1. **題目說明:**

Input:一段RISC-V組語

Output:將branch的部分用2-bit history去做prediction

1. **使用函式庫:** <iostream> <string> <vector> <fstream>函數所在函式庫
2. **與作業1的改變：**

之前作業1轉成opcode時，因為可以一行一行分別轉換完就好，所以是用一維的string陣列來記錄每一行組語，處理完就把他覆蓋掉。

但本次作業因為要考慮到branch，需要把整段組語都記錄起來，所以是改成用二維的string陣列來將整段組語記錄起來，方便後續作branch的動作。

1. **函式介紹:**

1.vector<string> splitStr2Vec(string s, string splitSep)

根據指定的符號[:,()]將一行的字串做切割。

2. void find\_label(string inst[][20], int address);

因為我的組語code，標籤跟程式碼是寫在同一行，所以會在這個function將整段組語code跑一遍，如果前方有標籤，會把inst[i][9]改成”Label”，方便之後在解讀組語時有一個比較條件。

3. void execute(string inst[][20], int address)

除了傳過來的二維陣列inst，新增了rg[20]，用來記錄register的值，如果今天是R2，我就會把值存在rg[2]，以此類推。

pre\_inst[]則是會記錄NTNTNT的值，一樣，如果今天發生在第i個instruction，我就會把值存在第i項。

接著會去跑一個while迴圈，並且會用一個i值去記錄目前進行到哪一個inst，接著分別去判斷屬於li、addi、branch

Branch的話，就會判斷有沒有做branch，如果有，pre\_inst[i]就會加上 ’T’否則加上’N’，並且會把i值改到branch後方的label處。最後呼叫output()來做prediction結果的印出。

4. void output(string pre\_inst[], string inst[][20], int address)

在此處輸入需要的entry數量，新增place[]陣列，以及taken[][]二維陣列，前者會去紀錄pre\_inst[](紀錄NTNT陣列)所在的index，後者則是會去儲存SN、WN等等的2-bit history狀況。最後的一大串程式就是在做prediction的運算及output。

輸出結果在下一頁，打勾的兩處多的output是我到時候demo方便講解用的，所以先留著。

輸出結果

