FOSS LAB REPORT

Amruth DD
Roll.NO:09
S4 CS

February 2020

1 Experiment 1

1.1 Aim

To implement a GUI application using QT

1.2 Source code

```
import sys
from PyQt4 import QtGui, QtCore
from PyQt4.QtCore import Qt
import random
from math import sqrt
from math import factorial

num = 0.0
newNum = 0.0
sumAll = 0.0
operator = ""

opVar = False
sumIt = 0
positions = [[10, 145], [50, 145], [90, 145], [10, 110], [50, 110], [90, 110], [10, 75], [50
random.shuffle(positions)

class Main(QtGui.QMainWindow):

    def __init__(self):
        QtGui.QMainWindow.__init__(self)
        self.initUI()

    def initUI(self):
```

```
self.line = QtGui.QLineEdit(self)
self.line.move(5,5)
self.line.setReadOnly(True)
self.line.setAlignment(Qt.AlignRight)
self.line.resize(200,25)

zero = QtGui.QPushButton("0",self)
zero.move(10,180)
zero.resize(35,30)

one = QtGui.QPushButton("1",self)
one.move(positions[0][0],positions[0][1])
one.resize(35,30)

two = QtGui.QPushButton("2",self)
two.move(positions[1][0],positions[1][1])
two.resize(35,30)

three = QtGui.QPushButton("3",self)
three.move(positions[2][0],positions[2][1])
three.resize(35,30)

four = QtGui.QPushButton("4",self)
four.move(positions[3][0],positions[3][1])
four.resize(35,30)

five = QtGui.QPushButton("5",self)
five.move(positions[4][0],positions[4][1])
five.resize(35,30)

six = QtGui.QPushButton("6",self)
six.move(positions[5][0],positions[5][1])
six.resize(35,30)

seven = QtGui.QPushButton("7",self)
seven.move(positions[6][0],positions[6][1])
seven.resize(35,30)

eight = QtGui.QPushButton("8",self)
eight.move(positions[7][0],positions[7][1])
eight.resize(35,30)

nine = QtGui.QPushButton("9",self)
nine.move(positions[8][0],positions[8][1])
nine.resize(35,30)
```

```
switch = QtGui.QPushButton("+/-",self)
switch.move(50,180)
switch.resize(35,30)
switch.clicked.connect(self.Switch)

point = QtGui.QPushButton(".",self)
point.move(90,180)
point.resize(35,30)
point.clicked.connect(self.pointClicked)

div = QtGui.QPushButton("/",self)
div.move(130,75)
div.resize(35,30)

mult = QtGui.QPushButton("*",self)
mult.move(130,110)
mult.resize(35,30)

minus = QtGui.QPushButton("-",self)
minus.move(130,145)
minus.resize(35,30)

plus = QtGui.QPushButton("+",self)
plus.move(130,180)
plus.resize(35,30)

sqrt = QtGui.QPushButton("fact",self)
sqrt.move(170,75)
sqrt.resize(35,30)
sqrt.clicked.connect(self.Fact)

squared = QtGui.QPushButton("^2",self)
squared.move(170,110)
squared.resize(35,30)
squared.clicked.connect(self.Squared)

equal = QtGui.QPushButton("=",self)
equal.move(170,145)
equal.resize(35,65)
equal.clicked.connect(self.Equal)

c = QtGui.QPushButton("C",self)
c.move(145,35)
c.resize(60,30)
c.clicked.connect(self.C)
```

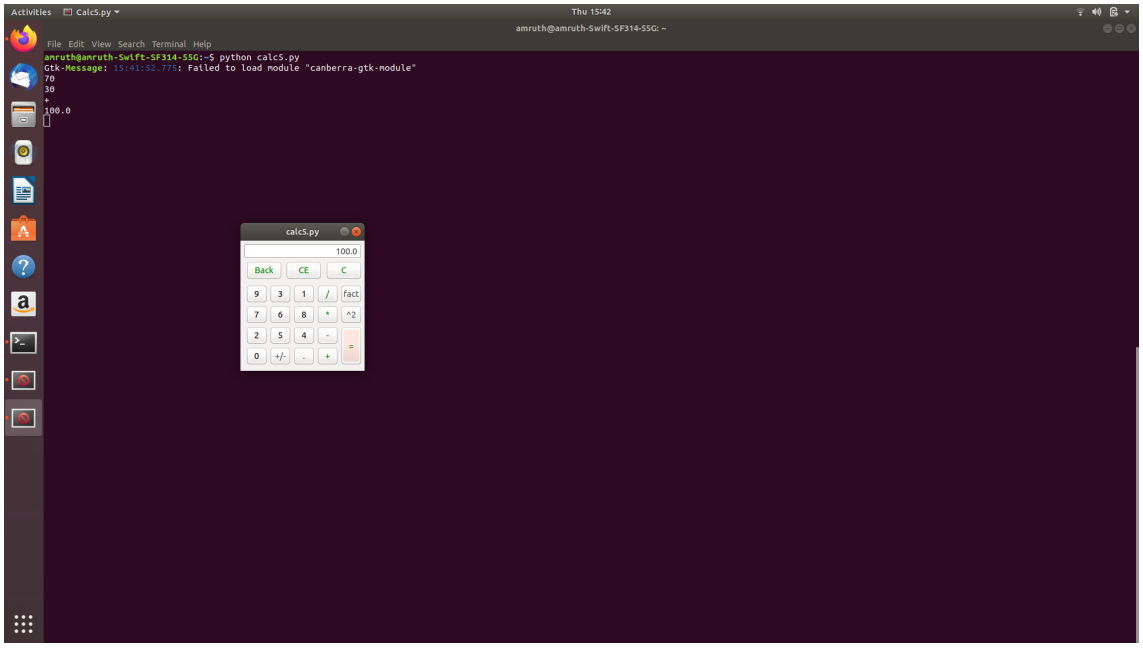
```
        ce = QtGui.QPushButton("CE",self)
        ce.move(77,35)
        ce.resize(60,30)
        ce.clicked.connect(self.CE)

        back = QtGui.QPushButton("Back",self)
        back.move(10,35)
        back.resize(60,30)
        back.clicked.connect(self.Back)

        nums = [zero,one,two,three,four,five,six,seven,eight,nine]

        ops = [back,c,ce,div,mult,minus,plus,equal]

        rest = [switch,squared,sqrt,point]

        for i in nums:
            i.setStyleSheet("color:black;")
            i.clicked.connect(self.Nums)

        for i in ops:
            i.setStyleSheet("color:green;")

        for i in ops[3:7]:
            i.clicked.connect(self.Operator)

        self.setGeometry(400,400,240,250)
        self.setFixedSize(210,220)
        self.setWindowTitle("")
        self.setWindowIcon(QtGui.QIcon(""))
        self.show()

    def Nums(self):
        global num
        global newNum
        global opVar

        sender = self.sender()

        newNum = int(sender.text())
        setNum = str(newNum)


        if opVar == False:
            self.line.setText(self.line.text() + setNum)
```

```
    else:
        self.line.setText(setNum)
        opVar = False



def pointClicked(self):
    global opVar

    if "." not in self.line.text():
        self.line.setText(self.line.text() + ".")

def Switch(self):
    global num

    try:
        num = int(self.line.text())

    except:
        num = float(self.line.text())

    num = num - num * 2

    numStr = str(num)

    self.line.setText(numStr)

def Operator(self):
    global num
    global opVar
    global operator
    global sumIt

    sumIt += 1

    if sumIt > 1:
        self.Equal()

    num = self.line.text()

    sender = self.sender()

    operator = sender.text()
```

```
    opVar = True



def Equal(self):
    global num
    global newNum
    global sumAll
    global operator
    global opVar
    global sumIt

    sumIt = 0

    newNum = self.line.text()

    print(num)
    print(newNum)
    print(operator)

    if operator == "+":
        sumAll = float(num) + float(newNum)

    elif operator == "-":
        sumAll = float(num) - float(newNum)

    elif operator == "/":
        try:
            sumAll = float(num) / float(newNum)
        except:
            self.line.setText("Bad human. No division by zero.")
            return

    elif operator == "*":
        sumAll = float(num) * float(newNum)

    print(sumAll)
    self.line.setText(str(sumAll))
    opVar = True

def Back(self):
    self.line.backspace()

def C(self):
    global newNum
    global sumAll
```

```
        global operator
        global num

        self.line.clear()

        num = 0.0
        newNum = 0.0
        sumAll = 0.0
        operator = ""

    def CE(self):
        self.line.clear()

    def Fact(self):
        global num

        num = float(self.line.text())
        n = sqrt(num)
        n = factorial(num)
        num = n

        self.line.setText(str(num))

    def Squared(self):
        global num

        num = float(self.line.text())

        n = num ** 2

        num = n

        self.line.setText(str(n))

def main():
    app = QtGui.QApplication(sys.argv)
    main= Main()
    main.show()

    sys.exit(app.exec_())

if __name__ == "__main__":
    main()
```

1.3 Output



1.4 Result

A GUI application(Calculator) was implemented using pyqt4