файл. По умолчанию в комплекте устанавливаются **IDLE** *(среда разработки Python-программ)*, документация, менеджер пакетов **pip** *(для скачивания и установки других пакетов на Python)*, стандартный набор тестов и **Python Launcher** для запуска программ с расширением *.py*.



На следующем этапе выберите путь установки и укажите*«Установку для всех пользователей» (Install for all users).*



**1.1.2 Установки и настройка IDE**

1. Перейдите на сайт JetBrains [**https://www.jetbrains.com/pycharm/**](https://www.jetbrains.com/pycharm/). PyCharm — это проект JetBrains.  
   
2. Нажмите «Download». Произойдет перенаправление на страницу загрузки PyCharm.  
   
3. Дальше необходимо выбрать версию PyCharm: **Community** или **Professional**. Professional — это платная версия с полным набором функций. Она идеально подходит для профессиональной разработки. Community бесплатная. Ею можно пользоваться благодаря набору базовых возможностей. После нажатия на кнопку «Загрузить» скачивание начнется автоматически.  
   
4. Теперь необходимо запустить установку (~ pycharm-community-2019.2). Нажмите «Next».  
   
5. Оставьте папку установки по умолчанию. Это папка C:\Program files(x86)\JetBrains\PyCharm Community Edition 2019.2. Если выбрана другая, лучше указать именно это направление. Нажмите «Next».
6. Поставьте галочки напротив обоих пунктов, если нужен ярлык для рабочего стола. Нажмите «Next».  
   
7. Далее установщик попросит указать имя для отображения в стартовом меню. По умолчанию это JetBrains. Нажмите «Install».
8. Установка начнется автоматически. Дождитесь завершения.
9. После завершения установки поставьте галочку напротив «Run PyCharm Community Edition» для запуска, а затем «Finish».  
   
10. Если вы только учите язык Python и установили PyCharm впервые, тогда выберите пункт «I do not have a previous version of PyCharm or I do not want to import my settings» и «ОК».
11. PyCharm запустится. Первый раз этой займет больше времени.
12. После приветственного окна нажмите «Create New Project».
13. Выберите директорию для сохранения проекта. Во втором поле нужно указать местоположение интерпретатора Python (который уже должен быть установлен). Обычно PyCharm находит его самостоятельно. Если не удалось, нужно указать путь к нему и нажать «Create».  
    
14. Нажмите «File > New > Python File». Введите название нового файла. Он откроется с чистой консолью. Можно начинать писать программу для проверки. Напишите что-то простое: например, программу, которая выводит «Hello World!» на экран. Используйте этот код, чтобы проверить работу PyCharm.
15. print("Hello World!")

В любом месте кликните правой кнопкой мыши и выберите «Run». Программа выполнится, а вывод будет показан на экране.

**1.1.3 Особенности задания имен переменных и правила организации кода в Python**

Переменные в Python не требуют объявления типа переменной (так как Python – язык с динамической типизацией) и являются ссылками на область памяти. Правила именования переменных:

имя переменной может состоять только из букв, цифр и знака подчёркивания

имя не может начинаться с цифры

имя не может содержать специальных символов @, $, %

найдено на [pyneng.readthedocs.io](https://pyneng.readthedocs.io/ru/latest/book/03_start/2_variables.html)

Внешний вид кода¶

Используйте 4 пробела на один уровень отступа. Символы табуляции запрещены

Ограничьте максимальную длину строки 90 символами

В случае если строка больше 90 символов, осуществите перенос на новую строку с двойным отступом относительно текущего отступа

Отделяйте функции верхнего уровня и определения классов двумя пустыми строками. Определения методов внутри класса отделяйте одной пустой строкой

Не рекомендуется размещение нескольких операторов в одной строке, кроме случаев, когда это не ухудшает удобочитаемости программы

При импорте имён из модулей с помощью инструции from запрещается использовать форму со звёздочкой

**1.2 Библиотека для анализа данных Pandas.**

**1.2.1 Описание библиотеки Pandas.**

Pandas – это библиотека с открытым исходным кодом на Python. Она предоставляет готовые к использованию высокопроизводительные структуры данных и инструменты анализа данных.

1. Модуль Pandas работает поверх NumPy и широко используется для обработки и анализа данных.
2. NumPy – это низкоуровневая структура данных, которая поддерживает многомерные массивы и широкий спектр математических операций с массивами. Pandas имеет интерфейс более высокого уровня. Он также обеспечивает оптимизированное согласование табличных данных и мощную функциональность временных рядов.
3. DataFrame – это ключевая структура данных в Pandas. Это позволяет нам хранить и обрабатывать табличные данные, как двумерную структуру данных.
4. Pandas предоставляет богатый набор функций для DataFrame. Например, выравнивание данных, статистика данных, нарезка, группировка, объединение, объединение данных и т.д.

**1.2.2 Практическая часть.**

. Необходимо написать программу, которая будет хранить информацию о продажах в магазине. (например, такую как: дата продажи, товар, код товара, цена, количество, скидка на товар, ед. измерения).

Реализуйте следующий функционал:

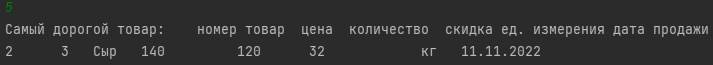
* Добавление новой записи
* Удаление записи по коду
* Вывести в алфавитном порядке товары
* Вывести все товары, проданные в указанную дату
* Вывести количество проданных товаров в указанную дату
* Вывести самый дорогой товар
* Вывести записи со скидкой более чем указанная
* Вывести кол-во товара указанной уд. изм
* Любая ваша функция.

Информация о 10 записях уже должны быть занесена в программу (из кода).

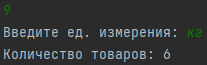
Функция открытия данных

****

Функция вывода самого договора товара

****

Функция вывода кол-во товара указанной в ед. изм

****

Программный код скрипты вынести в приложения.

* 1. **Создание пользовательской модели классов**
     1. **Основы работы с пользовательскими классами в Python.**

Атрибут:

Атрибут — это элемент класса. Например, у прямоугольника таких 2: ширина (width) и высота (height).

Метод:

* Метод класса напоминает классическую функцию, но на самом деле — это функция класса. Для использования ее необходимо вызывать через объект.
* Первый параметр метода всегда self (ключевое слово, которое ссылается на сам класс).

Конструктор:

* Конструктор — уникальный [**метод класса**](https://pythonru.com/primery/primery-raboty-s-klassami-v-python), который называется \_\_init\_\_.
* Первый параметр конструктора во всех случаях self (ключевое слово, которое ссылается на сам класс).
* Конструктор нужен для создания объекта.
* Конструктор передает значения аргументов свойствам создаваемого объекта.
* В одном классе всегда только один конструктор.
* Если класс определяется не конструктором, Python предположит, что он наследует конструктор родительского класса.
  + 1. **Практическая часть.**

Необходимо написать программу, которая будет хранить информацию о продажах в магазине. (например, такую как: название, код товара, цена, кол-во, скидка на товар, ед. измерения).

Классы:

Товары – свойства описаны выше;

Касса: Товар, дата продажи, список товаров, сумма.

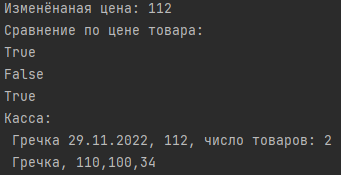
Методы:

Сравнение товаров по цене;

Пересчет суммы продажи товаров при изменении списка товаров;

Количество товаров в операции по кассе;

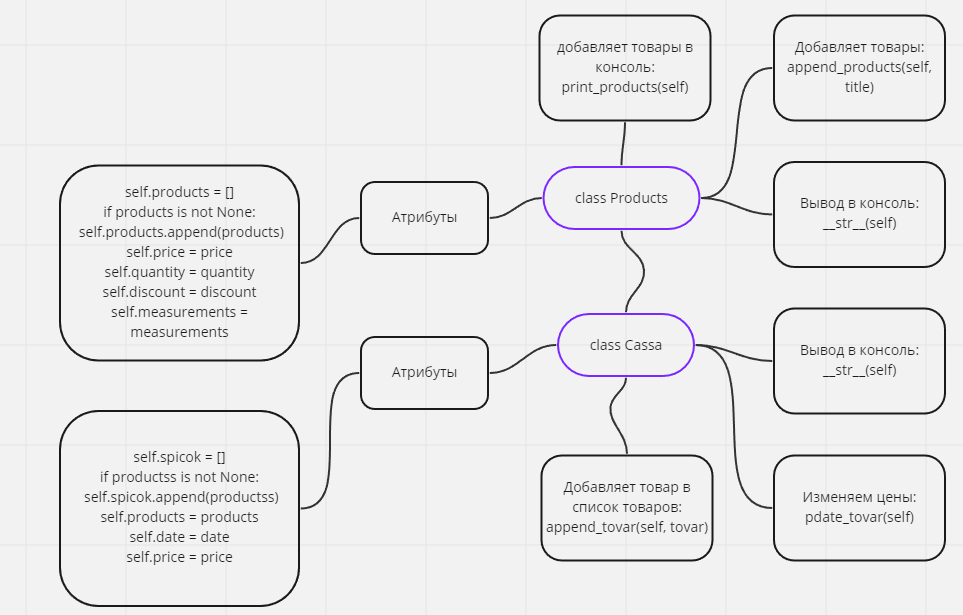
Дополните модель собственными классами, свойствами и методами при необходимости.

****

1 Изменение цены товара

2 Сравнение по цене товара

3 Проверка сколько в кассе товаров



Программный код скрипты вынести в приложения.

* 1. **Построение интерфейса приложения.**
     1. Библиотека PyQt5

PyQt — это библиотека, которая позволяет использовать фреймворк Qt GUI (GUI — это графический интерфейс пользователя) в Python.

from PyQt5 import QtWidgets, QtGui,QtCore

from mydesign import Ui\_MainWindow

import sys

class mywindow(QtWidgets.QMainWindow):

    def \_\_init\_\_(self):

        super(mywindow, self).\_\_init\_\_()

        self.ui = Ui\_MainWindow()

        self.ui.setupUi(self)

        self.ui.label.setFont(

            QtGui.QFont('SansSerif', 30)

        )

        self.ui.label.setGeometry(

            QtCore.QRect(10, 10, 200, 200)

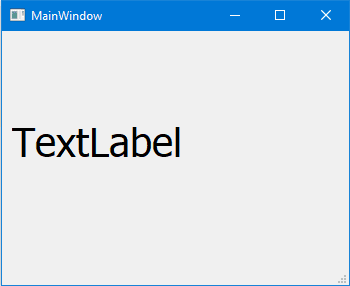
        ) # изменить геометрию ярлыка

app = QtWidgets.QApplication([])

application = mywindow()

application.show()

sys.exit(app.exec())

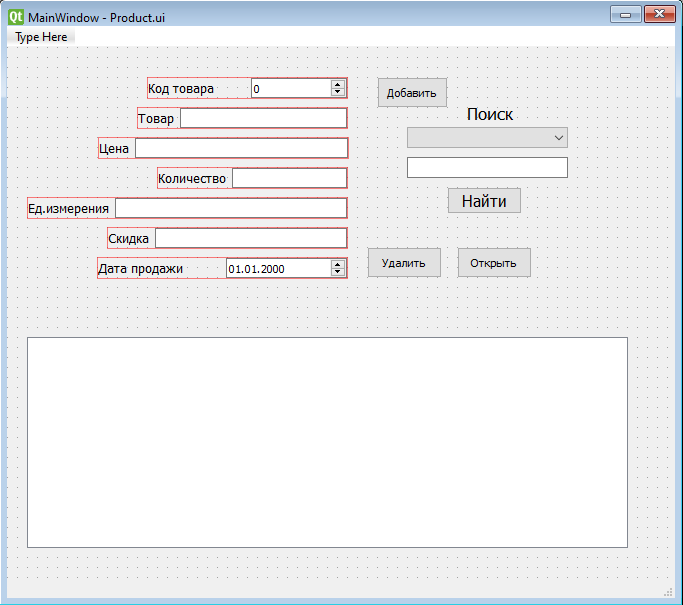


* + 1. Практическая часть

Необходимо написать программу, которая будет хранить информацию о продажах в магазине. (например, такую как: дата продажи, товар, код товара, цена, количество, скидка на товар, ед. измерения).

Реализуйте следующий функционал:

* Добавление новой записи
* Удаление записи по коду
* Вывести в алфавитном порядке товары
* Вывести все товары, проданные в указанную дату
* Вывести количество проданных товаров в указанную дату
* Вывести самый дорогой товар
* Вывести записи со скидкой более чем указанная
* Вывести кол-во товара указанной уд. изм
* Любая ваша функция.



from PyQt5.QtWidgets import QApplication, QMainWindow, QWidget, QTableWidgetItem  
from fenetre import Ui\_MainWindow  
  
  
class MyWidget(QMainWindow, Ui\_MainWindow):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.setupUi(self)  
 self.pbInsert.clicked.connect(self.insert\_staff)  
 self.pbFind.clicked.connect(self.find\_for\_val)  
 self.pbOpen.clicked.connect(self.open\_file)  
 self.pbDelete.clicked.connect(self.delete\_staff)  
 self.conn = None  
  
 def open\_file(self):  
 try:  
 self.conn = sqlite3.connect('staff\_db.db')  
 cur = self.conn.cursor()  
  
 data = cur.execute("select \* from staff")  
 col\_name = [i[0] for i in data.description]  
 data\_rows = data.fetchall()  
 except Exception as e:  
 print(f"Проблемы с подключением к БД. {e}")  
 return e  
 self.twStaffs.setColumnCount(len(col\_name))  
 self.twStaffs.setHorizontalHeaderLabels(col\_name)  
 self.twStaffs.setRowCount(0)  
 self.cbColNames.addItems(col\_name)  
 for i, row in enumerate(data\_rows):  
 self.twStaffs.setRowCount(self.twStaffs.rowCount() + 1)  
 for j, elem in enumerate(row):  
 self.twStaffs.setItem(i, j, QTableWidgetItem(str(elem)))  
 self.twStaffs.resizeColumnsToContents()

Программный код скрипты вынести в приложения.

**1.5 Написание локального веб-сервера при помощи модуль flask.**

1.5.1 Введение во flask.

 Flask – это микофреймворк для разработчиков, который позволяет им быстро и просто создавать и масштабировать веб-приложения.

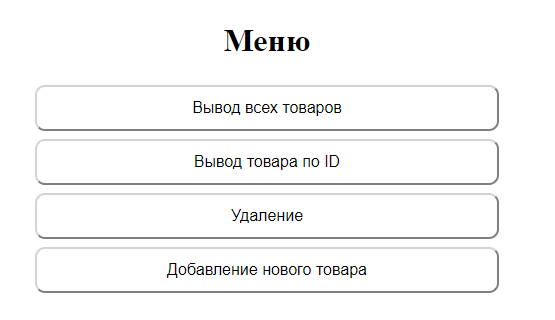
Flask Framework использует Jinja2 — приложение для обработки шаблонов — и Werkzeug — инструмент для работы с WSGI (стандартом взаимодействия между Python-программой, которая выполняется на стороне сервера, и самим веб-сервером). Для создания изолированной среды в Python используется модуль Virtualenv.

Веб-фреймворк Flask поддерживает:

* модульное тестирование — проверку на корректность отдельных модулей исходного кода программы;
* Google App Engine — мощную платформу для создания и запуска приложения в инфраструктуре Google;
* RESTfull-архитектуру — соответствующий набор архитектурных принципов для создания гибкой и масштабируемой сети.

Flask защищен от XSS-атак, когда злоумышленник внедряет на страницу вредоносный скрипт, и работает с безопасными cookie. В инструменте Werkzeug предусмотрен простой веб-сервер.

1.5.2 Написание веб-сервера для своего варианта



1. Выводит все товары



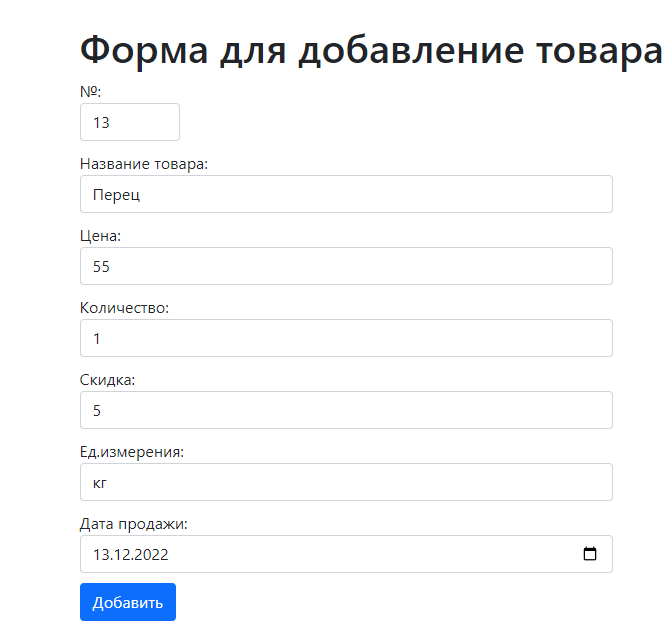
2. Вывод товара по ID



3. Удаление товара



4. Добавление нового товара



программный код

**1.6 Юнит тестирование Pytest**

1.6.1 Юнит-тестирование в Python.

Модульное тестирование, иногда блочное тестирование или юнит-тестирование — процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы, наборы из одного или более программных модулей вместе с соответствующими управляющими данными, процедурами использования и обработки.

Pytest - это платформа тестирования на основе python, которая используется для написания и выполнения тестовых кодов. В настоящее время сервисы REST в основном используются для тестирования API, хотя мы можем использовать pytest для написания простых и сложных тестов, т. Е. Мы можем писать коды для тестирования API, базы данных, пользовательского интерфейса и т. Д.

**def** **test\_asdict**():

"""\_asdict() должен возвращать словарь."""

t\_task = Task('do something', 'okken', True, 21)

t\_dict = t\_task.\_asdict()

expected = {'summary': 'do something',

'owner': 'okken',

'done': True,

'id': 21}

**assert** t\_dict == expected

1.6.2 Написание юнит-тестов для свой предметной области

# Проверка на отрицательную цену  
def test\_wrong\_price():  
 with pytest.raises(ValueError):  
 class5 = Cassa('Рис', '30.11.2022', -90)  
  
  
# Проверка на увелечение цены  
def test\_increase\_price():  
 class1 = Products(110, 'Гречка', 100, 34, 'кг')  
 class1.pricce += 100  
 assert class1.pricce == 210  
  
  
# Проверка при добавлении товара. (неверный тип)  
def test\_wrong\_type\_Fish():  
 with pytest.raises(TypeError):  
 class1 = Products('Гречка', 110, 100, 34, 'кг')  
 class4 = Cassa('Гречка', '29.11.2022', 110)  
 class4.append\_products("Гречка")  
  
  
# Сравнивае двух товаров по цене  
def test\_employees\_lt():  
 class2 = Products('Рис', 90, 110, 23, 'кг')  
 class3 = Products('Сыр', 140, 120, 32, 'кг')  
 assert (class2 < class3) == True  
  
  
# Проверка при добавлении товара (добавтли правильно)  
def test\_append\_Fish():  
 class3 = Products('Сыр', 140, 120, 32, 'кг')  
 class4 = Cassa('Гречка', '29.11.2022', 110, class3)  
 count\_before = len(class4.products)  
 class4.append\_tovar(class3)  
 assert count\_before == len(class4.products) - 1  
  
  
# Сравниваем двух товаров по цене  
def test\_employees\_lot():  
 class4 = Cassa('Гречка', '29.11.2022', 110)  
 class5 = Cassa('Рис', '30.11.2022', 90)  
 assert (class4 > class5) == False  
  
  
# Увелечиние цены  
def test\_increase\_volume():  
 class4 = Cassa('Гречка', '29.11.2022', 110)  
 class4.price += 50  
 assert class4.price == 160  
  
  
# увеличение количества  
def test\_increase\_volume1():  
 class1 = Products('Гречка', 110, 100, 34, 'кг')  
 class1.quantity += 2  
 assert class1.quantity == 112  
  
  
  
# Удаление товара  
def test\_delete\_fish():  
 class5 = Cassa('Рис', '30.11.2022', 90)  
 class1 = Products('Гречка', 110, 100, 34, 'кг')  
 class2 = Products('Рис', 90, 110, 23, 'кг')  
 class5.append\_tovar(class1)  
 class5.append\_tovar(class2)  
 with pytest.raises(TypeError):  
 class5.delete\_tovar("Гречка")

**Выводы по разделу.**

В результате выполнения практических заданий по разработке программных модулей на языке Python была проведена работа с такими модулями как: Pandas, … перечислить модули. В пункте 1.1 была описана среда разработки и особенности языка Python. В пункте 1.2 был создан скрипт по работе с csv файлами моего варианта. В … и т.д.