**(選定負責的區域後麻煩注記載姓名之後,) (◎以選擇)標題頁** **-** 含報告標題、作者、日期與所屬單位

**目錄 -** 列出全文章節與所屬頁面連結

**1.摘要 ◎ -** 解釋報告編寫動機、報告重點、結論、建議與關鍵字

**2.前言** **-** 簡要描述全文意旨與背景, 說明報告目標與報告內容組成架構

**內文 -** 依照章節標題, 詳細說明報告內容, 包括方法、操作步驟與所得結果的詳細說明

**3.可攜程式系統介紹,Python 程式語法 ◎**

**4.fossil、github 倉儲**

**5. PyQt5 簡介 ◎**

**6. Calculator 程式 ◎**

**(1).結論 -** 條列所獲致的重要成果, 與摘要及導言中的目標互相呼應

**(2).討論與建議** **-** 說明後續值得進行的方向、方法與可能遭遇問題的討論

**參考** **-** 詳列所引用的參考內容

**附錄** **-** 附加值得參考的詳細內容

**(4).轉PDF** [PDF建立頁面連結說明](https://helpx.adobe.com/tw/acrobat/using/links-attachments-pdfs.html)

國立虎尾科技大學

機械設計工程系

計算機程式 bg5 期末報告

PyQt5 事件導向計算器

PyQt5 Event-Driven Calculator Project

學生：

設計一乙 40623222蔡博淮(內文-可攜程式系統介紹、Python 程式語法)

設計一乙 40623223 任明彥(摘要)

設計一乙 40623224 鐘偉哲(內文-Calculator 程式)

設計一乙 40623231周駿麟(內文-PyQt5 簡介)

設計一乙 40623232 余建杰(前言-討論與建議)

設計一乙 40623233 謝宗宏(結論、討論與建議)

指導教授：嚴家銘

2018.01.03

摘要

本次報告的目的在於統整這學期的課程內容並加以檢視及複習，參照上課的日期依序介紹各教學內容:可攜程式系統介紹、Python程式語法、fossil網誌、github倉儲、PyQt5 簡介、Calculator 程式。為多加練習多人協同合作之形式，本報告由各組學員分段同時編寫，以利未來行事之效率。根據結果，多人協同有助於減少個人作業的壓力並能與協同者有較多的交流，有利於討論及改進錯誤，使時間能在有效的運用下縮短整體作業時間。最後，歸納報告之結論，並整合各學員討論及建議，做為後續相關課程的參考資料。

**關鍵詞：**檢視及複習、多人協同、行事之效率、討論及改進錯誤。

目錄

摘要 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . i

目錄 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . ii

表目錄 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . iii

圖目錄 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . iv

第一章 前言. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1

第二章 可攜程式系統介紹 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

2.1 啟動與關閉 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

2.2 啟動與關閉 2 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

第三章 Python 程式語法 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

3.1 變數命名 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

3.2 print 函式 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

3.3 重複迴圈 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

3.4 判斷式 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

3.5 數列 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

第四章 倉儲系統. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

4.1 Fossil SCM . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

4.2 Github 協同 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

第五章 PyQt5 簡介 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

5.1 PyQt5 架構 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

第六章Calculator 程式. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .6

6.1計算機製作. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .6

第七章 結論. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .7

7.1 結論與建議 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .7

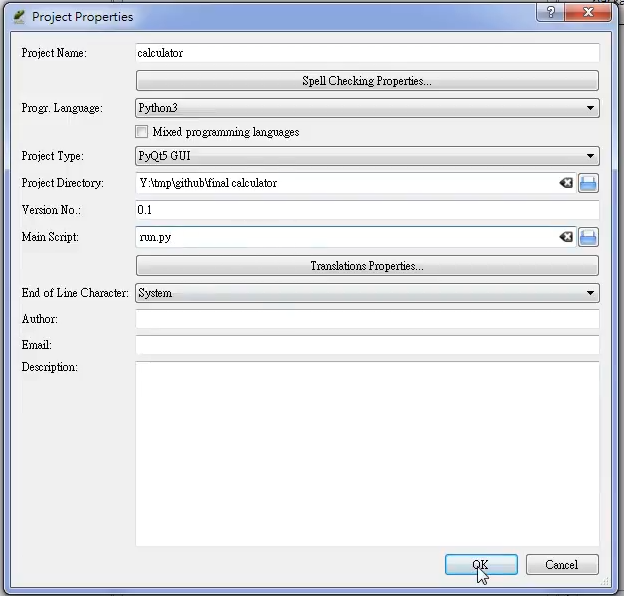
參考文獻. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

表目錄

圖目錄

前言

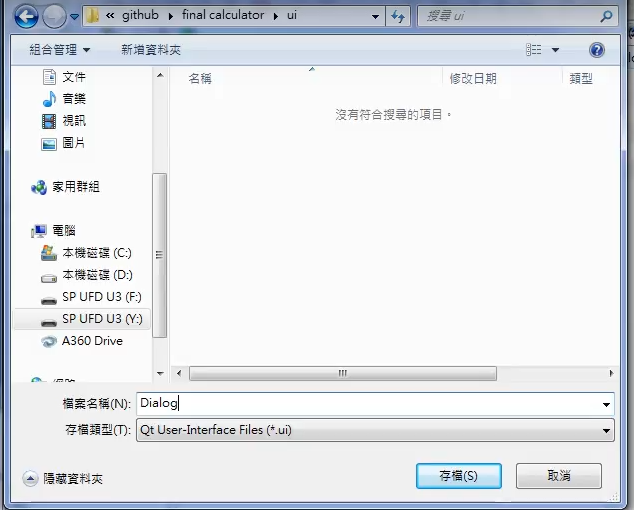
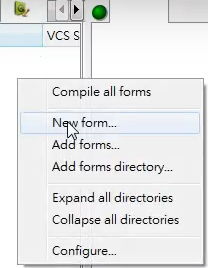
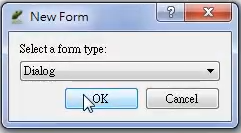
**Calculator 程式 計算機製作**

1. 先開啟eric6
2. 新增一個專案，完成後確認

給予一個專案名稱

選取到對應的資料夾

主要跑的程式

1. 建立一個表單

建立ui資料夾

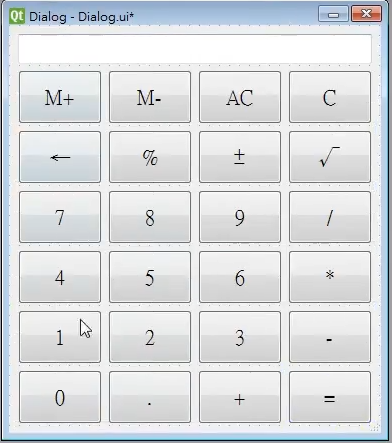
並存於底下

表單名稱為Dialog

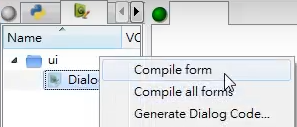
1. 用C:\Users\ADER\Desktop\未命名.png以及C:\Users\ADER\Desktop\未命名.png完成計算機表單並

C:\Users\ADER\Desktop\未命名.pngC:\Users\ADER\Desktop\未命名.png修改主詞和

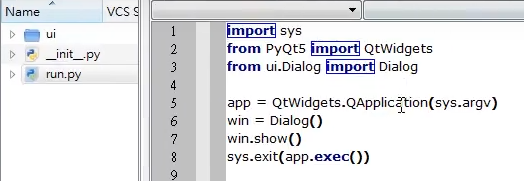
圖片大小

C:\Users\ADER\Desktop\未命名.png

C:\Users\ADER\Desktop\未命名.png

完成後把圖片的東西編寫上去

1. 在run.py的部分把程式撰寫上去



數字按鍵以zero, one, two, three, four, five, six, seven, eight, nine命名、顯示幕以display命名、等號以 equalButton命名。

與 MS, M+, 或 MC 按鍵相關的計算機記憶體數值, 存入 sumInMemory 變數對應的記憶空間，以 sumSoFar 儲存累計數值，使用者按下等號後， sumSoFar 重新計算結果，並顯示在 display 幕，Clear All 按鍵則重置 sumSoFar 為 0。

以 factorSoFar 儲存乘或除運算子運算過程所得的暫存數值； pendingAdditiveOperator 儲存使用者最後點按的加或減運算子字串；pendingMultiplicativeOperator 儲存使用者最後點按的乘或除運算子字串；waitingForOperand 界定使用者是否處理運算數輸入階段，若 waitingForOperand 為 True，表示計算機正在等待使用者"開始"輸入運算數，waitingForOperand 起始值為 True，只有重新進入 digitClicked 方法，display 才會 clear()，否則在顯示幕非為 0 的情況下，將堆疊數字字串。

因為考量先乘除後加減運算順序，將加減與乘除處理方法分開，若使用者輸入加減運算子後，緊接乘除運算子，計算機會先乘除運算後再加減；直接運算元可以在使用者按下按鍵後, 直接對 display 中的數值進行處理，無需其他暫存需求；pendingAdditiveOperator、 pendingMultiplicativeOperator、 sumSoFar、factorSoFar 與 waitingForOperand，在 Dialog 類別建構子中設定起始值。

**數字按鍵點按處理:**

使用者點按數字按鍵，將會送出該按鍵的 clicked() 訊號

按鍵的 clicked() 訊號將會根據設定， 觸發 digitClicked() 方法槽，由於 PyQt5 的 Push Button 以 Qt5 中的 QObject::sender() 送出訊號，此函式會傳回 sender 作為 QObject 的指標，因為此一與 Push Button 配合的 sender 為 Button 物件，因此可以在 digitClicked() 函式中，利用 sender().text() 取得按鍵的 text 字串，假如使用者點按 0，display 顯示字串 0，但是若一開始輸入兩個以上的 0，digitClicked() 應該仍只顯示 0 字串，但是若計算機處於等待新運算數輸入時 (以 waitingForOperand 判定) ，新數字在顯示前，display 應該要清除先前所顯示的數字，最後，除了在顯示幕為 0 之後的 0 按鍵輸入，digitClicked() 方法槽不會繼續判定是否清除顯示幕或堆疊數字字串外，所按的數字將會堆疊顯示。

**直接運算按鍵處理:**

Sqrt, x^2 與 1/x 等按鍵的處理方法為unaryOperatorClicked()，與數字按鍵的點按回應相同，透過 sender().text() 取得按鍵上的 text 字串；unaryOperatorClicked() 方法隨後根據 text 判定運算子後，利用 display 上的運算數進行運算後, 再將結果顯示在 display 顯示幕，若進行運算 Sqrt 求數值的平方根時，顯示幕中為負值，或 1/x 運算時， x 為 0， 都視為無法處理的情況，以呼叫 abortOperation() 處理；abortOperation() 方法則重置所有起始變數，並在 display 中顯示 "####"，直接運算子處理結束前，運算結果會顯示在 display 中，而且運算至此告一段落，計算機狀態應該要回復到等待新運算數的階段，因此 waitingForOperand 要重置為 True。

**加或減按鍵處理:**

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。

**乘或除按鍵處理:**

使用者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。

**小數點按鍵處理:**

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:**

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。

**退格按鍵處理:**

使用者按下退格按鍵後，由 backspaceClicked() 處理，這時可以利用 Python 字串數列中的 [:-1] ，保留除了最後一個字元的字串，離開 backspaceClicked() 前，將顯示幕中原有字串的 [:-1] 字串，顯示在 display 上，若退格後 display 上為空字串，則顯示 0，並且將 waitingForOperand 起始設為 True，表示等待新運算數中。

**清除按鍵處理:**

使用者按下 Clear 按鍵後，以 clear() 方法處理，進入函式後，將現有的運算數重置為 0，離開 clear() 前，將 waitingForOperand 起始設為 True，表示等待新運算數中，ClearAll 按鍵，則將所有變數全部重置為起始狀態。

**記憶體按鍵處理:**

clearMemory() 方法與 "MC" 按鍵對應，清除記憶體中所存 sumInMemory 設為 0；readMemory() 方法與 "MR" 按鍵對應，功能為讀取記憶體中的數值，因此將 sumInMemory 顯示在 display，作為運算數；setMemory() 方法則與 "MS" 按鍵對應，功能為設定記憶體中的數值，因此取 display 中的數字，存入 sumInMemory；addToMemory() 方法與 "M+" 按鍵對應，功能為加上記憶體中的數值，因此將 sumInMemory 加上 display 中的數值，因為 setMemory() 與 addToMemory() 方法，都需要取用 display 上的數值，因此必須先呼叫 equalClicked()，以更新 sumSoFar 與 display 上的數值。

**calculate() 方法:**

calculate() 方法中的運算，以 rightOperand 為右運算數當執行加或減運算時，左運算數為 sumSoFar，當執行乘或除運算時，左運算數為 factorSoFar，若運算過程出現除與 0 時，將會回傳 False。

**完成後，計算機就能加減乘除了~**

結論

有了這次的協同設計練習，讓我們體會到協同設計的重要性，在職場上不可能所有的東西都是你一個人自己做，一定是協同設計，每個人都負責不同的地方一點一點的拼湊出來的，至於為甚麼要用協同設計，主要的原因是因為人不可能都不休息和有問題都自己解決，有了這個協同設計當你在休息的時候換別人依序你所做的，有問題也可以提出來一起討論，以利提升行事的效率。

討論與建議

這次的計算機因為有了老師的範本才可以做得比較順利，在未來不太可能會有範本讓你參考，只能靠自己和自己協同設計的組員一步步的完成，所以一定要慢慢的提升自己的能力，畢竟一項設計不可能只依照少數組員來完成，一定是每個組員都負責不同的事加以提升作事的效率和減輕個人作業的壓力。