我對(與寫程式相關的)語言的看法、想像與期望

資訊三乙 10727219 許志仲

1. 曾經接觸過的語言

Visual Basic

在我高中的資訊課裡老師的教材中有教我們學習使用Visual Basic語言，這應該算是我第一個學會的程式語言，VB是由微軟基於Basic所開發的程式語言，是第一個支援視覺化介面設計的程式語言，它的特色有語法簡單易懂、交談式解譯與執行、自帶GUI等，在當時我們需要安裝Microsoft Visual Basic來編譯VB程式，最特別的就是我們需要先手動用拖曳的方式來創建自訂的視窗，像是新增Buttom、Label等物件，系統會在後台自動幫你生成對影的程式碼，之後再進行coding，縱使後來我還是發現Microsoft Visual Basic提供包含主控台視窗(或者說我們現在常用的黑底白次命令頁)、Windows Form…等模式，但我當時還是以為所有程式語言都是用這樣先自訂Windows Form的模式在運作的。

在我印象當中，VB是個相當簡單直接的程式語言，它的語法大部分都相當簡單直覺，讓我們這種高中生初學者很容易了解程式碼的意思，Microsoft Visual Basic在編譯過程中也會指出語法錯誤的程式碼，並提供相對應的錯誤說明、提供修改方法，撰寫過程中還提供自動選字功能，對我們來說算是相當方便的。

說道VB的缺點，正因為它的簡單，導致很容易造成漏洞，功能也不向其他語言來的強大，導致越來越多VB使用者轉向改用C#，微軟.NET開發團隊也在去年宣布不再將VB當作獨立語言發展，但我依然認為VB是很適合初學者學習程式語言的工具。

C/C++

C/C++應該是世界上最廣泛被使用的語言(之一)，具有高效、靈活、功能豐富…等特性，也是我大學踏入中原資工之後學到的第一個程式語言，C語言最初由貝爾實驗室基於B語言為基礎開發，更後來的C++為在C語言的基礎上，新開發得”包含class的C語言”，有點像是C語言的增強版，後續又持續增加operator overloading、exception…等功能。

說到一開始上計概時，因為CAL系統無法複製貼上的緣故，我們偶爾必須另外自己測試自己的程式撰寫邏輯，當時系上的同學都會使用Dev C++來寫程式，不過Dev c++有缺乏維護版本老舊的問題(聽說已經有新的開發團隊接手維護更新了)，現在大家比較常使用VS Code來寫程式，說到這個就要提到C++ 11 這個C++語言的更新版本，在C++11中提供了相當多的功能，像是用來確保記憶體管理的智慧型指標、lambda函式(不確定是否與scheme中的lambda function功能目的一樣)、如stoi()之類的型別轉換function(如果PAL系統能夠支援的話Project應該能夠寫得更好的)。

縱使擁有廣泛的擴展性和豐富的功能，說到C/C++語言我覺得缺點還是在於可以直接到記憶體位置進行修改的部分，這樣導致使用者很容易駭進系統，導致老大會在PAL系統裡加入多項檢查限制？，進而導致我在寫PL project時，送出程式碼後需要等待很久的時間，還要登入bbs申請增加執行時間，一定程度降低了我檢查程式的效率。

Python

不知道從甚麼時候開始，機器學習成為資工系專題題目的首選，系上幾乎每個教授都在做機器學習相關題目，而Python語言，憑藉著它嚴格一致的特性，以及numpy, scikit-learn…等對於科學計算相當方便的工具，成為了機器學習使用程式語言的首選。

第一次使用到Python寫程式應該是在大二工程數學的時候，我們必須要Python實現一些微積分函數圖形的繪畫，得益於numpy強大的計算能力，以及matplotlib的繪圖功能，我們能用python輕鬆地畫出很多複雜的圖形，雖然在這之前我們只學習過C以及java的語法，但是根據系上教授的說法，這是我們應該自己學會的。

在以往學過的語言，如C、Java中，變數再宣告過程中都是需要先給予固定的型別的，而Python是Dynamic Type Binding，可以在程式執行過程中任意變換type，這是我認為python最特別的地方，一開始不習慣時，常常需要一直注意這個變數到底是甚麼型別，在執行程式時總是一直跳出error訊息，說這個type沒有這個功能，我覺得這個功能有利有弊，優點是可以根據需要任意改變變數型別，不用再額外宣告同樣目的但是不同type的變數，像是在數字計算的程式當中，可能要同時宣告intnum、floatnum之類的變數(如同PL OurScheme程式一樣)，缺點是如果不是寫程式的人本身，要看懂程式就會變得相當複雜，因為不確定程式執行到現在這個變數到底是甚麼型別，就算是撰寫者本人，也有可能在程式行數過大時自己也搞混。

Rust

最後一個談到一個我沒有使用過的語言Rust，最初是從就讀那所雖然程式能力跟我們差不多但是比我們會打嘴砲的中央大學的高中同學聽來的。

相較於其他語言，Rust屬於較年輕的語言，從2010年作為Mozilla官方專案出現，到現在也只不過11年的時間，作為一個新興的程式語言，Rust具有不少高階語言的特性，但仍具有接近 C++ 的效能，其中我認為最大的優勢在於它的安全性，在C/C++裡，我們可以任意的讓指標指向任何記憶體，這種機制雖然自由，但也伴隨著安全性的風險，但在Rust的編譯器中，它會將危險的記憶體操作擋下並當作程式的Error，不過其餘無害的操作仍然可以執行，這樣子雖然導致程是在編譯時會導致出現比較多Error，但好處是程式實際在執行時比較不容易發生錯誤，就如同PAL系統一樣，我們常常在外部執行自己寫的程式時都沒有遇到Error，但是將程式丟進PAL系統時卻常常出現很多complie error(雖然說之後的風格檢查更麻煩就是了)。以目前來說，因為Rust屬於比較新興的語言，它的擴充性與功能相較於其他成熟語言還有不小的落差，而且Rust屬於語法相對複雜的語言，對於初學者可能相對較困難，但我認為它依然是個有趣的語言，它宛如跟PAL系統一樣的安全檢查讓我頗有共鳴。

1. 有興趣的領域

如果要說選擇就讀資工系的理由，除了打電腦遊戲之外，應該就是網路安全領域了吧！因為我覺得很帥，不論是越過系統防護破解資料、或是抓出系統漏洞防守敵人的攻擊，用想的就覺得是一門很厲害的學問。

在網路安全領域當中，程式能力應該也是相當重要的，我覺得在檢查系統程式漏洞、發現惡意code的處理方面，跟我們平常在寫程式時debug的作法是差不多的。

一開始我以為在網路安全方面，大家常使用的語言應該會以網頁、數據庫相關的程式語言為主流，像是JavaScript、SQL…等語言，但是根據google到的資料來看，Python有在網路安全領域也不乏有許多使用者，這個發現蠻出乎我意料的。

Python 在人工智慧盛行的最近幾年間儼然成為了最熱門的程式語言之一，很大一部分的原因就是因為它的功能廣泛且簡單易學。在網路安全的領域當中，常常在短時間有變化萬千的問題，很多既有的技術及功能可能在短時間內無法因應快速的變化而變得無用，因此大家才會轉而使用簡單且多功能、擴充性強大的語言，期望直接以自我的程式設計能力，因應不同狀況問題編寫對應的功能，而Python正好就是這些要求的首選語言。相較於其他的語言，我認為Python的優勢就在於它龐大的擴充函式庫可以幫助網路安全工程師在解決問題時找到適合的功能，