Lab 4: Python GUI Programming Report

學號： 109511207 姓名： 蔡宗儒

1. 請簡述 Python Tkinter GUI 程式主體架構 (如何引入模組.宣告主視窗及物件...等等)

首先以 import tkinter as tk 來引入函式庫，建立 GUI 程式主體架構主要有三個步驟，如下。

(1) 建立主視窗(設定視窗大小、位置跟視窗名稱)，ex. window = tk.Tk()、window.title('Lab4')、window.geometry('200x200')、window.resizable(False, False)。
tk.Tk(): GUI 的核心，用此函示建立架構
title(): 程式上方的文字
geometry(): 設定 width 跟 height
resizable(): 定義視窗是否能被放大和縮小，兩個 False 代表 width 跟 height 都不能被放大縮小。

(2) 設定視窗 widget(button、label、frame 等等)，設定 widget 的大小、位置以及需要連接的 event 等等，並將 widget 放入視窗內，button 可讓使用者在按下後執行某些行為；label 可建立文字或圖形標籤；frame 可以用來做為容器分割主視窗。ex. tk.Frame(window)、btn0 = tk.Button(...)、tk.Label(...)

(3) 定義每個事件的函示，像是按下 button 應該要有什麼呈現，就要寫出相對應的函式。

綜合以上這三個步驟便可做出簡易應用程式。

2. 請解釋 Python 計算機的程式碼(將程式貼上並加上註解)。
(請詳細標注能實現的功能，如：/0 能顯示錯誤訊息，0/任何數字為 0，第一次計算完的結果能繼續做運算，負數計算...等等)

```
# import package
import tkinter as tk

# x1 stores the 1st num, x2 stores the 2nd num
x1 = ''
x2 = ''

# operator stores the operator
operator = ''

# sign1 stores the sign of the 1st num, sign2 stores the sign of the 2nd num
sign1 = ''
sign2 = ''

# Display the calculation results in the upper window
def SetValue():
    tk.Label(f1, textvariable = var, height = 3).grid(column = 0, row = 0)

# Press button 0~9 to make corresponding judgments and store the values in respective
variables
def Click(num):
```

```
    global x1, x2, operator, sign1, sign2

    # When the operator has not been assigned an operator yet,
    # it means that the incoming number is intended for the 1st num
    if not operator:

        # The 1st num is positive, simply concatenate it
        if sign1 != '-':
            x1 = x1 + num
            var.set(x1)

        # The 1st num is negative, multiply the result by -1
        else:
            x1 = str(int(num) * -1)
            var.set(x1)

            # When the 1st num is negative, clear the variable,
            # and all the subsequent digits are treated as positive numbers for computation
            sign1 = ''

    # When the operator has been assigned an operator,
    # it means that the incoming number is intended for the 2nd num
    else:

        # The 2nd num is positive, simply concatenate it
        if sign2 != '-':
            x2 = x2 + num
            var.set(x1 + operator + x2)

        # The 2nd num is negative, multiply the result by -1
        else:
            x2 = str(int(num) * -1)
            var.set(x1 + operator + x2)

            # When the 2nd num is negative, clear the variable,
            # and all the subsequent digits are treated as positive numbers for computation
            sign2 = ''

# Press the button C will clear all variables,
# and when the program executes SetValue(), the label will be set to
zero
def Clear():
```

```
    global x1, x2, operator, sign1, sign2
    x1 = ''
    x2 = ''
    operator = ''
    sign1 = ''
    sign2 = ''
    var.set('0')

# Press the button + - x / will perform arithmetic operations on two numerical values
def Calculate(op):
    global x1, x2, operator, sign1, sign2

    if op in ['+', '-', 'x', '/']:

        # The 1st num has been assigned,
        # it means that this symbol is used for arithmetic operations
        # or determine whether the 2nd num is positive or negative
        if x1:

            # The operator has been assigned,
            # it means that this symbol is used to determine whether the 2nd num is positive or
negative
            if operator:
                sign2 = op
                var.set(x1 + operator + sign2)

            # The operator has not been assigned yet,
            # it means that this symbol is used for arithmetic operations
            else:
                operator = op
                var.set(x1 + operator)

        # The 1st num has not been assigned yet,
        # it means that this symbol is used to determine whether the 1st num is positive or
negative
        else:
            sign1 = op
            var.set(sign1)

    # The 2nd num has been assigned,
    # perform arithmetic operations and calculate the final result
    if x2:
```

```
        # Addition
        if operator == '+':
            ans = int(x1) + int(x2)

        # Substraction
        if operator == '-':
            ans = int(x1) - int(x2)

        # Multiplication
        if operator == 'x':
            ans = int(x1) * int(x2)

        # Division
        if operator == '/':
            if x2 != '0':
                ans = int(x1) // int(x2)

            # / 0 means ERROR
            else:
                ans = 'ERROR'
                x1 = ''
                x2 = ''
                operator = ''
                sign1 = ''
                sign2 = ''

        # Present the calculation result
        if op == '=':

            # If ERROR occurs
            if ans == 'ERROR':
                var.set(ans)
                x1 = ''
                x2 = ''
                operator = ''
                sign1 = ''
                sign2 = ''

            # store the result in the 1st num, allowing for further calculations based on this
result
            else:
```

```
                var.set(str(ans))
                x1 = str(ans)
                x2 = ''
                operator = ''
                sign1 = ''
                sign2 = ''

if __name__ == '__main__':

    # Create main window
    window = tk.Tk()
    window.title('Lab4')

    # Split the window into two sections
    f1 = tk.Frame(window)
    f2 = tk.Frame(window)
    f1.pack()
    f2.pack()

    # Set the string variable 'var'
    var = tk.StringVar()

    # Display the initial value as 0
    var.set('0')

    # Display the string variable "var" obtained in this iteration in the upper window
    SetValue()

    # Set button 0~9, plus(+), minus(-), divide(/), multiply(*), equal(=), clear(c)
    btn0 = tk.Button(f2, text = '0', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('0')).grid(row = 3, column = 0)
    btnc = tk.Button(f2, text = 'C', borderwidth = 5, width = 6, height = 2, command = lambda:
Clear()).grid(row = 3, column = 1)
    btne = tk.Button(f2, text = '=', borderwidth = 5, width = 6, height = 2, command = lambda:
Calculate('=')).grid(row = 3, column = 2)
    btnd = tk.Button(f2, text = '/', borderwidth = 5, width = 6, height = 2, command = lambda:
Calculate('/')).grid(row = 3, column = 3)
    btn1 = tk.Button(f2, text = '1', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('1')).grid(row = 2, column = 0)
    btn2 = tk.Button(f2, text = '2', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('2')).grid(row = 2, column = 1)
```

```
    btn3 = tk.Button(f2, text = '3', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('3')).grid(row = 2, column = 2)
    btnp = tk.Button(f2, text = '+', borderwidth = 5, width = 6, height = 2, command = lambda:
Calculate('+')).grid(row = 2, column = 3)
    btn4 = tk.Button(f2, text = '4', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('4')).grid(row = 1, column = 0)
    btn5 = tk.Button(f2, text = '5', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('5')).grid(row = 1, column = 1)
    btn6 = tk.Button(f2, text = '6', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('6')).grid(row = 1, column = 2)
    btnm = tk.Button(f2, text = '-', borderwidth = 5, width = 6, height = 2, command = lambda:
Calculate('-')).grid(row = 1, column = 3)
    btn7 = tk.Button(f2, text = '7', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('7')).grid(row = 0, column = 0)
    btn8 = tk.Button(f2, text = '8', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('8')).grid(row = 0, column = 1)
    btn9 = tk.Button(f2, text = '9', borderwidth = 5, width = 6, height = 2, command = lambda:
Click('9')).grid(row = 0, column = 2)
    btnx = tk.Button(f2, text = 'x', borderwidth = 5, width = 6, height = 2, command = lambda:
Calculate('x')).grid(row = 0, column = 3)

    # Repeat the loop to keep the computation ongoing
    window.mainloop()
```

3. 心得

GUI 是 Graphical User Interface，也就是用圖形方式顯示的使用者介面，像是這次做的小算盤，上次做的圈圈叉叉等。早期的 Command-Line Interface 通常只支援滑鼠，且要記住繁瑣的指令，對於使用者非常不友善，也才有了 GUI 的出現。

而 Tkinter 是可以將 Python 程式碼變成圖形化介面的套件庫，裡面給了很多一基本物件，像是 button、frame、label、scrollbar 等等，且 Tkinter 為內建，不需要 pip 安裝。

而本次實驗要實作一個小算盤，其實要考慮的點滿多的，像是要如何去存變數、要如何判斷運算子的功用是計算還是單純賦予正負號，要稍微想一下邏輯才能將程式碼完整地時做出來

4. Reference

[1] Day 23 : Tkinter-利用 Python 建立 GUI(基本操作及佈局篇)

[2] 不間斷 Python 挑戰 Day 30 - 使用 tkinter 開發 GUI 程式：常用視窗元件

[3] Python GUI 编程(Tkinter)