LAPORAN AKHIR PRAKTIKUM

Mata Praktikum	: Algoritma dan Pemprograman 2B
Kelas	: 1IA24
Praktikum Ke-	: 6
Tanggal	: Selasa, 16 mei 2022
Materi	: Pemprograman Visual Berbasis Python PYQT5
NPM	: 50422818
Nama	: Lutfi Robbani
Ketua asisten	: GEDE OKE
Nama asisten	:
Paraf asisten	:
Jumlah Lembar	: 8 lembar

LABORATORIUM TEKNIK INFORMATIKA

UNIVERSITAS GUNADARMA

2022

LISTENING

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

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

        self.setWindowTitle('Calculator')
        self.setGeometry(300, 300, 300, 300)

        self.text_box = QLineEdit(self)
        self.text_box.setReadOnly(True)
        self.text_box.setAlignment(Qt.AlignRight)
        self.text_box.setFont(QtGui.QFont('Arial', 12))

        vbox = QVBoxLayout()
        vbox.addWidget(self.text_box)

        self.buttons = [
            ['7', '8', '9', '/'],
            ['4', '5', '6', '*'],
            ['1', '2', '3', '-'],
            ['0', '.', '=', '+'],
            ['C', '<-']
        ]

        for row in self.buttons:
            hbox = QHBoxLayout()
            for label in row:
                button = QPushButton(label, self)
                button.clicked.connect(self.buttonClicked)
                hbox.addWidget(button)
            vbox.addLayout(hbox)

        self.setLayout(vbox)
        self.setStyleSheet("""
            QPushButton {
                background-color: #45AF50;
                border: 1px solid #CCCCCC;
                border-radius: 5px;
                padding : 10px;
                font-size: 14px;
                color: #FFFFFF;
            }
            QPushButton:hover {
                background-color: #45a049;
            }
            QPushButton:pressed {
                background-color: #3e8e41;
            }
            QLineEdit {
                background-color: #FFFFFF;
                border: 1px solid #CCCCCC;
                border-radius: 5px;
                padding: 5px;
```

```
            font-size: 14px;
        }
    """)

def buttonClicked(self):
    button = self.sender()
    key = button.text()
    if key == '=':
        try:
            result = str(eval(self.text_box.text()))
            self.text_box.setText(result)
        except:
            self.text_box.setText('Error')
    elif key == 'C':
        self.text_box.clear()
    elif key == '<-':
        text = self.text_box.text()[:-1]
        self.text_box.setText(text)
    else:
        self.text_box.setText(self.text_box.text() + key)


if __name__ == '__main__':
    app = QApplication(sys.argv)
    calc = Calculator()
    calc.show()
    sys.exit(app.exec_())
```

LOGIKA

```
1  import sys
2  from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout, QHBoxLayout, QPushButton, QLineEdit
3  from PyQt5.QtCore import Qt
4  from PyQt5 import QtGui
```

Pertama import class yang di perlukan untuk menjalankan program ini, disini ada class PyQt5.Qtwidgets, PyQt5.QtCore, PyQt5.

```
6  class Calculator(QWidget):
7      def __init__(self):
8          super().__init__()
9
10         self.setWindowTitle('Calculator')
11         self.setGeometry(300, 300, 300, 300)
12
13         self.text_box = QLineEdit(self)
14         self.text_box.setReadOnly(True)
15         self.text_box.setAlignment(Qt.AlignRight)
16         self.text_box.setFont(QtGui.QFont('Arial', 12))
```

Selanjutnya buat class calculator dengan parameter QWidget, yang di import dari class pyqt5.qtwidgets. setelah itu buat fungsi __init__ dengan parameter self.

Lalu judul projek dengan nama calculator, dengan Panjang kanan, kiri, atas, bawah 300 pixel, buat kolom QlineEdit untuk meletakan hasil perhitungan, dan hanya bisa di baca, diatur rata kanan dengan font arial besar huruf 12.

```
18         vbox = QVBoxLayout()
19         vbox.addWidget(self.text_box)
20
21         self.buttons = [
22             ['7', '8', '9', '/'],
23             ['4', '5', '6', '*'],
24             ['1', '2', '3', '-'],
25             ['0', '.', '=', '+'],
26             ['C', '<-']
27         ]
28
29         for row in self.buttons:
30             hbox = QHBoxLayout()
31             for label in row:
32                 button = QPushButton(label, self)
33                 button.clicked.connect(self.buttonClicked)
34                 hbox.addWidget(button)
35             vbox.addLayout(hbox)
```

Selanjutnya program membuat objek QVBoxLayout dengan nama vbox dan menambahkan self.text_box ke dalamnya menggunakan metode addWidget(). Ini berarti self.text_box akan ditempatkan di dalam vbox.

Setelah itu, program mendefinisikan variabel buttons sebagai daftar multidimensi yang berisi teks untuk tombol-tombol kalkulator yang akan ditampilkan di antarmuka . Setiap sub-daftar mewakili baris tombol pada kalkulator.

Dalam logika tersebut, terdapat nested loop untuk mengatur tata letak tombol-tombol kalkulator. Pertama, program melakukan iterasi melalui setiap sub-daftar dalam buttons, yang mewakili baris tombol pada kalkulator.

Kemudian, untuk setiap row (baris tombol), program membuat sebuah QHBoxLayout dengan nama hbox yang akan berisi tombol-tombol dalam baris tersebut. Program melakukan iterasi lagi melalui setiap label dalam row, dan membuat sebuah tombol dengan label yang sesuai menggunakan QPushButton. Tombol tersebut ditambahkan ke dalam hbox menggunakan metode addWidget().

Selain itu, program juga menghubungkan setiap tombol dengan sinyal clicked ke metode buttonClicked() menggunakan button.clicked.connect(self.buttonClicked). Ini memastikan bahwa metode buttonClicked() akan dipanggil ketika tombol diklik.

Setelah selesai dengan baris tombol saat ini, hbox (baris tombol) ditambahkan ke dalam vbox menggunakan metode addLayout(). Hal ini dilakukan untuk setiap baris tombol dalam buttons, sehingga semua tombol-tombol kalkulator akan ditambahkan ke dalam tata letak vertikal vbox.

Dengan demikian, logika di atas akan menampilkan tombol-tombol kalkulator sesuai dengan konfigurasi dalam buttons di bawah kotak teks dalam tata letak vertikal. Selain itu, setiap tombol juga akan terhubung dengan metode buttonClicked() untuk menangani klik tombol tersebut.

```
37          self.setLayout(vbox)
38          self.setStyleSheet("""
39              QPushButton {
40                  background-color: #45AF50;
41                  border: 1px solid #CCCCCC;
42                  border-radius: 5px;
43                  padding : 10px;
44                  font-size: 14px;
45                  color: #FFFFFF;
46              }
47              QPushButton:hover {
48                  background-color: #45a049;
49              }
50              QPushButton:pressed {
51                  background-color: #3e8e41;
52              }
53              QLineEdit {
54                  background-color: #FFFFFF;
55                  border: 1px solid #CCCCCC;
56                  border-radius: 5px;
57                  padding: 5px;
```

```
            font-size: 14px;
        }
    """)

```

Selanjutnya gunakan self.setLayout(vbox) untuk mengatur tata letak utama objek QWidget dengan menggunakan QVBoxLayout vbox yang telah dibuat sebelumnya. gunakan juga self.setStyleSheet() untuk mengatur gaya tampilan antarmuka pengguna menggunakan CSS. Setiap elemen antarmuka user diberi properti gaya sesuai dengan selektor yang diberikan.

Tombol QPushButton diberi properti gaya seperti warna latar belakang, border, radius, padding, ukuran font, dan warna teks. Ketika tombol dihover (melayang di atasnya), warna latar belakang tombol akan berubah. Ketika tombol ditekan, warna latar belakang tombol akan berubah lagi. Kotak teks QLineEdit diberi properti gaya seperti warna latar belakang, border, radius, padding, dan ukuran font

```
    def buttonClicked(self):
        button = self.sender()
        key = button.text()
        if key == '=':
            try:
                result = str(eval(self.text_box.text()))
                self.text_box.setText(result)
            except:
                self.text_box.setText('Error')
        elif key == 'C':
            self.text_box.clear()
        elif key == '<-':
            text = self.text_box.text()[:-1]
            self.text_box.setText(text)
        else:
            self.text_box.setText(self.text_box.text() + key)

```

Ketika tombol diklik, metode buttonClicked(self) dipanggil. Dalam metode ini, tombol yang diklik didapatkan menggunakan self.sender() dan teks tombol disimpan dalam variabel key.

Jika key adalah '=', ekspresi matematika dalam self.text_box.text() dievaluasi menggunakan eval(). Hasil evaluasi dikonversi menjadi string dan ditetapkan sebagai teks dalam self.text_box. Jika key adalah 'C', teks dalam self.text_box dibersihkan. Jika key adalah '<-', satu karakter terakhir dihapus dari teks dalam self.text_box. Jika key bukan '=', 'C', atau '<-', teks tombol ditambahkan ke dalam teks yang sudah ada dalam self.text_box.

```
79
80  if __name__ == '__main__':
81      app = QApplication(sys.argv)
82      calc = Calculator()
83      calc.show()
84      sys.exit(app.exec_())
```

Objek QApplication dibuat dengan menggunakan QApplication(sys.argv). Setelah itu objek Calculator dibuat dengan menggunakan Calculator(). Jendela Calculator ditampilkan dengan menggunakan calc.show(). Kemudian aplikasi Qt dieksekusi dengan menggunakan app.exec_(). Setelah aplikasi selesai dieksekusi, sys.exit() digunakan untuk keluar dari program.

Program menjalankan aplikasi kalkulator dengan membuat objek QApplication, objek Calculator, menampilkan jendela kalkulator, dan menjalankan siklus utama aplikasi Qt hingga jendela ditutup.

OUTPUT