МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

**«Московский технический университет связи и информатики» (МТУСИ)**

Кафедра «Системное программирование»

**ОТЧЕТ**

по лабораторной работе № 6 дисциплина **«Введение в информационные технологии»**

Выполнил: Студент гр.БВТ2207

Кузьмин И.А.

“ ” 2023г.

Проверил:

“ ” 2023г.

Москва, 2023

# Введение

# Цель работы:

# Задание:

Написать калькулятор на PyQt5

.

# Реализация

В процессе выполнения лабораторной работы использованы следующие информационные источники [1-2].

**Практическая реализация**



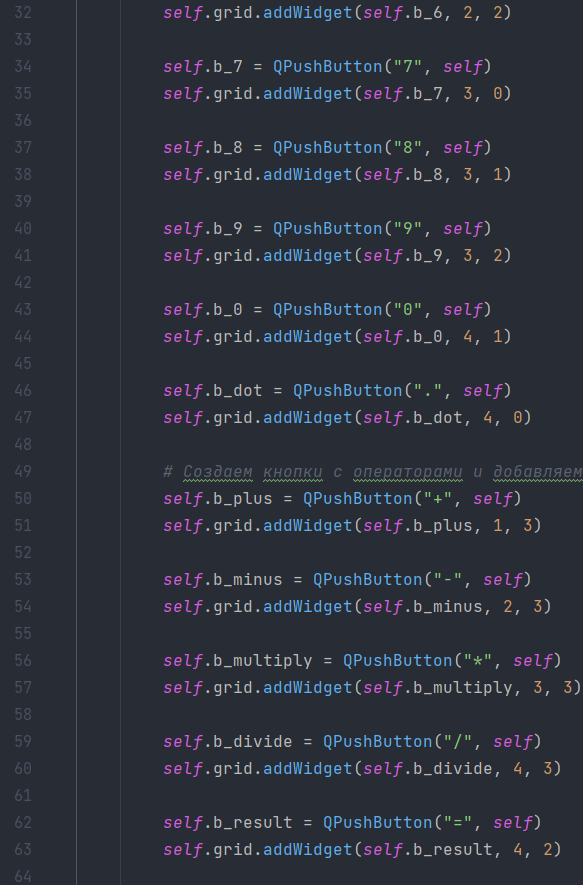
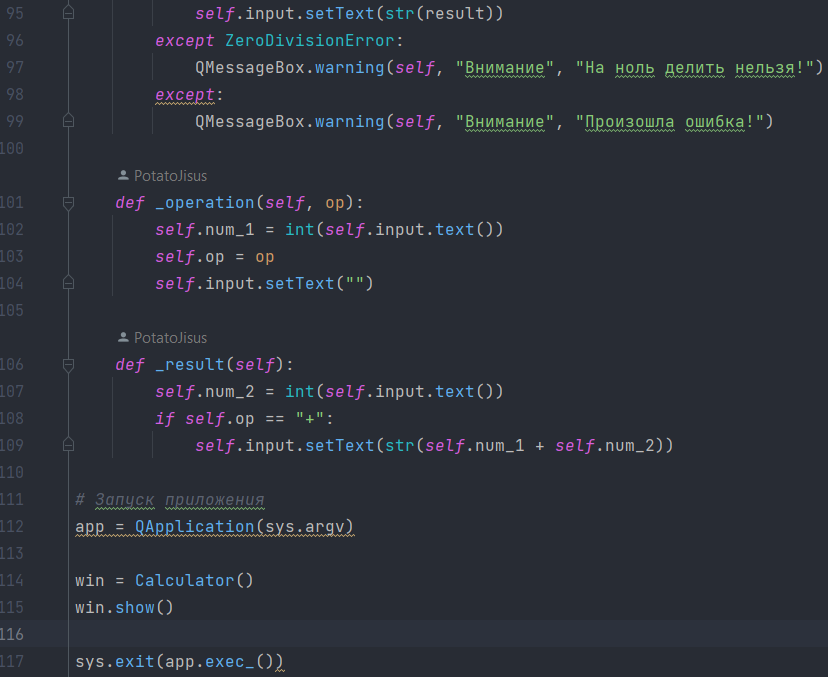
  

Рис. 1 – код калькулятора

Этот код представляет собой простой калькулятор с графическим интерфейсом, разработанным с использованием библиотеки PyQt5. Давайте рассмотрим его по частям:

1. Импорт необходимых модулей:

```

import sys

from PyQt5.QtWidgets import QApplication, QWidget, QLineEdit, QHBoxLayout, QVBoxLayout, QGridLayout, QPushButton, QMessageBox

```

В этом блоке кода мы импортируем модуль `sys` для работы с системными параметрами и `QApplication`, `QWidget`, `QLineEdit`, `QHBoxLayout`, `QVBoxLayout`, `QGridLayout`, `QPushButton`, `QMessageBox` из PyQt5 для создания графического интерфейса и взаимодействия с пользователем.

2. Определение класса `Calculator`, унаследованного от `QWidget`:

```

class Calculator(QWidget):

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

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

```

В этом классе мы создаем главное окно калькулятора.

3. Создание интерфейса калькулятора:

self.grid = QGridLayout()

self.setLayout(self.grid)

Здесь мы создаем сетку для размещения элементов интерфейса и устанавливаем ее в качестве макета для главного окна калькулятора.

4. Создание поля ввода и кнопок для цифр и операторов:

```

# Создаем поле ввода и добавляем его на соответствующий слой

self.input = QLineEdit(self)

self.grid.addWidget(self.input, 0, 0, 1, 4)

# Создаем кнопки с цифрами и добавляем их на соответствующие слои

# ...

# Создаем кнопки с операторами и добавляем их на соответствующие слои

# ...

```

Здесь мы создаем поле ввода, кнопки с цифрами от 0 до 9, кнопку с десятичной точкой и кнопки с операторами (`+`, `-`, `\*`, `/`). Каждый элемент размещается на сетке с помощью метода `addWidget()`.

5. Подключение обработчиков событий для кнопок:

```

# Подключаем обработчики событий для каждой кнопки

self.b\_1.clicked.connect(lambda: self.\_button("1"))

self.b\_2.clicked.connect(lambda: self.\_button("2"))

# ...

self.b\_result.clicked.connect(self.\_calculate)

```

Здесь мы связываем каждую кнопку с соответствующим методом. Например, при нажатии кнопки "1" вызывается метод `\_button("1")`, который добавляет символ "1" в поле ввода. Метод `\_calculate()` вызывается при нажатии кнопки "=" и выполняет вычисление выражения, введенного пользователем, используя функцию `eval()`.

6. Установка параметров главного окна и запуск приложения

:

```

self.setGeometry(100, 100, 100, 100)

self.setWindowTitle('Mega Duper Super Calculator v 0.1 alpha beta - trash')

self.show()

# Запуск приложения

app = QApplication(sys.argv)

win = Calculator()

win.show()

sys.exit(app.exec\_())

```

Здесь мы устанавливаем размеры и заголовок главного окна, а затем запускаем приложение с помощью `QApplication`. Создается экземпляр класса `Calculator`, окно отображается на экране, и приложение начинает свой цикл выполнения.

Таким образом, данный код создает простой калькулятор с графическим интерфейсом, который позволяет пользователю вводить числа и выполнять базовые математические операции.

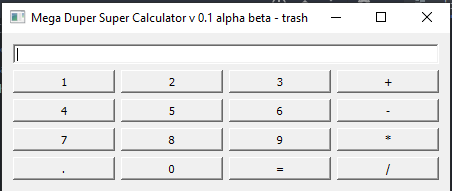


Рис. 2 – окно калькулятора

# Заключение

# Список использованных источников