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»

Лабораторная работа №3

«Создание калькулятора»

По дисциплине

«Введение в ИТ»

Выполнил: студент

Группы БИН2002

Сас А.М.

Проверил: Чайка А.Д.

Москва 2021 г.

Содержание

[**1.** **Цель работы** 3](#_Toc89954318)

[**2.** **Выполнение** 3](#_Toc89954319)

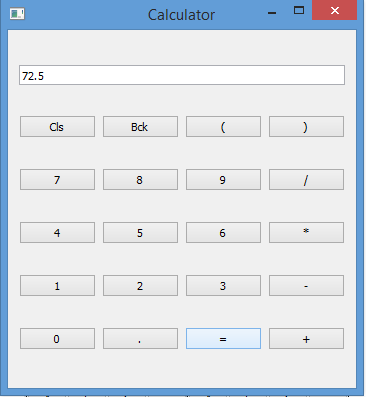
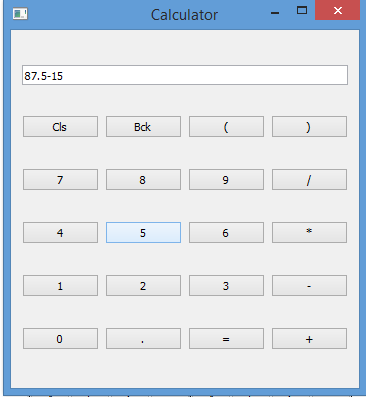
[**3.** **Вывод** 5](#_Toc89954320)

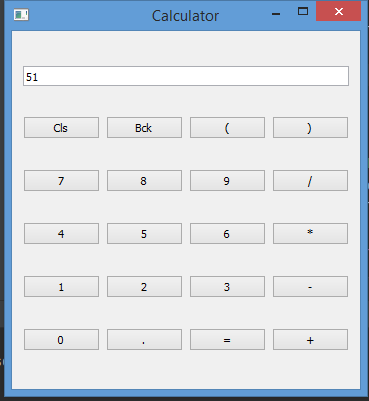
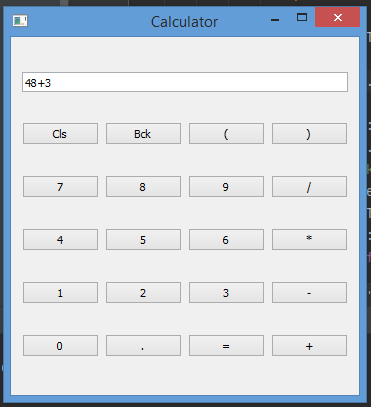
1. **Цель работы**

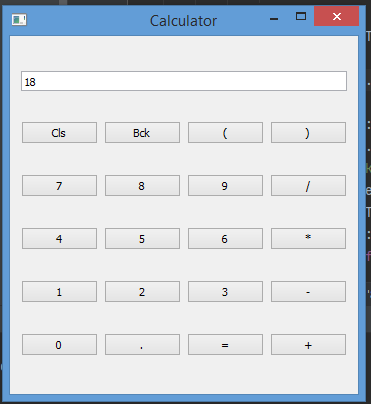
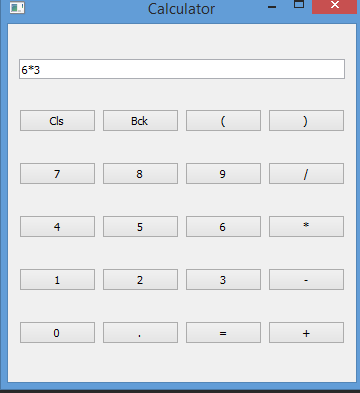
Создание калькулятора с использованием Python и библиотеки PyQt5

1. **Выполнение**

Калькулятор может выполнять все требуемые функции (Рисунок 1)







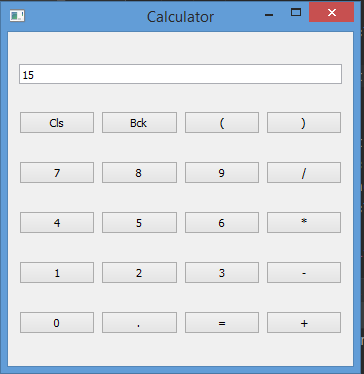
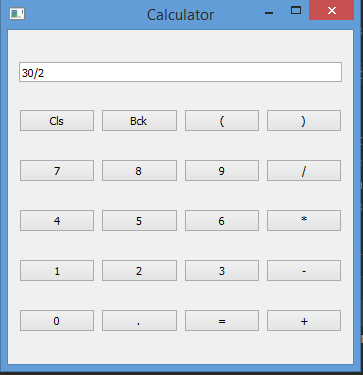


Рисунок 1 - Проверка работы калькулятора

При попытке деления на ноль приложение выдает ошибку (Рисунок 2)

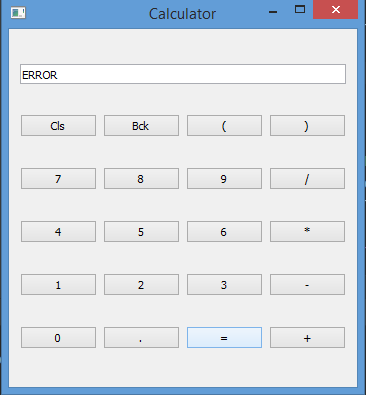
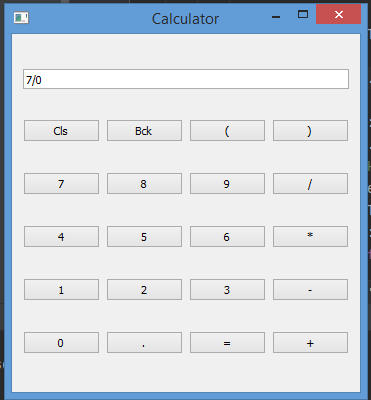


Рисунок 2 - Попытка деления на 0

1. **Вывод**

В ходе выполнения работы, я научилась созданию оконных приложений с помощью PyQt

**Приложение:**

**Программный код файла main.py**

import sys

import sympy

from PyQt5.QtWidgets import QApplication, QWidget, QGridLayout, QPushButton, QLineEdit

class Window(QWidget):

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

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

grid = QGridLayout()

self.calcText = QLineEdit()

self.setLayout(grid)

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

names = ['Cls', 'Bck', '(', ')',

'7', '8', '9', '/',

'4', '5', '6', '\*',

'1', '2', '3', '-',

'0', '.', '=', '+']

positions = [(i+1,j) for i in range(6) for j in range(4)]

buttons = []

for position, name in zip(positions, names):

button = QPushButton(name)

buttons.append(button)

grid.addWidget(button, \*position)

buttons[0].clicked.connect(lambda: self.butonact(names[0]))

buttons[1].clicked.connect(lambda: self.butonact(names[1]))

buttons[2].clicked.connect(lambda: self.butonact(names[2]))

buttons[3].clicked.connect(lambda: self.butonact(names[3]))

buttons[4].clicked.connect(lambda: self.butonact(names[4]))

buttons[5].clicked.connect(lambda: self.butonact(names[5]))

buttons[6].clicked.connect(lambda: self.butonact(names[6]))

buttons[7].clicked.connect(lambda: self.butonact(names[7]))

buttons[8].clicked.connect(lambda: self.butonact(names[8]))

buttons[9].clicked.connect(lambda: self.butonact(names[9]))

buttons[10].clicked.connect(lambda: self.butonact(names[10]))

buttons[11].clicked.connect(lambda: self.butonact(names[11]))

buttons[12].clicked.connect(lambda: self.butonact(names[12]))

buttons[13].clicked.connect(lambda: self.butonact(names[13]))

buttons[14].clicked.connect(lambda: self.butonact(names[14]))

buttons[15].clicked.connect(lambda: self.butonact(names[15]))

buttons[16].clicked.connect(lambda: self.butonact(names[16]))

buttons[17].clicked.connect(lambda: self.butonact(names[17]))

buttons[18].clicked.connect(lambda: self.butonact(names[18]))

buttons[19].clicked.connect(lambda: self.butonact(names[19]))

#for keyindx in range(0, len(names)):

# print(names[keyindx])

# buttons[keyindx].clicked.connect(lambda: self.butonact(names[keyindx]))

self.move(300, 150)

self.setWindowTitle('Calculator')

self.show()

def make\_calculate(self):

try:

return eval(str(sympy.sympify(self.calcText.text(), evaluate=True)))

except Exception as e:

print(e)

return 'ERROR'

def butonact(self, param):

nowLine = self.calcText.text()

if param in ['7', '8', '9', '/',

'4', '5', '6', '\*',

'1', '2', '3', '-',

'0', '.', '+', '(', ')']:

if len(nowLine) > 0:

if nowLine[-1] in ['/', '\*', '-', '.', '+'] and param in ['/', '\*', '-', '.', '+', '=']:

pass

else:

self.calcText.setText(nowLine + str(param))

else:

self.calcText.setText(nowLine + str(param))

elif param in ['Cls', 'Bck', '=']:

if param == 'Cls':

self.calcText.setText('')

elif param == 'Bck':

if len(nowLine) > 0:

self.calcText.setText(nowLine[:-1])

elif param == '=':

res = str(self.make\_calculate())

#print(res)

self.calcText.setText(res)

#print(param)

app = QApplication(sys.argv)

win = Window()

win.show()

sys.exit(app.exec())