國立虎尾科技大學

計算機程式 ag2 期末報告

PyQt5 事件導向計算器

PyQt5Event-DrivenCalculatorProject

學生：

設計一甲 40623104 曾昱瑄

設計一甲 40623105 陳微云

設計一甲 40623113 吳承哲

設計一甲 40623114 吳信億

設計一甲 40623115 吳隆廷

指導教授：嚴家銘

2018.01.10

摘要

機械設計一甲-計算機程式第二組的期末簡報，透過 Github 網站協同，分工合作完成計算機介面與程式。

- 各學員根據抽籤結果決定處理之按鈕
- 編譯各種按鍵之處理方式
- 期末計算機程式之心得感想

本簡報重點在於練習計算機的邏輯與 PyQt5 程式的運用以及分工合作之重要性。

ag8 計算機程式分配：

40623105 加減運算

40623104 記憶按鍵與其他按鍵

40623113 上一步、清除與等號按鍵

40623114 小數點與變號按鍵

40623115 乘除運算

目錄

壹、 數字按鍵處理 By 全組學員

1-1 數字按鍵程式

- 點按數字按鍵, 將會送出該按鍵的訊號
- 儲存按鍵發出的訊號後，判斷視窗上是否有計算中的數字
- 如果無計算中的數字且訊號數字為字串 0 則不動作
- 如果無計算中的數字且訊號數字不為字串 0 則顯示並堆疊顯示所按下之數字

程式如下：

```
    number = [self.one, self.two,
    self.three, self.four, self.five, self.six, self.seven, self.eight, self.nine, self.zero]
    for i in number:
    i.clicked.connect(self.digitClicked)

def digitClicked(self):
        clickedButton = self.sender()
        digitValue = int(clickedButton.text())
        if self.display.text() == '0' and digitValue == 0.0:
            return
        if self.waitingForOperand:
            self.display.clear()
            self.waitingForOperand = False
        self.display.setText(self.display.text() + str(digitValue))
```

貳、 加減運算按鍵處理 By40623105

●使用者按下加或減運算子按鍵時，程式設定以 additiveOperatorClicked() 處理進入 additiveOperatorClicked() 後，必須先查是否有尚未運算的乘或除運算子，因為必須先乘除後才能加減

●先處理乘與除運算後，再處理加或減運算後，將 sumSoFar 顯示在 display 後，必須重置 sumSoFar 為 0，表示運算告一段落

```
        self.plus.clicked.connect(self.additiveOperatorClicked)
        self.minus.clicked.connect(self.additiveOperatorClicked)

        '''加或減按下後進行的處理方法'''
        clickedButton = self.sender()
        clickedOperator = clickedButton.text()
        operand = float(self.display.text())
        if self.pendingMultiplicativeOperator:
            if not self.calculate(operand, self.pendingMultiplicativeOperator):
                self.abortOperation()
                return

            self.display.setText(str(self.factorSoFar))
            operand = self.factorSoFar
            self.factorSoFar = 0.0
            self.pendingMultiplicativeOperator = ''
        if self.pendingAdditiveOperator:
            if not self.calculate(operand, self.pendingAdditiveOperator):
                self.abortOperation()
                return
            self.display.setText(str(self.sumSoFar))
        else:
            self.sumSoFar = operand
        self.pendingAdditiveOperator = clickedOperator
        self.wait = True
    def multiplicativeOperatorClicked(self):


        # 若有等待加或減的運算子，執行運算
        if self.pendingAdditiveOperator:
```

```
            if not self.calculate(operand, self.pendingAdditiveOperator):
                self.abortOperation()
                return

            self.pendingAdditiveOperator = ''
        else:
            self.sumSoFar = operand
        self.display.setText(str(self.temp + self.sumSoFar))
        self.sumSoFar = 0.0
        self.waitingForOperand = True
    def calculate(self, rightOperand, pendingOperator):
        if pendingOperator == "+":
            self.sumSoFar += rightOperand

        elif pendingOperator == "-":
            self.sumSoFar -= rightOperand

        elif pendingOperator == "*":
            self.factorSoFar *= rightOperand

        elif pendingOperator == "/":
            if rightOperand == 0.0:
                return False

            self.factorSoFar /= rightOperand

        return True
    def clearAll(self):
```

參、 乘除運算按鍵處理 By40623115

3-1 乘除運算程式

- 按下乘或除運算子按鍵時，程式設定以 multiplicativeOperatorClicked() 處理
- 進入 multiplicativeOperatorClicked() 後，無需檢查是否有尚未運算的加或減運算子，因為乘除運算有優先權
- 先處理乘與除運算後，再處理加或減運算，將 sumSoFar 顯示在 display 後，必須重置 sumSoFar 為 0，表示運算告一段落

程式如下：

```
for i in multiply_divide:
    clicked.connect(self.multiplicativeOperatorClicked)

def multiplicativeOperatorClicked(self):
    clickedButton = self.sender()
    clickedOperator = clickedButton.text()
    operand = float(self.display.text())
    if self.pendingMultiplicativeOperator:
        if not self.calculate(operand, self.pendingMultiplicativeOperator):
            self.abortOperation()
        return
        self.display.setText(str(self.factorSoFar))
    else:
      self.factorSoFar = operand
    self.pendingMultiplicativeOperator = clickedOperator
    self.waitingForOperand = True
```

肆、 小數點與變號按鍵處理 By40623114

4-1 小數點按鍵程式

- 使用者按下小數點按鍵後，以 pointClicked() 方法處理，直接在 display 字串中加上 "." 字串數值變號按鍵處理。

程式：

```
self.pointButton.clicked.connect(self.pointClicked)

def pointClicked(self):
    if self.waitingForOperand:
        self.display.setText('0')

    if "." not in self.display.text():
        self.display.setText(self.display.text() + ".")
    self.waitingForOperand = False
```

4-2 變號按鍵程式

- 使用者按下變號按鍵後，由 changeSignClicked() 處理，若顯示幕上為正值，則在 display 字串最前面，疊上 "-" 字串。
- 假如顯示幕上為負值，則設法移除 display 上字串最前方的 "-" 字元。

程式：

```
self.changeSignButton.clicked.connect(self.changeSignClicked)

 def changeSignClicked(self):
     text = self.display.text()

             value = float(text)


              if value > 0.0:

        text = "-" + text
     elif value < 0.0:
        text = text[1:]
     self.display.setText(text)
```

伍、上一步、清除與等號按鍵處理 By40623113

- AC 或 C 可以清除掉所有的記憶，
- AC
  All Clear 全部清除鍵
  按下 AC 按鍵，可將整個運算式清除。
  範例： 235+882-762 ，按下 AC 按鍵，則整個運算資料接清除
  C
  Clear 清除鍵
  按下 C 按鍵，可清除運算式的最後數據
  範例： 235+882-762 ，按下 C 按鍵，則清除 762 這組數據
- = 可以鍵算出最後結果

  Equal 等於

  按下等於按鍵，可使前面運算得出結果

  範例: 235+882-762，按下=按鍵，則會計算出 355 這組數據結果

```
def clear(self):
        '''清除鍵按下後的處理方法'''
        #留著前面的數字
        if self.waitingForOperand:
            #下面不會執行
            return
        #清除
        self.display.setText('0')
        self.waitingForOperand = True
```

```
def clearAll(self):
        '''全部清除鍵按下後的處理方法'''
        #重設預設值
        self.sumSoFar = 0.0
        self.factorSoFar = 0.0
        self.pendingAdditiveOperator = ''
        self.pendingMultiplicativeOperator = ''
        self.display.setText('0')
        self.waitingForOperand = True
    def equalClicked(self):
        '''等號按下後的處理方法'''
        operand = float(self.display.text())
        '''
        同乘除
        '''
        if self.pendingMultiplicativeOperator:
            if not self.calculate(operand,
self.pendingMultiplicativeOperator):
                self.abortOperation()
                return
            operand = self.factorSoFar
            self.factorSoFar = 0.0
```

```
            self.pendingMultiplicativeOperator = ''
        '''
        同加減
        '''
        if self.pendingAdditiveOperator:
            if not self.calculate(operand, self.pendingAdditiveOperator):
                self.abortOperation()
                return
            self.pendingAdditiveOperator = ''
        else:
            self.sumSoFar = operand
        self.display.setText(str(self.sumSoFar))
        self.sumSoFar = 0.0
        self.waitingForOperand = True
```

以上就是寫出這些功能的程式

陸、記憶體按鍵與其他按鍵處理 By40623104

要記得刪除→ (小步驟：先描述按鈕之功能後，將程式貼上，記得填寫心得)

柒、 期末簡報心得與結論

40623104：

40623105：我在上傳時，常常遇到很多的錯誤，還好組長總是很熱心地幫助我，讓我可以順利的提交。組員們也都分工合作，使我們可以完成計算機。雖然我自己製作的時候，還是沒有很流暢，但是至少還是完成了，謝謝一路上幫助我的同學跟組長。

40623113：當我第一次上課的時候我就隱約知道阿=…這堂課是場硬仗，每次上課都是帶著緊張得心情去的，深怕當天內容不會，不過好在老師都會錄製當天的影片，使我能夠回家好好複習， 同學們也會互相幫忙，有幾天還熬夜趕工，真是煎熬的一堂課，後來分組後有一個很好的組長才能度過難關，當然 TA 與組員也是不可忽視的一環，從頭到尾也學到了很多程式語言的東西，原來電腦的世界是這麼廣闊。

40623114：由計算機程式當程式語言的第一步，只能在學習中慢慢了解每一行程式的意義；獨立完成簡直是不可能事，還好有強大的隊長支持著我們兩步兩步地往前走，成功結束這門課程。

40623115：在做完這些計算機按鍵後，我發現程式語言的的世界原來如此的廣大，從 Python 語言到 eric6，了解到網路協同的重要性。