Отчет по лабораторной работе №10

Фильтрация изображений. Контуры на изображении

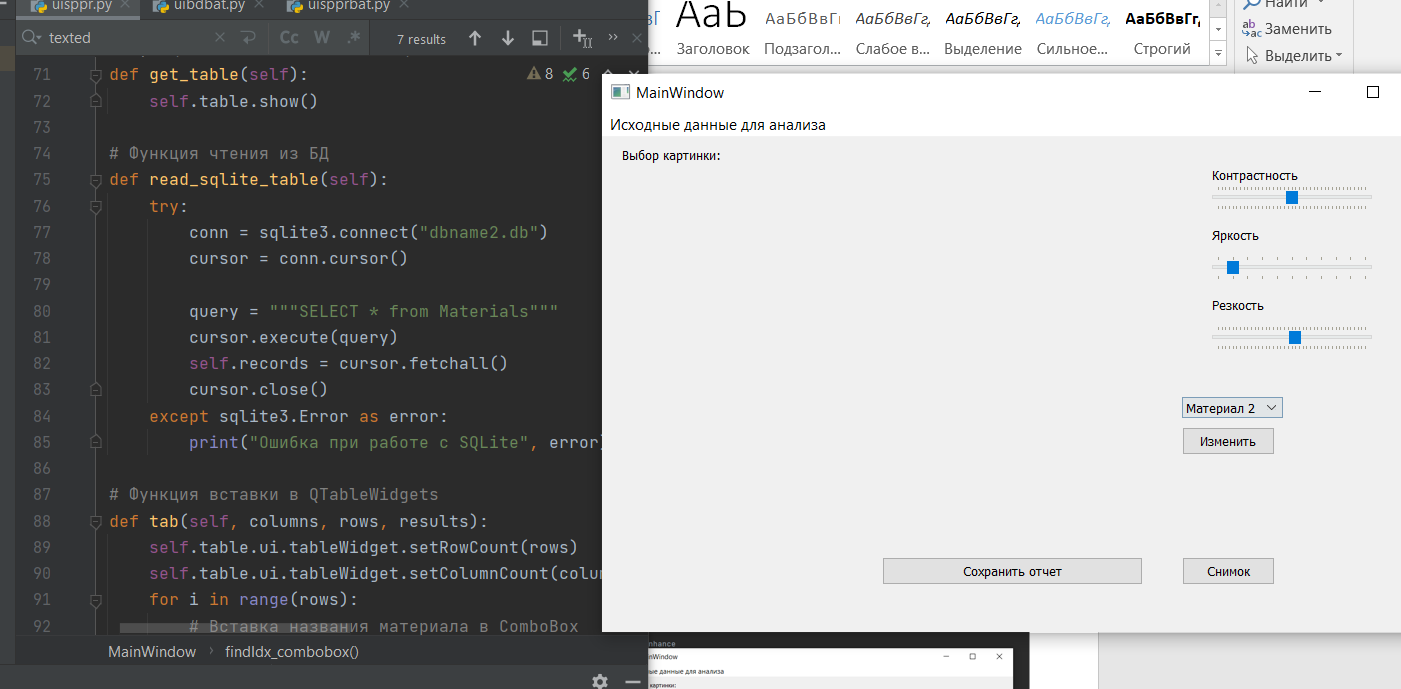
# Цель:

создание системы управления базой данных для СППР

«Определение качества поверхности пористого материала».

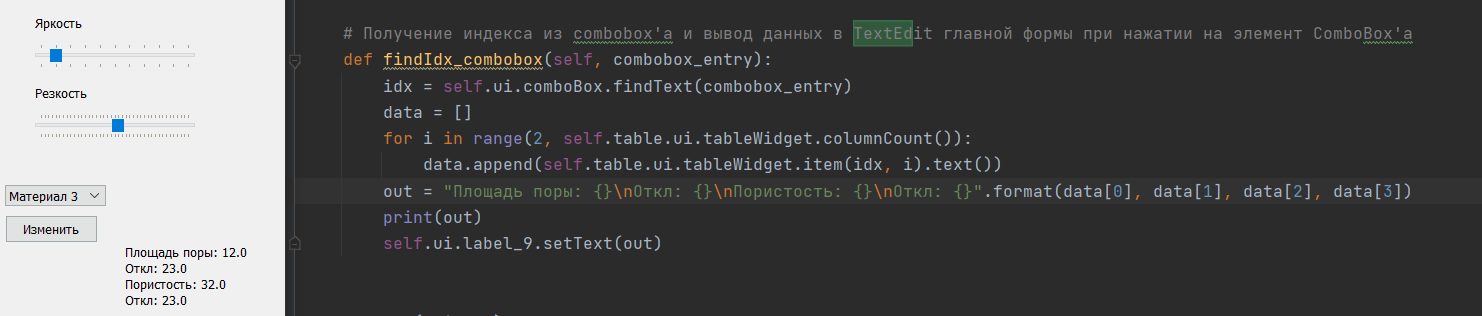
Задание 1.

Данные с таблицы синхронизированы с данными из бд(внесение данных в таблицу производится при запуске программы)



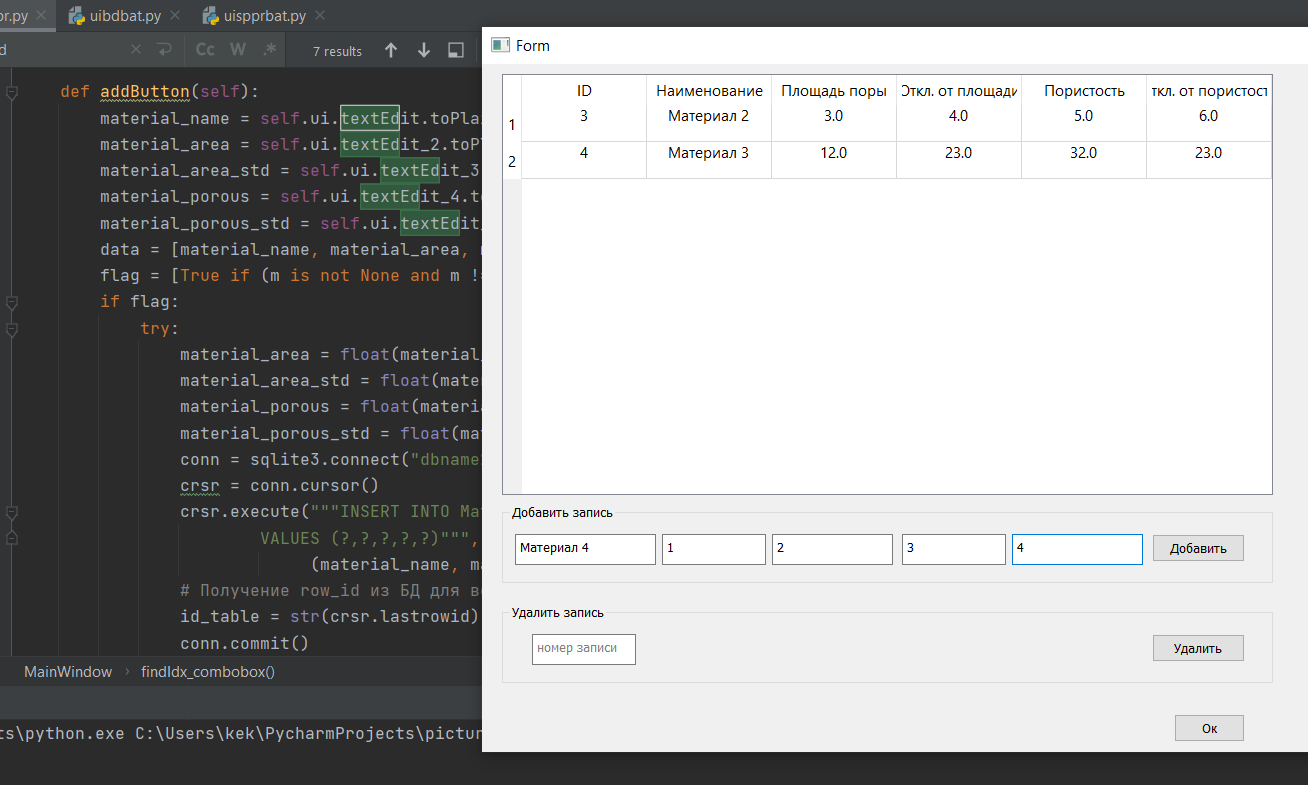
# Функция чтения из БД  
def read\_sqlite\_table(self):  
 try:  
 conn = sqlite3.connect("dbname2.db")  
 cursor = conn.cursor()  
  
 query = """SELECT \* from Materials"""  
 cursor.execute(query)  
 self.records = cursor.fetchall()  
 cursor.close()  
 except sqlite3.Error as error:  
 print("Ошибка при работе с SQLite", error)

Отображение данных справа внизу при выборе из списка:



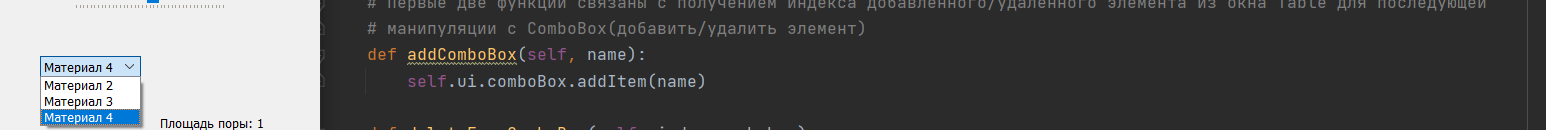
# Получение индекса из combobox'а и вывод данных в TextEdit главной формы при нажатии на элемент ComboBox'а  
def findIdx\_combobox(self, combobox\_entry):  
 idx = self.ui.comboBox.findText(combobox\_entry)  
 data = []  
 for i in range(2, self.table.ui.tableWidget.columnCount()):  
 data.append(self.table.ui.tableWidget.item(idx, i).text())  
 out = "Площадь поры: {}\nОткл: {}\nПористость: {}\nОткл: {}".format(data[0], data[1], data[2], data[3])  
 print(out)  
 self.ui.label\_9.setText(out)

Добавление нового элемента в таблицу:

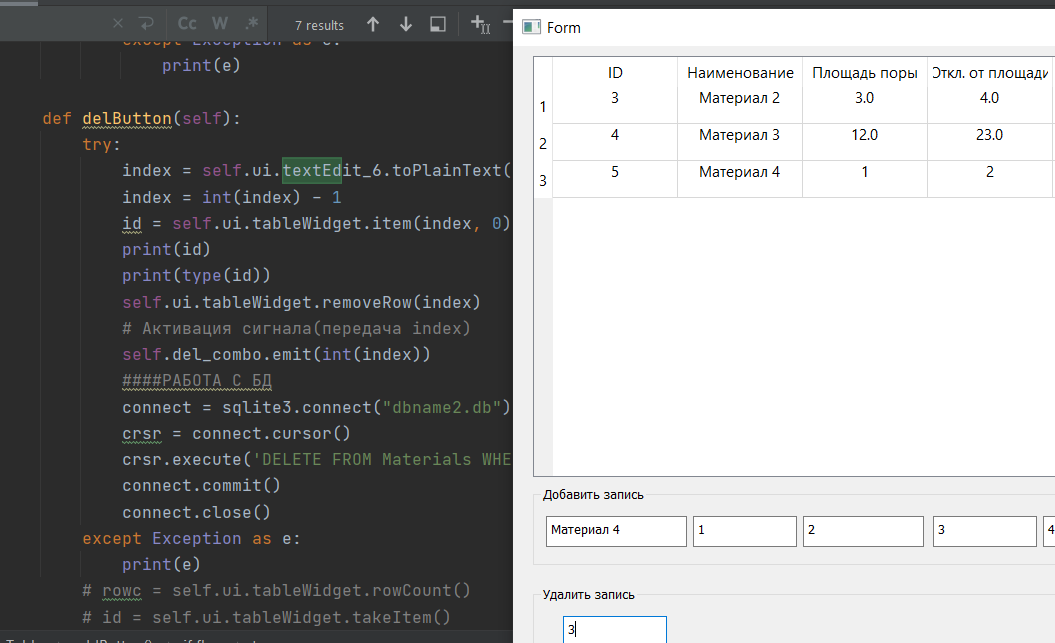


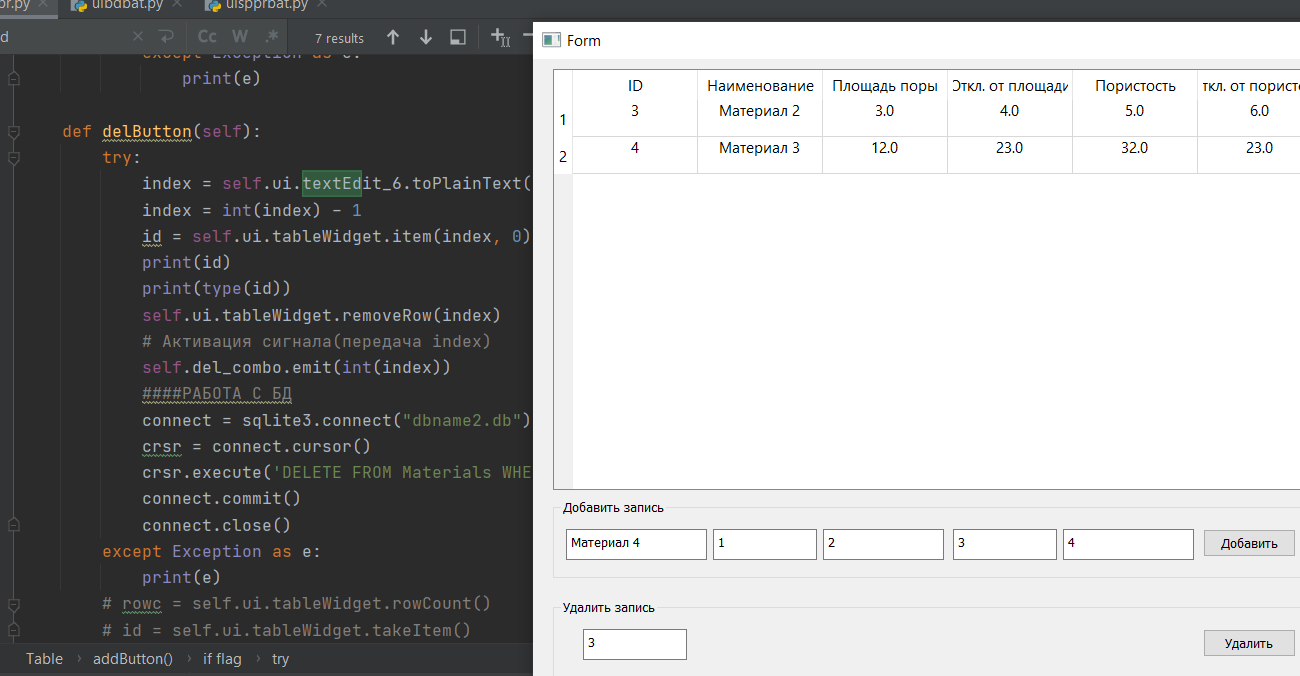
def addButton(self):  
 material\_name = self.ui.textEdit.toPlainText()  
 material\_area = self.ui.textEdit\_2.toPlainText()  
 material\_area\_std = self.ui.textEdit\_3.toPlainText()  
 material\_porous = self.ui.textEdit\_4.toPlainText()  
 material\_porous\_std = self.ui.textEdit\_5.toPlainText()  
 data = [material\_name, material\_area, material\_area\_std, material\_porous, material\_porous\_std]  
 flag = [True if (m is not None and m != '') else False for m in data]  
 if flag:  
 try:  
 material\_area = float(material\_area)  
 material\_area\_std = float(material\_area\_std)  
 material\_porous = float(material\_porous)  
 material\_porous\_std = float(material\_porous\_std)  
 conn = sqlite3.connect("dbname2.db")  
 crsr = conn.cursor()  
 crsr.execute("""INSERT INTO Materials(NAME, PORE\_AREA\_MEAN, PORE\_AREA\_STD, POROUS\_MEAN, POROUS\_STD)   
 VALUES (?,?,?,?,?)""",  
 (material\_name, material\_area, material\_area\_std, material\_porous, material\_porous\_std))  
 # Получение row\_id из БД для вставки в таблицу значения ID  
 id\_table = str(crsr.lastrowid)  
 conn.commit()  
 conn.close()  
  
 # Добавление новой строки в таблицу  
 rowPosition = self.ui.tableWidget.rowCount()  
 self.ui.tableWidget.insertRow(rowPosition)  
  
 # Вставка id в data  
 data = [id\_table] + data  
 print(data[0])  
 # Вставка в таблицу  
 for i in range(6):  
 item = QtWidgets.QTableWidgetItem()  
 item.setText(data[i])  
 item.setTextAlignment(Qt.AlignHCenter)  
 self.ui.tableWidget.setItem(rowPosition, i, item)  
  
 # Активация сигнала(передача material\_name)  
 self.add\_combo.emit(material\_name)  
 except Exception as e:  
 print(e)

Добавление материала в список:



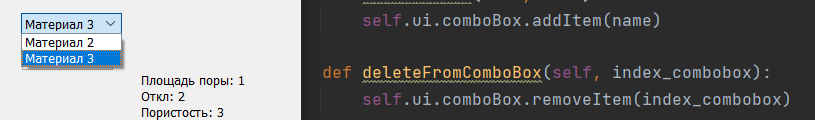
def addComboBox(self, name):  
 self.ui.comboBox.addItem(name)

Удаление записи из таблицы:



def delButton(self):  
 try:  
 index = self.ui.textEdit\_6.toPlainText()  
 index = int(index) - 1  
 id = self.ui.tableWidget.item(index, 0).text()  
 print(id)  
 print(type(id))  
 self.ui.tableWidget.removeRow(index)  
 # Активация сигнала(передача index)  
 self.del\_combo.emit(int(index))  
 ####РАБОТА С БД  
 connect = sqlite3.connect("dbname2.db")  
 crsr = connect.cursor()  
 crsr.execute('DELETE FROM Materials WHERE ID=?', (id,))  
 connect.commit()  
 connect.close()  
 except Exception as e:  
 print(e)

Удаление материала из списка:



def deleteFromComboBox(self, index\_combobox):  
 self.ui.comboBox.removeItem(index\_combobox)