MODUL 5
WIDGET DALAM PYQT

29. KELAS QLABEL, QLINEEDIT, DAN QPUSHBUTTON

Kontrol dasar dalam pemrograman GUI adalah label, text box dan button. Dalam penggunaan PyQt ketiga control tersebut direpresentasikan dengan kelas QLabel, QLineEdit dan QPushButton. Kelas QLabel digunakan untuk menapilkan teks atau gambar di dalam form. Beberapa metode yang sering digunakan dalam kelas QLabel adalah sebagai berikut

Metode	Fungsi
setText()	Memasukkan teks ke dalam objek label
setPixmap()	Memasukkan gambar ke dalam objek label
setNum()	Memasukkan nilai numerik ke dalam objek label
Clear()	Mengosongkan teks di dalam objek label

Sedangkan kelas QLineEdit digunakan untuk menampilkan control input (text box /edit box). QLineEdit hanya mampu menampung teks satu baris saja. Jika text lebih dari satu baris maka anda harus menggunakan QTextEdit. Metode yang sering digunakan pada QLineEdit adalah sebagai berikut.

Metode	Fungsi
setEchoMode()	Menentukan tampilan teks pada QLineEdit. Beberpa opsi yang dapat digunakan antara lain: Normal, NoEcho, Password dan PasswordEchoOnEdit
maxLength()	Menentukan karakter maksimal yang dapat dimasukkan ke dalam kontrol
setText()	Memasukkan teks ke dalam QLineEdit
clear()	Mengosongkan teks di dalam QLineEdit
setReadOnly()	Menjadikan control bersifat read only
isReadOnly()	Memeriksa apakah control bersifat read only

setEnabled()	Mengaktifkan control ketika melewati nilai true.
setFocus()	Memindahkan focus ke control QLineEdit

Sedangkan signal yang sering digunakan pada kelas QLineEdit adalah sebagai berikut.

Metode	Fungsi
textChanged()	Aktivasi pada saat user memasukkan teks ke dalam control QLineEdit
returnPressed()	Aktivasi pada saar user menekan tombol
editingFinished()	Aktivasi saat focus sudah meninggalkan control QLineEdit

Kelas QPushButton digunakan untuk menampilkan tombol di dalam form. Untuk contoh penggunaan kelas QLabel, QLineEdit dan Qpush button tuliskan kode program berikut pada computer anda.

```
##########################################################
# Nama file: DemoQLineEdit.py
##########################################################
import sys
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
    def __init__(self):
        super().__init__()
        self.setupUi()
    def setupUi(self):
        self.resize(400, 200)
        self.move(300, 300)
        self.setWindowTitle('Demo QLabel, QLineEdit, dan
QPushButton')
        self.label1 = QLabel()
        self.label1.setText('Bilangan pertama')
```

```
        self.numberEdit1 = QLineEdit()
        vbox1 = QVBoxLayout()
        vbox1.addWidget(self.label1)
        vbox1.addWidget(self.numberEdit1)
        self.label2 = QLabel()
        self.label2.setText('Bilangan kedua')
        self.numberEdit2 = QLineEdit()
        vbox2 = QVBoxLayout()
        vbox2.addWidget(self.label2)
        vbox2.addWidget(self.numberEdit2)
        self.label3 = QLabel()
        self.label3.setText('Hasil Perhitungan')
        self.resultEdit = QLineEdit()
        self.resultEdit.setReadOnly(True)
        vbox3 = QVBoxLayout()
        vbox3.addWidget(self.label3)
        vbox3.addWidget(self.resultEdit)
        vbox4 = QVBoxLayout()
        vbox4.addLayout(vbox1)
        vbox4.addLayout(vbox2)
        vbox4.addLayout(vbox3)
        vbox4.addStretch()
        self.addButton = QPushButton('&Tambah')
        self.substractButton = QPushButton('&Kurang')
        self.mulButton = QPushButton('K&ali')
        self.divButton = QPushButton('&Bagi')
        vbox5 = QVBoxLayout()
        vbox5.addWidget(self.addButton)
        vbox5.addWidget(self.substractButton)
        vbox5.addWidget(self.mulButton)
```

```
        vbox5.addWidget(self.divButton)
        vbox5.addStretch()
        layout = QHBoxLayout()
        layout.addLayout(vbox4)
        verticalLine = QFrame();
        verticalLine.setFrameShape(QFrame.VLine)
        verticalLine.setFrameShadow(QFrame.Sunken)
        layout.addWidget(verticalLine)
        layout.addLayout(vbox5)
        self.setLayout(layout)
        self.addButton.clicked.connect(self.addButtonClick)

self.substractButton.clicked.connect(self.substractButton
Click)
        self.mulButton.clicked.connect(self.mulButtonClick)
        self.divButton.clicked.connect(self.divButtonClick)
    def calculate(self, operator):
        a = float(self.numberEdit1.text())
        b = float(self.numberEdit2.text())
        if operator == '+': c = a + b
        elif operator == '-': c = a - b
        elif operator == '*': c = a * b
        else: c = a / b
        self.resultEdit.setText(str(c))
    def addButtonClick(self):
        self.calculate('+')
    def substractButtonClick(self):
        self.calculate('-')
    def mulButtonClick(self):
        self.calculate('*')
    def divButtonClick(self):
```

```
            self.calculate('/')
    if __name__ == '__main__':
        a = QApplication(sys.argv)
        form = MainForm()
        form.show()
        a.exec_()
```

30. KELAS QTEXTEDIT

Kelas QTextEdit berfungsi untuk memasukkan data teks yang jumlahnya lebih dari satu baris. Metode yang dapat digunakan dalam QTextEdit adalah setDocument() untuk mengisi teks ke dalam control dan method document() untuk mengambil teks dalam control. Dalam dokumen control QTextEdit bertipe QTextDocument, yang dapat dikonversi ke string menggunakan metode toPlainText(). Untuk contoh penggunaan kelas QTextEdit tuliskanlah kode program ini ke dalam computer anda.

```
##############################################################
# Nama file: DemoQTextEdit.py
##############################################################
    import sys
    from PyQt5.QtGui import *
    from PyQt5.QtCore import *
    from PyQt5.QtWidgets import *
    class MainForm(QWidget):
        def __init__(self):
            super().__init__()
            self.setupUi()
        def setupUi(self):
            self.resize(400, 200)
            self.move(300, 300)
            self.setWindowTitle('Demo QTextEdit')
            self.label1 = QLabel()
            self.label1.setText('No. HP')
```

```
        self.phoneEdit = QLineEdit()
        vbox1 = QVBoxLayout()
        vbox1.addWidget(self.label1)
        vbox1.addWidget(self.phoneEdit)
        self.label2 = QLabel()
        self.label2.setText('Pesan')
        self.messageEdit = QTextEdit()
        vbox2 = QVBoxLayout()
        vbox2.addWidget(self.label2)
        vbox2.addWidget(self.messageEdit)
        vbox3 = QVBoxLayout()
        vbox3.addLayout(vbox1)
        vbox3.addLayout(vbox2)
        self.sendButton = QPushButton('&Kirim SMS')
        self.cancelButton = QPushButton('&Batal')
        hbox = QHBoxLayout()
        hbox.addStretch()
        hbox.addWidget(self.sendButton)
        hbox.addWidget(self.cancelButton)
        layout = QVBoxLayout()
        layout.addLayout(vbox3)
        horizontalLine = QFrame();
        horizontalLine.setFrameShape(QFrame.HLine)
        horizontalLine.setFrameShadow(QFrame.Sunken)
        layout.addWidget(horizontalLine)
        layout.addLayout(hbox)
        self.setLayout(layout)
if __name__ == '__main__':
    a = QApplication(sys.argv)
    form = MainForm()
```

```
    form.show()
    a.exec_()
```

31. KELAS QRADIOBUTTON DAN QCHECKBOX

Untuk menampilkan control yang diigunakan sebagai pilihan, anda dapat menggunakan kelas QRadioButton atau QCheckBox. Beberapa metode yang digunakan dalam QRadioButton adalah sebgai berikut.

Metode	Fungsi
isChecked()	Memberikan nilai true jika control radio button aktof
seticon()	Menampilkan gambar icon pada rkontrol radio button
setText()	Memasukkan teks ke dalam control
setChecked()	Mengaktifkan control radio button

Sedangkan signal yang sering digunakan dalam kelas QRadioButton adalah sebagai berikut.

Metode	Fungsi
toggled()	Aktivasi saat control radio button berubah
clicked()	Aktivasi saat user melakukan klik pada control radio button

Untuk contoh control penggunaan QRadioButton tuliskan kode profram berikut pada computer anda.

```
##########################################################
# Nama file: DemoQRadioButton.py
##########################################################
```

```
import sys
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
```

```
    def __init__(self):
        super().__init__()
        self.setupUi()
    def setupUi(self):
        self.resize(400, 150)
        self.move(300, 300)
        self.setWindowTitle('Demo QRadioButton')
        self.label1 = QLabel()
        self.label1.setText('Bilangan pertama')
        self.numberEdit1 = QLineEdit()
        self.label2 = QLabel()
        self.label2.setText('Bilangan kedua')
        self.numberEdit2 = QLineEdit()
        grid = QGridLayout()
        grid.addWidget(self.label1, 0, 0)
        grid.addWidget(self.numberEdit1, 0, 1)
        grid.addWidget(self.label2, 1, 0)
        grid.addWidget(self.numberEdit2, 1, 1)
        self.addRadio = QRadioButton()
        self.addRadio.setText('&Tambah')
        self.addRadio.setChecked(True)
        self.substractRadio = QRadioButton()
        self.substractRadio.setText('&Kurang')
        self.mulRadio = QRadioButton()
        self.mulRadio.setText('K&ali')
        self.divRadio = QRadioButton()
        self.divRadio.setText('&Bagi')
        hbox = QHBoxLayout()
        hbox.addWidget(self.addRadio)
        hbox.addWidget(self.substractRadio)
```

```
        hbox.addWidget(self.mulRadio)
        hbox.addWidget(self.divRadio)
        self.resultLabel = QLabel('<b>Hasil penjumlahan:
</b>')


        self.calculateButton = QPushButton('Hitung')
        layout = QVBoxLayout()
        layout.addLayout(grid)
        layout.addLayout(hbox)
        layout.addWidget(self.resultLabel)
        layout.addWidget(self.calculateButton)
        layout.addStretch()
        self.setLayout(layout)
        self.addRadio.clicked.connect(
          lambda: self.resultLabel.setText('<b>Hasil
penjumlahan: </b>'))
        self.substractRadio.clicked.connect(
          lambda: self.resultLabel.setText('<b>Hasil
pengurangan: </b>'))
        self.mulRadio.clicked.connect(
          lambda: self.resultLabel.setText('<b>Hasil
perkalian: </b>'))
        self.divRadio.clicked.connect(
          lambda: self.resultLabel.setText('<b>Hasil
pembagian: </b>'))

self.calculateButton.clicked.connect(self.calculateButton
Click)
    def calculateButtonClick(self):
        a = float(self.numberEdit1.text())
        b = float(self.numberEdit2.text())
        if self.addRadio.isChecked(): c = a + b
```

```
        elif self.substractRadio.isChecked(): c = a - b
        elif self.mulRadio.isChecked(): c = a * b
        else: c = a / b
        index = str(self.resultLabel.text()).index(':')
        s = str(self.resultLabel.text())[:index+1]
        self.resultLabel.setText('%s %.2f %s' % (s, c,
'</b>'))
if __name__ == '__main__':
    a = QApplication(sys.argv)
    form = MainForm()
    form.show()
    a.exec_()
```

Sedangkan metode yang sering digunakan untuk Kelas QCheckBox adalah sebagai berikut.

Metode	Fungsi
isChecked()	Memberikan nilai true jika control check box button aktof
seticon()	Menampilkan gambar icon pada kontrol check box
setText()	Memasukkan teks ke dalam control
setChecked()	Mengaktifkan control check box

Tuliskan kode program berikut pada computer anda untuk contoh implementasi dari check box.

```
##################################################################
# Nama file: DemoQCheckBox.py
##################################################################
```

```
import sys
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
```

```
class MainForm(QWidget):
    def __init__(self):
        super().__init__()
        self.setupUi()
    def setupUi(self):
        self.resize(300, 100)
        self.move(300, 300)
        self.setWindowTitle('Demo QCheckBox')
        self.label = QLabel()
        self.label.setText('Tentukan pilihan Anda:')
        self.perlCheck = QCheckBox()
        self.perlCheck.setText('Perl')
        self.pythonCheck = QCheckBox()
        self.pythonCheck.setText('Python')
        self.rubyCheck = QCheckBox()
        self.rubyCheck.setText('Ruby')
        self.phpCheck = QCheckBox()
        self.phpCheck.setText('PHP')
        hbox1 = QHBoxLayout()
        hbox1.addWidget(self.perlCheck)
        hbox1.addWidget(self.pythonCheck)
        hbox1.addWidget(self.rubyCheck)
        hbox1.addWidget(self.phpCheck)
        self.okButton = QPushButton('&OK')
        self.exitButton = QPushButton('Keluar')
        hbox2 = QHBoxLayout()
        hbox2.addStretch()
        hbox2.addWidget(self.okButton)
        hbox2.addWidget(self.exitButton)
        layout = QVBoxLayout()
```

```
        layout.addWidget(self.label)
        layout.addLayout(hbox1)
        horizontalLine = QFrame();
        horizontalLine.setFrameShape(QFrame.HLine)
        horizontalLine.setFrameShadow(QFrame.Sunken)
        layout.addWidget(horizontalLine)
        layout.addLayout(hbox2)
        layout.addStretch()
        self.setLayout(layout)
        self.okButton.clicked.connect(self.okButtonClick)
        self.exitButton.clicked.connect(self.close)
    def okButtonClick(self):
        choices = []
        if self.perlCheck.isChecked():
choices.append('Perl')
        if self.pythonCheck.isChecked():
choices.append('Python')
        if self.rubyCheck.isChecked():
choices.append('Ruby')
        if self.phpCheck.isChecked(): choices.append('PHP')
        QMessageBox.information(self, 'Informasi',
repr(choices))
if __name__ == '__main__':
    a = QApplication(sys.argv)
    form = MainForm()
    form.show()
    a.exec_()
```

32. KELAS QCOMBOBOX DAN QFONTCOMBOBOX

Kelas QComboBox digunakan untuk menampilkan combo box pada GUI. Daftar metode yang sering digunakan dalam kelasQComboBoxadalah sebagai berikut.

Metode	Fungsi
setItemText()	Mengubah text untuk pilihan dalam control QComboBox
removeItem()	Menghapus item dalam kontrolQComboBox
Clear()	Menghapus semua item dalam QComboBox
currentText()	Mendapoatkan text dalam item yang sedang aktif
setCurrentIndex()	Mengaktifkan item pada control ke indeks tertentu
Count()	Mendapatkan jumlah item dalam control QComboBox
setMaxCount()	Menentukan jumlah item maksimum yang dapat dimasukkan dalam kontrolQComboBox
setEditable()	Menentukan text pada kontrolQComboBox dapat dirubah atau tidak
addItem()	Menambah satu item ke dalam control QComboBox
addItems()	Menambahkan beberapa item ke dalam control QComboBox
itemText()	Mendapatkan text item yang berada pada indeks tertentu
currentIndex()	Mendapatkan indeks dari item yang sedang aktif

Untuk implementasi kelas dan control QComboBox tulislah kode program berikut pada computer anda!

```
##################################################################
# Nama file: DemoQComboBox.py
##################################################################
```

```
import sys
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
```

```
    def __init__(self):
        super().__init__()
        self.setupUi()
    def setupUi(self):
        self.resize(300, 100)
        self.move(300, 300)
        self.setWindowTitle('Demo QComboBox')
        self.combo = QComboBox()
        for i in range(1,11):
            self.combo.addItem('Item ke-%d' % i)
        self.getTextButton = QPushButton('Ambil Teks')
        layout = QVBoxLayout()
        layout.addWidget(self.combo)
        layout.addWidget(self.getTextButton)
        layout.addStretch()
        self.setLayout(layout)

self.getTextButton.clicked.connect(self.getTextButtonClic
k)
    def getTextButtonClick(self):
        QMessageBox.information(self, 'Informasi',
            'Anda memilih: ' + self.combo.currentText())
if __name__ == '__main__':
    a = QApplication(sys.argv)
    form = MainForm()
    form.show()
    a.exec_()
```

Sedangkan kelas QFontComboBox merupakan turunan dari kelas QComboBox, yang digunakan secara khusus untuk menampilkan jenis font yang ada. Contoh implementasi kelas QTextComboBox adalah sebagai berikut!

```
#########################################################
# Nama file: DemoQFontComboBox.py
#########################################################
import sys
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
   def __init__(self):
      super().__init__()
      self.setupUi()
   def setupUi(self):
      self.resize(300, 100)
      self.move(300, 300)
      self.setWindowTitle('Demo QFontComboBox')
      self.fontCombo = QFontComboBox()
      self.fontCombo.setEditable(False)
      self.label = QLabel('Contoh Teks')
      self.label.setFont(QFont('DejaVu Sans',18))
      layout = QVBoxLayout()
      layout.addWidget(self.fontCombo)
      layout.addWidget(self.label)
      layout.addStretch()
      self.setLayout(layout)

self.fontCombo.activated.connect(self.fontComboActivated)
   def fontComboActivated(self):
      self.label.setFont(
        QFont(self.fontCombo.currentText(), 18)
      )
if __name__ == '__main__':
```

```
    a = QApplication(sys.argv)
    form = MainForm()
    form.show()
    a.exec_()
```

33. KELAS QSPINBOX

Kelas QSpinBox digunakan untuk memasukkan tipe bilangan bulat. Kelas ini memungkinkan pengguna memasukkan inlai secara manual ataupuan dengan menggunakan tombol yang tersedia pad akelas control ini. Beberapa metode yang dapat digunakan pad akelas QSpinBox adalah sebagai berikut.

Metode	Fungsi
setMinimum()	Menentukan nilai minimum yang dapt dityampilkan
setMaximum()	Menentukan nilai maksimum yang dpat ditampilkan
value()	Mengambil nilai yang sedang aktif pada kontrolQSpinBox
setValue()	Mengubah nilai yang sedang akti pad akontrol QSpinBox
setRange()	Menentukan range nilai yang dapat ditampilkamn p[ada control QSpinBox
setSingleStep()	Menentukan selisih dari nilai sebelumnya atau nilai berikutnya

Tuliskan kode program berikut pada konputer anda untuk membuat kelas QSpinBox.

```
##############################################################
# Nama file: DemoQSpinBox.py
##############################################################
```

```
import sys
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
```

```
    def __init__(self):
        super().__init__()
        self.setupUi()
    def setupUi(self):
        self.resize(300, 100)
        self.move(300, 300)
        self.setWindowTitle('Demo QSpinBox')
        self.fontLabel = QLabel('Jenis huruf')
        self.fontCombo = QFontComboBox()
        self.fontCombo.setEditable(False)
        self.sizeLabel = QLabel('Ukuran huruf')
        self.sizeSpinBox = QSpinBox()
        self.sizeSpinBox.setRange(8,20)
        self.sizeSpinBox.setSingleStep(1)
        self.sizeSpinBox.setValue(18)
        self.sampleLabel = QLabel('Contoh Teks')
        self.sampleLabel.setFont(QFont('DejaVu Sans',18))
        layout = QGridLayout()
        layout.addWidget(self.fontLabel, 0, 0)
        layout.addWidget(self.fontCombo, 0, 1)
        layout.addWidget(self.sizeLabel, 1, 0)
        layout.addWidget(self.sizeSpinBox, 1, 1)
        layout.addWidget(self.sampleLabel, 2, 0, 1, 2)
        #layout.addStretch()
        self.setLayout(layout)
        self.fontCombo.activated.connect(self.changeFont)

self.sizeSpinBox.valueChanged.connect(self.changeFont)
    def changeFont(self):
        self.sampleLabel.setFont(
```

```
            QFont(self.fontCombo.currentText(),
    self.sizeSpinBox.value())
        )
    if __name__ == '__main__':
        a = QApplication(sys.argv)
        form = MainForm()
        form.show()
        a.exec_()
```

34. KELAS QDATEEDIT, QTIMEEDIT, DAN QDATETIMEEDIT

Ketiga kelas tersebut merupakan kela control yang digunakan untuk memsaukkan data tanggal dan waktu. QDateEdit merupakan tanggal, QTimeEdit merupakan waktu, dan jika anda ingin memasukkan format tanggal dan waktu maka gunakanlah QDateTimeEdit. Untuk implementasi kelas QDateTimeEdit tuliskan kode program berikut ini.

```
##############################################################
# Nama file: DemoQDateTimeEdit.py
##############################################################
```

```
    import sys
    from PyQt5.QtGui import *
    from PyQt5.QtCore import *
    from PyQt5.QtWidgets import *
    class MainForm(QWidget):
        def __init__(self):
            super().__init__()
            self.setupUi()
        def setupUi(self):
            self.resize(400, 100)
            self.move(300, 300)
            self.setWindowTitle('Demo QDateTimeEdit')
            self.dateLabel = QLabel('Tanggal')
            self.dateEdit = QDateEdit()
```

```
        self.dateEdit.setDisplayFormat('dddd dd/MM/yyyy')
        self.dateEdit.setDate(QDate.currentDate())
        self.timeLabel = QLabel('Waktu')
        self.timeEdit = QTimeEdit()
        self.timeEdit.setDisplayFormat('hh:mm')
        self.timeEdit.setTime(QTime.currentTime())
        self.dateTimeLabel = QLabel('Tanggal dan Waktu')
        self.dateTimeEdit = QDateTimeEdit()
        self.dateTimeEdit.setDisplayFormat('dddd dd/MM/yyyy
hh:mm')

self.dateTimeEdit.setDateTime(QDateTime.currentDateTime()
)
        self.okButton = QPushButton('&OK')
        hbox = QHBoxLayout()
        hbox.addStretch()
        hbox.addWidget(self.okButton)
        layout = QGridLayout()
        layout.addWidget(self.dateLabel, 0, 0)
        layout.addWidget(self.dateEdit, 0, 1)
        layout.addWidget(self.timeLabel, 1, 0)
        layout.addWidget(self.timeEdit, 1, 1)
        layout.addWidget(self.dateTimeLabel, 2, 0)
        layout.addWidget(self.dateTimeEdit, 2, 1)
        layout.addLayout(hbox, 3, 0, 1, 2)
        self.setLayout(layout)
        self.okButton.clicked.connect(self.okButtonClick)
    def okButtonClick(self):
        QMessageBox.information(self, 'Informasi',
            'Date: ' + self.dateEdit.date().toString() +
'\n' +
```

```
            'Time: ' + self.timeEdit.time().toString() +
'\n' +
            'Datetime: ' +
self.dateTimeEdit.dateTime().toString() + '\n')
if __name__ == '__main__':
    a = QApplication(sys.argv)
    form = MainForm()
    form.show()
    a.exec_()
```

Kelas QSpinBox digunakan untuk memasukkan tipe bilangan bulat. Kelas ini memungkinkan pengguna memasukkan inlai secara manual ataupuan dengan menggunakan tombol yang tersedia pad akelas control ini. Beberapa metode yang dapat digunakan pad akelas QSpinBox adalah sebagai berikut.

35. KELAS QCALENDARWIDGET

Kelas QCalendarWidget digunakan untuk menampilakn kelaender di dalam form. Secara default tanggal yang ditampilakan adalah tanggal aktof hari ini. Tetapi adna dapat menetukan sendiri tanggal aktof dengan menggunakan metode setSelecteddate(). Untuk contoh implementasi kelas QCalendarWidget tuliskan kode program berikut ke computer anda.

```
##########################################################
# Nama file: DemoQCalendarWidget.py
##########################################################
import sys
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
    def __init__(self):
        super().__init__()
        self.setupUi()
    def setupUi(self):
```

```
        self.resize(400, 100)
        self.move(300, 300)
        self.setWindowTitle('Demo QCalendarWidget')
        self.calendar = QCalendarWidget()
        self.calendar.setGridVisible(True)

self.calendar.setHorizontalHeaderFormat(QCalendarWidget.L
ongDayNames)
        self.shortNamesCheck = QCheckBox('Nama hari
pendek')
        self.dateEdit = QDateEdit()
        self.dateEdit.setDisplayFormat('dd/MM/yyyy')
        self.dateEdit.setDate(QDate.currentDate())
        self.setButton = QPushButton('Tentukan Tanggal')
        self.getButton = QPushButton('Ambil Tanggal')
        hbox = QHBoxLayout()
        hbox.addWidget(self.dateEdit)
        hbox.addWidget(self.setButton)
        hbox.addWidget(self.getButton)
        layout = QVBoxLayout()
        layout.addWidget(self.calendar)
        layout.addWidget(self.shortNamesCheck)
        layout.addLayout(hbox)
        self.setLayout(layout)

self.shortNamesCheck.clicked.connect(self.shortNamesCheck
Click)
 self.setButton.clicked.connect(self.setButtonClick)
 self.getButton.clicked.connect(self.getButtonClick)
   def shortNamesCheckClick(self):
      if self.shortNamesCheck.isChecked():
```

```
self.calendar.setHorizontalHeaderFormat(QCalendarWidget.S
hortDayNames)
      else:

self.calendar.setHorizontalHeaderFormat(QCalendarWidget.L
ongDayNames)
   def setButtonClick(self):
 self.calendar.setSelectedDate(self.dateEdit.date())
   def getButtonClick(self):
      QMessageBox.information(self, 'Informasi',
         'Tanggal aktif: ' +
self.calendar.selectedDate().toString())
if __name__ == '__main__':
   a = QApplication(sys.argv)
   form = MainForm()
   form.show()
   a.exec_()
```

36. KELAS QSLIDER DAN QLCDNUMBER

Kelas Qslider digunakan untuk menampilkan control slider di dlaam form. Dengan control slider ini, anda dapat mengirimkan nilai ke dalam program secara lebih interaktif melalui slider. Metode yang paling sering digunakan dalam kelasQslider adalah setValue() yang digunakan untuk menentukan nilai dari control tersebut. Selain metode setValue(), metode lain yang digunakan dalam kelas QSlider aldah setMinimum() dan setMaximum() yang berfungsi untuk menentukan nilai maksimum dan anilai minimum dari slider.

Sedangkan kelasQLCDNumber digunakan untuk menampilkan angka ke dalam form dalam bentuk format angka LCD. Tuliskan contoh implementasi dari kelasQSlider dan QLCDNumber berukit ke dalam computer anda.

```
#############################################################
# Nama file: DemoQSlider.py
```

```
##################################################################
    import sys
    from PyQt5.QtGui import *
    from PyQt5.QtCore import *
    from PyQt5.QtWidgets import *
    class MainForm(QWidget):
        def __init__(self):
            super().__init__()
            self.setupUi()
        def setupUi(self):
            self.resize(400, 100)
            self.move(300, 300)
            self.setWindowTitle('Demo QSlider dan QLCDNumber')
            self.slider = QSlider(Qt.Horizontal)
            self.slider.setMinimum(-1)
            self.slider.setMaximum(101)
            self.slider.setValue(45)
            self.lcd = QLCDNumber()
            self.lcd.setDigitCount(3)
            self.lcd.display(45)
            layout = QVBoxLayout()
            layout.addWidget(self.slider)
            layout.addWidget(self.lcd)
            self.setLayout(layout)
            self.slider.sliderMoved.connect(self.sliderMoved)
        def sliderMoved(self):
            self.lcd.display(str(self.slider.value()))
    if __name__ == '__main__':
        a = QApplication(sys.argv)
        form = MainForm()
        form.show()
```

```
        a.exec_()
```

37. KELAS QLISTWIDGET

KelasQListWidget digunakan untuk membuat control listbox. Metode yang sering digunakan dalam kelas QListWidget adalah sebagai berikut.

Metode	Fungsi
addItem()	Menambahkan item ke dalam control
addItems()	Menambahkan leih dari satu item ke dalam control
clear()	Menghapus semua item dalam control
count()	Menghitung jumlah item di dalam control
currentItem()	Mendapatkan item yang sedang dipilih
currentRow()	Mendapatkan indeks dari item yangs edang dipilih
Item()	Mendapatkan ite berdasarkan indeks yang dilewatkan sebagai parameternya
setrCurrentItem()	Menentukan atau mengubah nilai pada item yang sedang aktif
setSortingEnabled()	Membuat daftar item di dalam control dapat diurutkan
sortItems()	Mengurutkan daftar item di dalam control
takeItem()	Mengambil atau menghapus item pada indeks yang dijadikan sebagai parameternya

Jika anda ingin memindahkan item yang ada pada satu list ke list lainnya, metode currentRow() dapat digunakan untuk mendapatkan ideks dair item yang akan dipindahkan. Lalu memnggunakan. Metode currentItem() untuk mendapatkan isi dari item yang akan dipindahkan, setelah itu memindahkan item yang adapa pada list awal dengan metyode takeItem(). Tuliskan kode program berikut untuk implementasi dari kelas QListWidget.

```
##########################################################
# Nama file: DemoQListWidget.py
##########################################################
```

```
    import sys
```

```
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
   def __init__(self):
      super().__init__()
      self.setupUi()
   def setupUi(self):
      self.resize(400, 300)
      self.move(300, 300)
      self.setWindowTitle('Demo QListWidget')
      self.label = QLabel('&Elemen baru')
      self.itemEdit = QLineEdit()
      self.label.setBuddy(self.itemEdit)
      self.addItemButton = QPushButton('Tambah')
      hbox1 = QHBoxLayout()
      hbox1.addWidget(self.itemEdit)
      hbox1.addWidget(self.addItemButton)
      hbox1.addStretch()
      vbox1 = QVBoxLayout()
      vbox1.addWidget(self.label)
      vbox1.addLayout(hbox1)
      self.list1 = QListWidget()
      self.moveRightButton = QPushButton('>')
      self.moveRightAllButton = QPushButton('>>')
      self.moveLeftButton = QPushButton('<')
      self.moveLeftAllButton = QPushButton('<<')
      vbox2 = QVBoxLayout()
      vbox2.addWidget(self.moveRightButton)
      vbox2.addWidget(self.moveRightAllButton)
```

```
        vbox2.addWidget(self.moveLeftButton)
        vbox2.addWidget(self.moveLeftAllButton)
        vbox2.addStretch()
        self.list2 = QListWidget()
        hbox2 = QHBoxLayout()
        hbox2.addWidget(self.list1)
        hbox2.addLayout(vbox2)
        hbox2.addWidget(self.list2)
        layout = QVBoxLayout()
        layout.addLayout(vbox1)
        layout.addLayout(hbox2)
        self.setLayout(layout)

self.addItemButton.clicked.connect(self.addItemButtonClic
k)

self.moveRightButton.clicked.connect(self.moveRightButton
Click)

self.moveRightAllButton.clicked.connect(self.moveRightAll
ButtonClick)

self.moveLeftButton.clicked.connect(self.moveLeftButtonCl
ick)

self.moveLeftAllButton.clicked.connect(self.moveLeftAllBu
ttonClick)
   def addItemButtonClick(self):
       if len(self.itemEdit.text()) == 0: return
       item = self.itemEdit.text()
       self.list1.addItem(item)
       self.itemEdit.clear()
       self.itemEdit.setFocus()
```

```
    def moveRightButtonClick(self):
        if self.list1.currentRow() < 0: return
        self.list2.addItem(self.list1.currentItem().text())
        self.list1.takeItem(self.list1.currentRow())

    def moveRightAllButtonClick(self):
        for index in range(self.list1.count()):
self.list2.addItem(self.list1.item(index).text())
        self.list1.clear()

    def moveLeftButtonClick(self):
        if self.list2.currentRow() < 0: return
        self.list1.addItem(self.list2.currentItem().text())
        self.list2.takeItem(self.list2.currentRow())

    def moveLeftAllButtonClick(self):
        for index in range(self.list2.count()):
self.list1.addItem(self.list2.item(index).text())
        self.list2.clear()
if __name__ == '__main__':
    a = QApplication(sys.argv)
    form = MainForm()
    form.show()
    a.exec_()
```

38. KELAS QPROGRESSBAR

KelasQProgressBar digunakan untuk menapilkkan control progress bar pda form. Nilai progress bar ini dapat disesuaikan nilai maksimum dan minimumnya. Tuliskan contoh kode program berikut untuk contoh implementasi dari progress bar.

```
#############################################################
# Nama file: DemoQProgressBar.py
#############################################################
import sys
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
    def __init__(self):
        super().__init__()
        self.setupUi()
    def setupUi(self):
        self.resize(400, 400)
        self.move(300, 300)
        self.setWindowTitle('Demo QProgressBar')
        self.label1 = QLabel('Bilangan Ganjil')
        self.list1 = QListWidget()
        vbox1 = QVBoxLayout()
        vbox1.addWidget(self.label1)
        vbox1.addWidget(self.list1)
        self.label2 = QLabel('Bilangan Genap')
        self.list2 = QListWidget()
        vbox2 = QVBoxLayout()
        vbox2.addWidget(self.label2)
        vbox2.addWidget(self.list2)
        hbox = QHBoxLayout()
        hbox.addLayout(vbox1)
        hbox.addLayout(vbox2)
        self.progress = QProgressBar()
        self.progress.setMinimum(0)
        self.progress.setMaximum(10000)
```

```
        self.progress.setValue(0)
        self.startButton = QPushButton('Mulai...')
        layout = QVBoxLayout()
        layout.addLayout(hbox)
        layout.addWidget(self.progress)
        layout.addWidget(self.startButton)
        self.setLayout(layout)

    self.startButton.clicked.connect(self.startButtonClick)
    def startButtonClick(self):
        self.progress.setValue(0)
        for i in range(0,1000):
            QApplication.processEvents()
            if i % 2 == 1: self.list1.addItem(str(i))
            else: self.list2.addItem(str(i))
     self.progress.setValue(self.progress.value()+1)
if __name__ == '__main__':
    a = QApplication(sys.argv)
    form = MainForm()
    form.show()
    a.exec_()
```

39. KELAS QTREEWIDGET

Kelas QTreeWidget digunakan untuk mernmpilkan daftar item yang disusun berdasarkan hiirarkinya. Signal yang digunakan dalam kelas QTreeWidget adalah itemClicked(). Tuliskan kode program berikut pada computer anda untuk demo dari kelas QTreeWidget.

```
##################################################
# Nama file: DemoQTreeWidget.py
##################################################
    import sys
```

```
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
   def __init__(self):
      super().__init__()
      self.setupUi()
   def setupUi(self):
      self.resize(700, 300)
      self.move(300, 300)
      self.setWindowTitle('Demo QTreeWidget')
      self.tree = QTreeWidget()
      self.tree.setColumnCount(3)
      columnHeaders = ['Judul Buku', 'Penulis',
'Penerbit']
      self.tree.setHeaderLabels(columnHeaders)
      parent1 = self.addTopLevel('Python')
      self.addChild(parent1,
        'Python 3 Object Oriented Programming',
        'Dusty Phillips',
        'PACKT Publishing')
      self.addChild(parent1,
        'Numerical Python',
        'Robert Johansson',
        'Apress')
      self.addChild(parent1,
        'A Primer Scientific Programming with Python',
        'Hans Peter Langtangen',
        'Springer')
      parent2 = self.addTopLevel('Ruby')
      self.addChild(parent2,
```

```
            'Beginning Ruby',
            'Peter Cooper',
            'Apress')
        self.addChild(parent2,
            'Ruby Under a Microscope',
            'Pat Shaughnessy',
            'No Starch Press')
        self.lineEdit = QLineEdit()
        layout = QVBoxLayout()
        layout.addWidget(self.tree)
        layout.addWidget(self.lineEdit)
        self.setLayout(layout)
     self.tree.itemClicked.connect(self.treeItemClick)
def treeItemClick(self):
        item = self.tree.currentItem()
        self.lineEdit.setText(item.text(0))
    def addTopLevel(self, category):
        item = QTreeWidgetItem()
        item.setText(0, category)
        self.tree.addTopLevelItem(item)
        return item
    def addChild(self, parent, title, author,
publisher):
        item = QTreeWidgetItem(parent)
        item.setText(0, title)
        item.setText(1, author)
        item.setText(2, publisher)
        parent.addChild(item)
        return item
if __name__ == '__main__':
    a = QApplication(sys.argv)
```

```
    form = MainForm()
    form.show()
    a.exec_()
```

40. KELAS QTABLEWIDGET

Keals QTableWidget digunakan untuk menampilkan daftar item dalam bentuk tabular. Metode yang sering digunakan dalam QTableWidget adalah sebagai berikut

Metode	Fungsi
columnCount()	Mendapatkan jumlah kolom pada control
currentColumn()	Mendapatkan nomor kolom aktif pada control
currentItem()	Mendapatkan item aktif pada control
currentRow()	Mendpatkan nomor baris aktif pada control
setColumnCount()	Menentukan jumlah kolompada control
setCurrentItem()	Mengubah item aktif pada control
setCurrentCell()	Mengaktifkan item tertentu berdasarkan nomor baris dan klom yang dilewatkan sebagai parameternya
setHorizontalHeaderLabels()	Menentukan judul kolom yang akan ditampilkan pada control
setItem()	Menentukan nilai item pada baris dan kolom tertentu
setRowCount()	Menentukan jumlah baris pada kontrol
takeItem()	Mengambil atau menghapus item pada control berdasarkan baris dan kolomnya

Tuliskan kode program berikut pada computer anda untuk membuat table widget.

```
##################################################################
# Nama file: DemoQTableWidget.py
##################################################################
    import sys
```

```
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
class MainForm(QWidget):
   def __init__(self):
      super().__init__()
      self.setupUi()
   def setupUi(self):
      self.resize(700, 300)
      self.move(300, 300)
      self.setWindowTitle('Demo QTableWidget')
      self.table = QTableWidget()
      self.table.setRowCount(5)
      self.table.setColumnCount(3)
      columnHeaders = ['Judul Buku', 'Penulis',
'Penerbit']
 self.table.setHorizontalHeaderLabels(columnHeaders)
      self.addRow(0,
        ['Python 3 Object Oriented Programming',
        'Dusty Phillips',
        'PACKT Publishing'])
      self.addRow(1,
        ['Numerical Python',
        'Robert Johansson',
        'Apress'])
      self.addRow(2,
        ['A Primer Scientific Programming with Python',
        'Hans Peter Langtangen',
        'Springer'])
      self.addRow(3,
        ['Beginning Ruby',
```

```
            'Peter Cooper',
            'Apress'])
        self.addRow(4,
            ['Ruby Under a Microscope',
            'Pat Shaughnessy',
            'No Starch Press'])
        self.lineEdit = QLineEdit()
        layout = QVBoxLayout()
        layout.addWidget(self.table)
        layout.addWidget(self.lineEdit)
        self.setLayout(layout)
 self.table.itemClicked.connect(self.tableItemClick)
def tableItemClick(self):
        item = self.table.currentItem()
        self.lineEdit.setText(item.text() +
            ' [baris: %d, kolom: %d]' %
            (self.table.currentRow(),
self.table.currentColumn()))
    def addRow(self, row, itemLabels=[]):
        for i in range(0,3):
            item = QTableWidgetItem()
            item.setText(itemLabels[i])
            self.table.setItem(row, i, item)
if __name__ == '__main__':
    a = QApplication(sys.argv)
    form = MainForm()
    form.show()
    a.exec_()
```