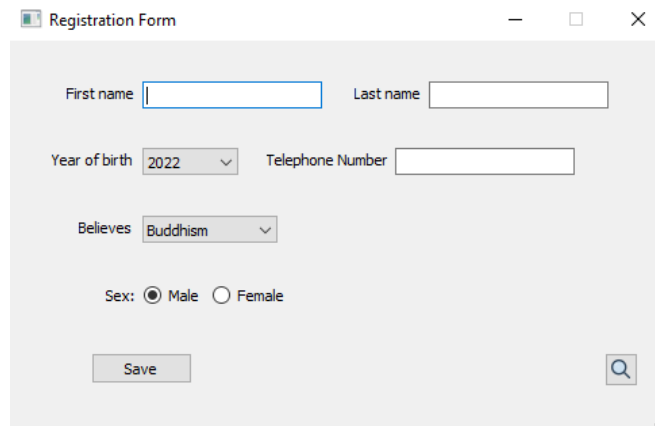
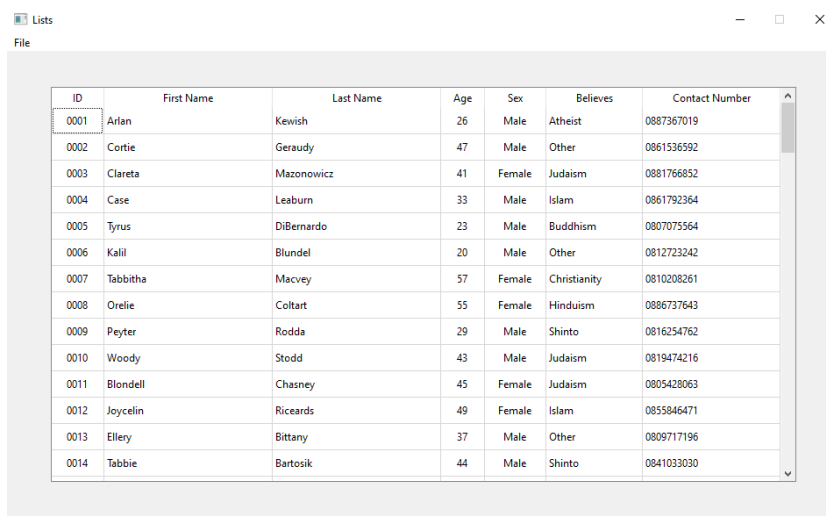


Registration Form เป็นแอปพลิเคชันที่มีหน้าที่รับข้อมูลที่ใช้กรอกเข้ามาเพื่อเขียนเก็บไว้ในไฟล์ text ใน directory ของตัวแอปพลิเคชัน



A screenshot of a 'Registration Form' window. It contains several input fields: 'First name' and 'Last name' (text boxes), 'Year of birth' (a dropdown menu showing '2022'), 'Telephone Number' (a text box), 'Believes' (a dropdown menu showing 'Buddhism'), and 'Sex' (radio buttons for 'Male' and 'Female', with 'Male' selected). At the bottom, there is a 'Save' button and a magnifying glass icon.

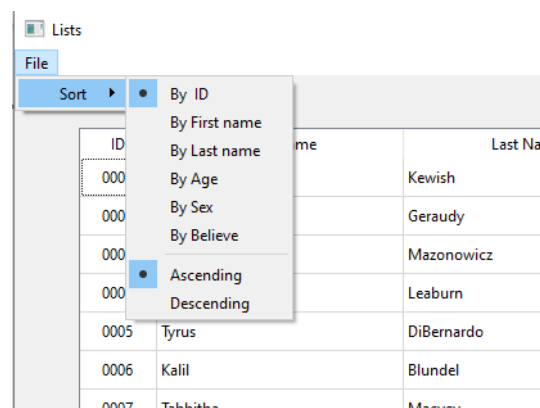
โดยในแอปพลิเคชันจะมีปุ่ม  ที่สามารถเรียกดูข้อมูลที่ถูกเก็บไว้ในไฟล์ได้ โดยไฟล์จะถูกอ่านแล้วแปลงเป็นตารางก่อนขึ้นแสดงผล



A screenshot of a 'Lists' window displaying a table of data. The table has 7 columns: ID, First Name, Last Name, Age, Sex, Believes, and Contact Number. The data is sorted by ID in ascending order.

ID	First Name	Last Name	Age	Sex	Believes	Contact Number
0001	Arlan	Kewish	26	Male	Atheist	0887367019
0002	Cortie	Geraudy	47	Male	Other	0861536592
0003	Clareta	Mazonowicz	41	Female	Judaism	0881766852
0004	Case	Leaburn	33	Male	Islam	0861792364
0005	Tyrus	DiBernardo	23	Male	Buddhism	0807075564
0006	Kalil	Blundel	20	Male	Other	0812723242
0007	Tabbitha	Macvey	57	Female	Christianity	0810208261
0008	Orelie	Coltart	55	Female	Hinduism	0886737643
0009	Peyter	Rodda	29	Male	Shinto	0816254762
0010	Woody	Stodd	43	Male	Judaism	0819474216
0011	Blondell	Chasney	45	Female	Judaism	0805428063
0012	Joycelin	Riceards	49	Female	Islam	0855846471
0013	Ellery	Bittany	37	Male	Other	0809717196
0014	Tabbie	Bartosik	44	Male	Shinto	0841033030

โดยในหน้าต่างแสดงผลนี้ จะมี menu bar ที่ชื่อว่า File ซึ่งภายใน File นี้จะมีคำสั่ง Sort เพื่อเรียงข้อมูลในตารางตามไทป์ของข้อมูลที่ใช้เลือก รวมทั้งสามารถเลือกให้ Sort แบบกลับหลังได้ด้วย



MainWindow

คำสั่งเรียกใช้งานไฟล์หน้าต่างแสดงผล ListWindow เพื่อให้สามารถเขียนฟังก์ชันแสดงผลหน้าต่างได้เมื่อทำการกดปุ่มที่กำหนด รวมทั้งทำให้สามารถใช้ตัวแปรกับฟังก์ชันร่วมกันได้ด้วย

```
1 from PyQt5 import QtCore, QtGui, QtWidgets
2 from ListWindow import Ui_listWindow #call the ListWindow from other file to summon it with openWindow function
3 from datetime import datetime
4 from pathlib import Path
5 import winsound
6
7 class Ui_mainWindow(object):
8
9     l = Ui_listWindow() #placeholder for self parameter
10    fileName = Ui_listWindow.fileName #borrow the variable from ListWindow
11    filePath = Ui_listWindow.filePath
12
13    def isFileExists(self): #If the file didn't exist, create new file
14        if (Ui_listWindow.findFile(self.l, self.fileName, self.filePath) == None): #borrow the function from ListWindow
15            file = open(self.fileName, "x")
16        else:
17            return
```

ตัวแปร fileName และ filePath ที่ถูกเรียกมาจากหน้าต่าง ListWindow โดย fileName มีหน้าที่เก็บค่าเป็นชื่อไฟล์ที่จะถูกอ่านและเขียนเพื่อบันทึกข้อมูลที่ได้รับจากหน้าต่าง Main ส่วน filePath เป็นตัวแปรที่เอาไว้เก็บ directory ของไฟล์ใน fileName

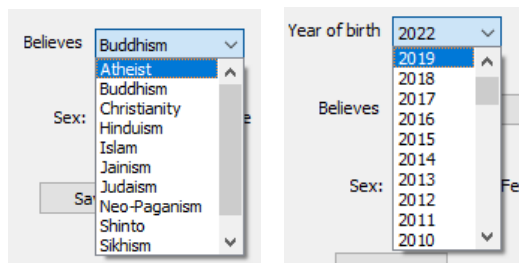
ฟังก์ชัน isFileExists() มีหน้าที่ตรวจสอบว่ามีไฟล์ข้อมูลพร้อมใช้หรือไม่ โดยใช้ฟังก์ชัน findFile จากไฟล์ ListWindow ถ้าไม่มีให้สร้างไฟล์ขึ้นโดยใช้ตัวแปร fileName ในการตั้งชื่อไฟล์

นี่คือส่วนของโปรแกรมในไฟล์ ListWindow

```
6 class Ui_listWindow(object):
7
8     fileName = "peopleLists.txt"
9     filePath = Path(__file__).parent.absolute() #Find the directory of current file source=StackOverflow
10
11    def findFile(self, name, path):
12        for root, dirs, files in os.walk(path): #Scan for file in given path return NoneType if not found source=StackOverflow
13            if name in files:
14                return os.path.join(root, name)
```

ตัวแปร believes และ possibleYear มีหน้าที่เก็บข้อมูลเป็นลิสต์ไว้เพื่อแสดงผลใน widget

dropdown box ในหน้าต่าง Main โดย believes จะเก็บเป็นชื่อศาสนาและความเชื่อต่าง ๆ ส่วน possibleYear จะเก็บเลขปีค.ศ.ตั้งแต่ปี 1900 จนถึงปัจจุบัน



```
19 believes = ["Atheist", "Buddhism", "Christianity", "Hinduism", "Islam", "Jainism", "Judaism", "Neo-Paganism", "Shinto", "Sikhism", "Other"]
20 possibleYear = []
21
22 for i in range(datetime.today().year, 1900, -1): #find all possible birthYear to add in combobox
23     possibleYear.append(str(i))
```

ฟังก์ชัน loadUp() มีไว้เพื่อเขียนข้อมูลลงไปไฟล์ทันทีที่เปิดแอปพลิเคชันขึ้นมา โดยฟังก์ชัน loadUp() จะดึงข้อมูลจากไฟล์ (ที่ผ่านการรันดีแล้วว่ามีไฟล์อยู่แน่นอน ด้วยฟังก์ชัน isFileExists()) มาอ่านเพื่อตรวจสอบว่าไฟล์นั้นว่างเปล่าหรือไม่ ถ้าไฟล์ว่างเปล่าก็ไม่จำเป็นต้องปรับแต่งอะไร แต่ถ้าหากไฟล์มีข้อมูลอยู่ก่อนแล้ว ฟังก์ชัน loadUp() จะทำการเรียกใช้ฟังก์ชัน deFrag() เพื่อให้ deFrag() ทำการลบสมาชิก-list ที่เป็นช่องว่างทิ้งไป เพื่อว่าในไฟล์ข้อมูลจะมีการเว้นข้ามบรรทัด ซึ่งจะทำให้เกิดสมาชิก-list ที่มีค่าว่างขึ้นได้

```
def loadUp(self):          #Clear out the empty line in file and rewrite the file
    fileRead = open(UI_listWindow.findFile(self.l, self.fileName, self.filePath), "r")
    data = fileRead.read()
    fileRead.close()
    if (data == ""):
        return
    people = data.splitlines()
    actualPeople = self.deFrag(people, "")
    fileWrite = open(UI_listWindow.findFile(self.l, self.fileName, self.filePath), "w")
    for i in actualPeople:
        person = i.split()
        for info in person:
            fileWrite.write(info)
            if not info == person[len(person) -1]:
                fileWrite.write(" ")
        fileWrite.write("\n")
    fileWrite.close()

def deFrag(self, lists, val):    #Filter out certain value in all of the set
    return [value for value in lists if value != val]
```

หลังจากจัดการกับความเสี่ยงที่จะมี-list-ว่างปะปนอยู่ในข้อมูลแล้ว ฟังก์ชัน loadUp() จะทำการเขียนข้อมูลที่ผ่านการตรวจสอบและจัดระเบียบแล้วลงไปในไฟล์ข้อมูล เพื่อให้โปรแกรมทำงานได้อย่างถูกต้อง

ฟังก์ชัน getID() มีไว้เพื่อเปิดอ่านไฟล์ด้วยฟังก์ชัน readFile จากหน้า ListWindow แล้วตรวจสอบดูว่า เลข ID ของข้อมูลสุดท้ายในไฟล์เป็นเลขอะไร เพื่อที่จะ return ไปให้ฟังก์ชันที่ใช้ในการรวบรวมข้อมูลนำไปอ้างอิงเพื่อกำหนด ID ให้ข้อมูลใหม่ที่ถูกป้อนเข้ามา

```
46     def getID(self):          #read file to continue from the latest ID
47         people = UI_listWindow.readFile(self.l)
48         if len(people) == 0:
49             return 0
50         for i in people:
51             person = i.split()
52             return int(person[0])
```

เมื่อกดปุ่ม Save จะเป็นการเรียกฟังก์ชัน getData() ขึ้นมาพร้อมกับส่งเสียง ฟังก์ชัน getData() เป็นฟังก์ชันที่จะส่งให้นำค่าทั้งหมดจากแบบฟอร์มในหน้า Main มาเก็บเป็นตัวแปร จากนั้นจะทำการเช็คข้อมูลด้วยการเรียกใช้ฟังก์ชัน checkData() หากข้อมูลที่เก็บมาไม่มีปัญหาอะไร จะทำการเอาตัวแปรมา format ให้พร้อมสำหรับการเขียนลงไฟล์ จากนั้นจึงเรียกใช้ฟังก์ชัน register() เพื่อทำการเขียนข้อมูลลงไปในไฟล์

```
self.saveButton.clicked.connect(lambda: [self.getData(), winsound.MessageBeep(winsound.MB_OK)])
```

```
57 def getData(self): #retrive data from input boxes
58     id = "{0:04d}".format(self.getID() + 1) #count list of people and add 1 to assign current person ID and format it to 4digits number
59     name = self.firstName.text()
60     lastName = self.lastName.text()
61     birthYear = self.birthYear.currentText()
62     believes = self.believesBox.currentText()
63     if (self.maleButton.isChecked()):
64         sex = "Male"
65     elif (self.femaleButton.isChecked()):
66         sex = "Female"
67     tel = self.tel.text()
68     if (self.checkData(name, lastName, birthYear, tel)):
69         age = datetime.today().year - (int(birthYear))
70         person = str(id + " " + name.capitalize() + " " + lastName.capitalize() + " " + str(age) + " " + sex + " " + believes + " " + tel)
71         self.register(person)
```

ฟังก์ชัน checkData() จะทำการเช็คข้อมูลที่รับมาว่ามีปัญหาหรือไม่ ด้วยการขึ้นข้อความเตือนรวมถึงจะส่งค่าเป็น False กลับไปทันทีถ้ากรอกข้อมูลไม่ครบทุกช่อง หรือมีตัวเลขหรือสัญลักษณ์พิเศษปนอยู่ในชื่อหรือนามสกุล หรือมีตัวอักษรอยู่ในเบอร์โทรศัพท์ หรือใส่เบอร์โทรศัพท์ไม่ครบ 10 ตัว แต่หากไม่มีปัญหาอะไรก็จะคืนเป็นค่า True เพื่อให้ฟังก์ชัน getData() นำข้อมูลไป format

```
def checkData(self, name, lastName, birthYear, tel):
    if (len(name) == 0 or len(lastName) == 0 or len(birthYear) == 0 or len(tel) == 0): #force user to only fill in all the boxes
        self.resultText.setText("You must fill in all the information!")
        return False
    if (not name.isalpha() or not lastName.isalpha()): #force user to only fill in letters
        self.resultText.setText("The name must contain only letters!")
        return False
    if (not tel.isnumeric()): #force user to only fill in number
        self.resultText.setText("Telephone number must contain only numbers!")
        return False
    if (not len(tel) == 10): #force user to only fill exactly 10 digits of telephone number
        self.resultText.setText("Telephone number must have exactly 10 digits!")
        return False
    return True
```

ฟังก์ชัน register() จะรับ string ข้อมูลที่ขึ้นกันไว้ด้วย “ “ มาเขียนลงไฟล์ แล้วแสดงข้อความแจ้งบอกว่าการบันทึกไฟล์สำเร็จ แล้วจึงเรียกฟังก์ชัน clearBoxes() เพื่อทำการล้างข้อมูลในกล่องข้อความในแบบฟอร์มทั้งหมด เพื่อให้พร้อมรับข้อมูลใหม่

```
def register(self, person):    #write data on file
    file = open(UI_listWindow.findFile(self.l, self.fileName, self.filePath),"a")
    for i in person:
        file.write(str(i))
    file.write("\n")
    file.close()
    self.resultText.setText("Data saved successfully.")
    self.clearBoxes()
```

```
def clearBoxes(self):    #clear input boxes
    self.firstName.clear()
    self.lastName.clear()
    self.tel.clear()
```

ListWindow

ฟังก์ชัน loadData() เป็นฟังก์ชันที่จะทำการนำข้อมูลจากในไฟล์มาบรรจุในตาราง โดยเริ่มจากการเรียกใช้ฟังก์ชัน processData() โดยจะเรียกฟังก์ชัน readFile() เพื่อ return ข้อมูลจากการอ่านไฟล์มาส่งเป็นพารามิเตอร์ เพื่อดึงข้อมูลในไฟล์มาอ่าน แล้วทำการจัดรูปแบบให้อยู่ในรูปแบบลิสต์ แล้วทำการ sort ข้อมูลโดยใช้ฟังก์ชัน sortAscension() เพื่อกำหนดว่าข้อมูลจะเรียงจากหน้าไปหลังหรือหลังไปหน้า และฟังก์ชัน sortType เพื่อกำหนดว่าจะเรียงข้อมูลด้วยข้อมูลหมวดใด

```
def loadData(self):
    people = self.processData(self.readFile())
    people.sort(reverse=self.sortAscension(), key=self.sortType)    #manually create sort function because given method can't sort age as int
    self.listTable.setRowCount(len(people))
    for row in range(len(people)):
        for column in range(len(people[row])):
            self.listTable.setItem(row, column, QTableWidgetItem(people[row][column]))
            if (column in [0, 3, 4]):
                self.listTable.item(row, column).setTextAlignment(QtCore.Qt.AlignVCenter | QtCore.Qt.AlignHCenter)
```

ฟังก์ชัน readFile()

```
def readFile(self):
    file = open(self.findFile(self.fileName, self.filePath), "r")
    data = file.read()
    people = data.splitlines()
    file.close()
    return people
```

ฟังก์ชัน processData()

```
def processData(self, data):
    people = []
    for i in data:
        person = i.split()
        people.append(person)
    return people
```

ฟังก์ชัน sortAscension() และ sortType() ทำงานโดยการเช็คค่า action ใน menu bar ใดที่ถูกเลือกอยู่

```
def sortAscension(self):
    if (self.actionAscending.isChecked() == True):
        return False
    elif (self.actionDescending.isChecked() == True):
        return True
```

```
def sortType(self, person):
    if (self.actionBy_ID.isChecked() == True):
        return person[0]
    elif (self.actionBy_First_Name.isChecked() == True):
        return person[1]
    elif (self.actionBy_Last_name.isChecked() == True):
        return person[2]
    elif (self.actionBy_Age.isChecked() == True):
        return int(person[3])
    elif (self.actionBy_Sex.isChecked() == True):
        return person[4]
    elif (self.actionBy_Believe.isChecked() == True):
        return person[5]
```

โดยจะมีการสร้างกรุปของ menu bar เป็น 2 กรุปสำหรับการจัดเรียงข้อมูล กรุปแรกสำหรับการเลือกจัดตามหมวดหมู่ และกรุปที่ 2 สำหรับเลือกรูปแบบการเรียงลำดับ การสร้างกรุปจะทำให้ action ในกรุปสามารถทำงานได้แค่ครั้งละ 1 action เพื่อป้องกันไม่ให้เกิดการ sort ด้วยหลายหมวดหมู่ และใส่คำสั่งให้เมื่อกดเลือกใช้กรุปใดในการจัดลำดับก็ตาม จะทำการเรียกใช้ loadData() ใหม่อีกครั้งเพื่ออัปเดตตาราง

```
groupType = QtWidgets.QActionGroup(listWindow) #grouped the sort type for sort
groupType.addAction(self.actionBy_ID)
groupType.addAction(self.actionBy_First_Name)
groupType.addAction(self.actionBy_Last_name)
groupType.addAction(self.actionBy_Age)
groupType.addAction(self.actionBy_Sex)
groupType.addAction(self.actionBy_Believe)
groupType.triggered.connect(lambda: self.loadData())

groupAscension = QtWidgets.QActionGroup(listWindow) #grouped the Ascension option for sort
groupAscension.addAction(self.actionAscending)
groupAscension.addAction(self.actionDescending)
groupAscension.triggered.connect(lambda: self.loadData())
```

คำสั่งพิเศษนอกฟังก์ชัน มีไว้เพื่อป้องกันการการคลิกหัวตาราง หรือการ resize ตารางด้วยการคลีกลาก และป้องกันไม่ให้เกิดการพิมพ์ทับข้อมูลในตารางได้

```
self.listTable.horizontalHeader().setSectionsClickable(False)
self.listTable.horizontalHeader().setSectionResizeMode(QtWidgets.QHeaderView.Fixed)
self.listTable.setEditTriggers(QtWidgets.QTableWidget.NoEditTriggers)
self.listTable.verticalHeader().setVisible(False)
```
