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»

Кафедра «Математическая кибернетика и информационные технологии»

Лабораторная работа № 6

Создание оконного приложения-калькулятора

по дисциплине

«Введение в информационные технологии»

Выполнил: студент гр. БВТ2201
Шамсутдинов Р.Ф.

Проверил:

Москва, 2023 г

Импортируем нужные модули

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

Добавляем макеты на которые потом будем прикреплять кнопки и ввод

```
class Calculator(QWidget):
    def __init__(self):
        super(Calculator, self).__init__()

        self.vbox = QVBoxLayout(self)
        self.hbox_input = QHBoxLayout()
        self.hbox_first = QHBoxLayout()
        self.hbox_second = QHBoxLayout()
        self.hbox_third = QHBoxLayout()
        self.hbox_result = QHBoxLayout()

        self.vbox.addLayout(self.hbox_input)
        self.vbox.addLayout(self.hbox_first)
        self.vbox.addLayout(self.hbox_second)
        self.vbox.addLayout(self.hbox_third)
        self.vbox.addLayout(self.hbox_result)
```

Добавляем кнопки и строку ввода

```
        self.input = QLineEdit(self)
        self.hbox_input.addWidget(self.input)

        self.b_1 = QPushButton("1", self)
        self.hbox_first.addWidget(self.b_1)

        self.b_2 = QPushButton("2", self)
        self.hbox_first.addWidget(self.b_2)

        self.b_3 = QPushButton("3", self)
        self.hbox_first.addWidget(self.b_3)

        self.b_4 = QPushButton("4", self)
        self.hbox_second.addWidget(self.b_4)

        self.b_5 = QPushButton("5", self)
        self.hbox_second.addWidget(self.b_5)

        self.b_6 = QPushButton("6", self)
        self.hbox_second.addWidget(self.b_6)

        self.b_7 = QPushButton("7", self)
        self.hbox_third.addWidget(self.b_7)

        self.b_8 = QPushButton("8", self)
        self.hbox_third.addWidget(self.b_8)

        self.b_9 = QPushButton("9", self)
        self.hbox_third.addWidget(self.b_9)

        self.b_0 = QPushButton("0", self)
        self.hbox_third.addWidget(self.b_0)

        self.b_decimal = QPushButton(".", self)
        self.hbox_third.addWidget(self.b_decimal)

        self.b_plus = QPushButton("+", self)
        self.hbox_first.addWidget(self.b_plus)

        self.b_minus = QPushButton("-", self)
        self.hbox_first.addWidget(self.b_minus)

        self.b_mult = QPushButton("*", self)
        self.hbox_second.addWidget(self.b_mult)

        self.b_div = QPushButton("/", self)
        self.hbox_second.addWidget(self.b_div)

        self.b_result = QPushButton("=", self)
        self.hbox_result.addWidget(self.b_result)
```

Обрабатываем нажатия на кнопки вызовом нужных лямда функций

```
        self.b_plus.clicked.connect(lambda: self._operation("+"))
        self.b_minus.clicked.connect(lambda: self._operation("-"))
        self.b_mult.clicked.connect(lambda: self._operation("*"))
        self.b_div.clicked.connect(lambda: self._operation("/"))
        self.b_result.clicked.connect(self._result)

        self.b_1.clicked.connect(lambda: self._button("1"))
        self.b_2.clicked.connect(lambda: self._button("2"))
        self.b_3.clicked.connect(lambda: self._button("3"))
        self.b_4.clicked.connect(lambda: self._button("4"))
        self.b_5.clicked.connect(lambda: self._button("5"))
        self.b_6.clicked.connect(lambda: self._button("6"))
        self.b_7.clicked.connect(lambda: self._button("7"))
        self.b_8.clicked.connect(lambda: self._button("8"))
        self.b_9.clicked.connect(lambda: self._button("9"))
        self.b_0.clicked.connect(lambda: self._button("0"))
        self.b_decimal.clicked.connect(lambda: self._button("."))

        self.initUI()
```

Создаем две функции, первая конкатенирует введенные значения в одну строку, вторая проверяет их на правильность

```
    def _button(self, param):

        line = self.input.text()
        self.input.setText(line + param)

    def _operation(self, op):

        flag = True
        try:
            a = float(self.input.text())
        except ValueError:
            flag = False

        if not flag:
            self.error_message('input_error')
        else:
            self.num_1 = float(self.input.text())
            self.op = op
            self.input.setText("")
```

Создаем еще две функции, первая отвечает за логику работы калькулятора, а вторая за обработку ошибок

```
def _result(self):

    flag = True
    try:
        a = float(self.input.text())
    except ValueError:
        flag = False

    if not flag:
        self.error_message('input_error')
    else:
        self.num_2 = float(self.input.text())
        if self.op == "+":
            self.input.setText(str(self.num_1 + self.num_2))
        elif self.op == "-":
            self.input.setText(str(self.num_1 - self.num_2))
        elif self.op == "*":
            self.input.setText(str(self.num_1 * self.num_2))
        elif self.op == "/":
            if self.num_2 == 0:
                self.error_message('div_by_zero')
            else:
                self.input.setText(str(self.num_1 / self.num_2))

def error_message(self, error_type):

    if error_type == 'div_by_zero':
        msg = QMessageBox.question(self,'error', 'You cannot divide by zero!', QMessageBox.Yes)
        self.input.setText("")

    if error_type == 'input_error':
        msg = QMessageBox.question(self,'error', 'You can only input numbers', QMessageBox.Yes)
        self.input.setText("")
```

Создаем функцию, которая отвечает за параметры юзер интерфейса, вызываем ее

```
def initUI(self):

    self.setGeometry(300, 300, 800, 800)
    self.setWindowTitle('Calculator')
    self.show()
```

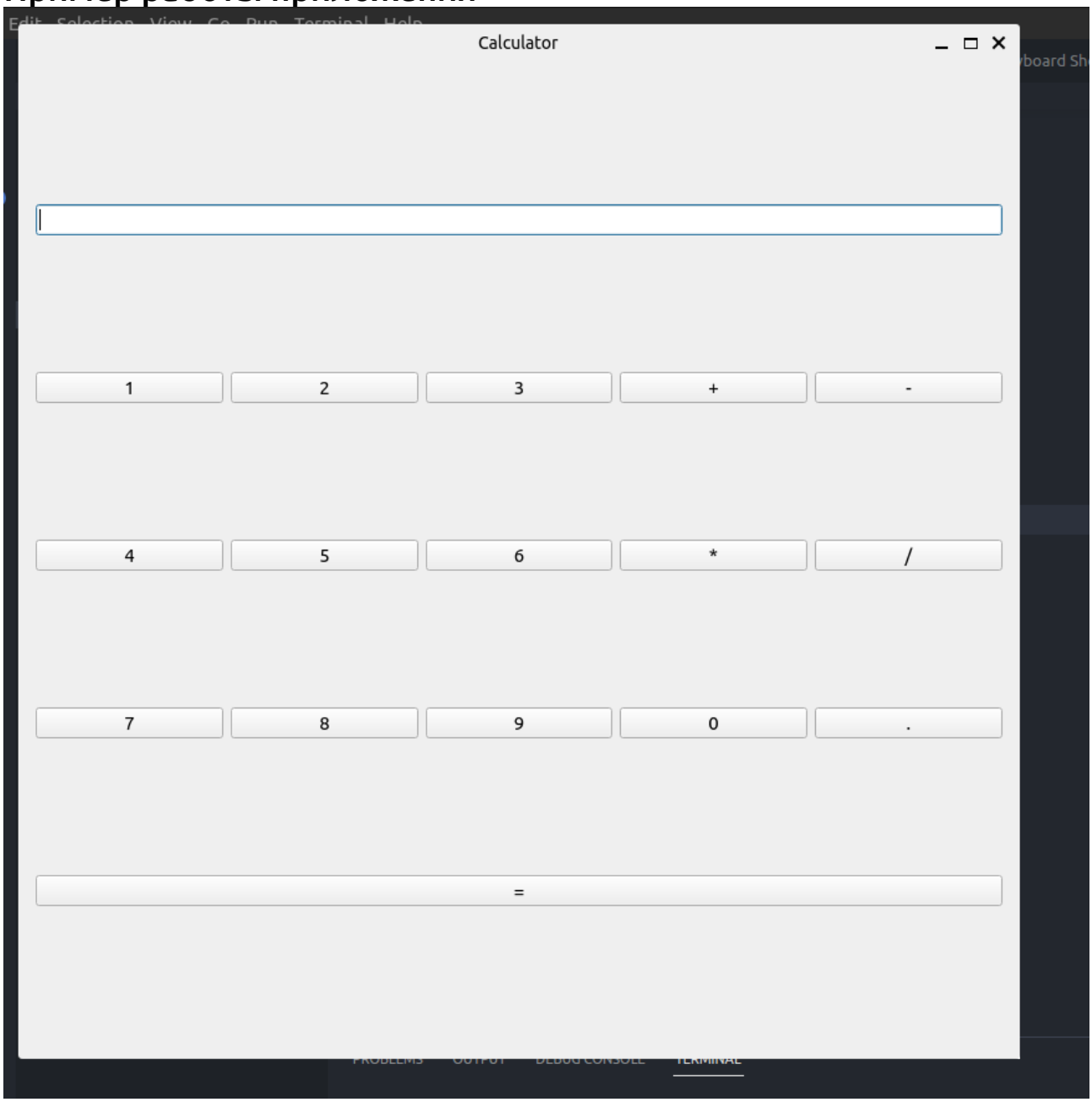
```
self.initUI()
```

Используя конструкцию иф нейм равно мейн, которая запускает наше приложение, только в том случае, если был запущен файл, в тором находится данная конструкция

```
if __name__ == '__main__':

    app = QApplication(sys.argv)
    win = Calculator()
    sys.exit(app.exec_())
```

Пример работы приложения



Вывод: проделав данную работу, я научился создавать простое десктопное приложение, работать в ооп стиле, так как использовал библиотеку PyQt5.