Реализовать обработку всех исключений при регистрации

Лабораторная работа №6: Создание оконного приложения-калькулятора

---

1. **Создаем новый проект в PyCharm с названием Calculator**

2. **Устанавливаем библиотеку для создания оконных приложений PyQt5**

```
pip3 install PyQt5
```

3. **В проекте создаем новый файл с названием calculator.py**

4. **Импортируем необходимые библиотеки для создания приложения**

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

Библиотека sys используется для получения информации об операционной системе.

Библиотека PyQt5 используется для создания оконных приложений.

PyQt5 используется в нашем проекте не полностью для уменьшения объема зависимостей приложения. Поэтому мы импортируем лишь некоторые классы из нее.

QApplication – управляет потоком управления и основными настройками приложения с графическим интерфейсом

QWidget – является базовым классом для всех объектов пользовательского интерфейса

QLineEdit – виджет, который разрешает вводить и редактировать одну строку текста

QHBoxLayout – выстраивает виджеты по горизонтали

QVBoxLayout – выстраивает виджеты по вертикали

QPushButton – кнопка, на которую можно нажимать

5. **Создаем класс Calculator и наследуем его от класса QWidgets**

```
class Calculator(QWidget):
  def __init__(self):
```

```
super(Calculator, self).__init__()
```

6. **Внутри конструктора создаем оси выравнивания**

```
self.vbox = QVBoxLayout(self)
self.hbox_input = QHBoxLayout()
self.hbox_first = QHBoxLayout()
self.hbox_result = QHBoxLayout()

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

Вертикальная ось будет главной в окне.
К ней привязываем горизонтальные оси выравнивания с помощью функции addLayout()

7. **В конструкторе создаем виджеты и привязываем их к соответствующим осям выравнивания**

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

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

Привязка виджетов к осям осуществляется с помощью функции addWidget()

8. **Создаем события, отвечающие за реакции на нажатия по кнопкам**

```
self.b_plus.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"))
```

Функция connect(<имя_функции/метода>), вызывает функцию/метод с именем указанным в аргументах. В указанную функцию/метод нельзя передавать аргументы. Для решения этой проблемы используем lambda-функции.

9. **Создаем метод класса для обработки кнопок, отвечающих за ввод цифр в линию ввода текста**

```
def _button(self, param):
  line = self.input.text()
  self.input.setText(line + param)
```

Уже существующая строка в линии ввода конкатенируется с аргументом param и устанавливается как отображаемый в линии ввода текст.

10. **Создаем метод класса для обработки нажатия на кнопку математической операции**

```
def _operation(self, op):
  self.num_1 = int(self.input.text())
  self.op = op
  self.input.setText("")
```

Запоминаем первое введенное число в целочисленном типе данных.
Запоминаем в качестве операции аргумент op.
Очищаем линию ввода.

11. **Создаем метод класса для обработки нажатия на кнопку результата**

```
def _result(self):
    self.num_2 = int(self.input.text())
    if self.op == "+":
        self.input.setText(str(self.num_1 + self.num_2))
```

Запоминаем второе введенное число в целочисленном типе данных. Производим вычисление в зависимости от операции и устанавливаем его в качестве текста в линию ввода.

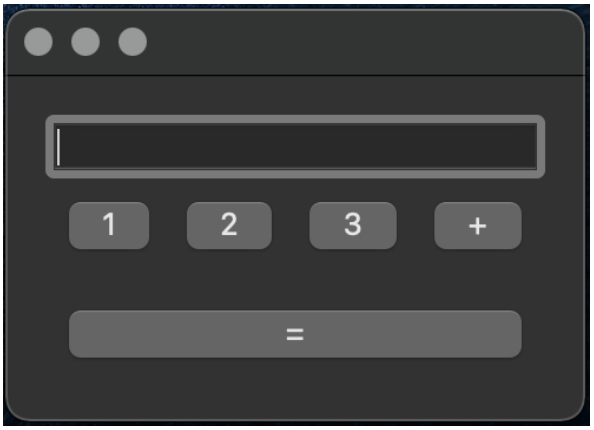
12. **Запускаем приложение**

```
app = QApplication(sys.argv)

win = Calculator()
win.show()

sys.exit(app.exec_())
```

Приложение должно выглядеть подобным образом:



13. **Домашнее задание:**

- Обработать все возможные исключения
- Добавить кнопку для добавления плавающей точки
- Добавить кнопки для математических операций вычитания, умножения, деления
- Создать для этих кнопок методы-обработчики