問題解決方式與創意技術：

1. 創意技術：

在遇到端點的時候，大部分的組別都是採用迴轉的方式，但是這樣我們認為會讓所花費的時間增加以及在迴轉的時候不穩定的機會可能會大大增加，因此我們這組是採用倒退的方式，這樣就不需要再多花費迴轉的時間，可以直接讀取到RFID卡之後，就執行下一個動作（通常會倒退走到下一個node，在按照演算法提供的指令繼續下來的行動）。

1. 問題以及問題解決方法：
2. 硬體設備的問題（包含Arduino板、馬達控制模組L298N、升壓模組等等）：

首先，我們花最多時間在上面的問題第一個絕對是硬體設備上問題的排除，在project剛開始的時候，我們就因為接線路時不慎短路，使得升壓模組以及L298N燒壞了，所以經過了大翻新，換了新的零件，線路重接之後，終於，硬體問體在短暫的一個月左右都維持得還不錯。但是，在Final Project的前一個星期六最後一次open lab的時候，因為我們的code問題其實解決得差不多了，但是測試到一半的時候，突然發現車子不會動了，因為我們之前也有發生過類似的情形（詳見問題二），但是之後測試過後發現好像情況沒那麼簡單，經過了一連串的零件替換過程，像是把Arduino板、L298N、升壓模組全部在明達405找過新的零件替換一遍，還是不見成效。後來去問了別組的解決方法，把充電的插頭直接插到arduino板的孔，之後發現好像還是不太行，又換了幾片arduino 後，發現在405的arduino板都是有問題的，之後又重新接回最原本的那塊，突然發現可以動了，再把全部都接回原本的零件，發現依舊可以動，但是速度變得跟以前差很多，不過因為經過了幾個小時的停止，發現可以動就已經很慶幸了，所以我們就用這個速度為前提下，做了一些參數上的修正。

1. Serial Library的相關問題：

大概在project剛開始的時候，我們把接頭都插上去了之後，發現怎麼只有一邊的輪子會動，明明在arduino上是讓兩邊的速度都一樣，但不果怎麼擺arduino的code，都無法讓那個輪子動，後來上網查了一下才知道，當使用了Serial Library中的函數之後，0和1的接腳就無法作為Digital input或output來使用，所以我們的解決方法是決定直接把Serial Begin給拿掉，結果不出所料的，輪子很順利的兩邊都會動了。

1. 在arduino 中，不可在同一個函數同時做read 和write：

在測試藍芽階段的時候，我們原本寫的是想要讓arduino傳給藍芽，然後藍芽再回傳某個對應的值給arduino來測試他們兩個之間的溝通是可以雙向且順暢的，但是arduino中的監看埠那邊就會有收到一些很奇怪的東西，後來發現arduino不可再同一函數中同時有read和write且分成兩個不同的函數時，也要注意它不可在時距太小的間隔中同時做read 和write ，不然也會有問題。

1. 藍芽連接電腦的問題：

在起初的時候，我們要嘗試藍芽連接電腦的時候，發現電腦可以偵測到藍芽，會顯示「已配對」，但是每次要使用藍芽去進行傳輸的時候，每次都抓不到，按照之前藍芽教學的投影片去做了調整之後依舊還是無法改善問題，後來請教助教之後，發現換了一片藍芽之後就可以成功連線，他會顯示「已連線」。這問題也就此解決了，不過後來我們有想說可能是我們在測試的時候測試的方法可能有問題，也有可能不全然是硬體的問題。

1. 轉彎以及循跡解決的過程：

循跡：

在起初的時候，我們就已經把P control寫的差不多了，但是就像之前上課時那樣，車子無法穩定的走直線，會有蛇行的情況發生，所以我們試著引入PID control ，但是我們一開始PID control 的公式理解錯誤，導致車子比原本只有 P control 的時候還要走的更不穩，所以我們把PID先捨棄，先從PD開始，狀況有好一些，維持使用大約到了六月初，後來經過更多次的實驗之後覺得PD遇到error太大的時候會無法修正，直接出軌。後來決定使用PDα，防止速度變化過大。但是還是有一些不穩定，主要是因為還是無法修正error過大的問題，所以最終我們是採用PIDα，最終前進的循跡已經修正的還滿不錯的。但是倒退走的循跡有時候還是滿不穩定的，主要是因為我們的紅外線感測器是放在比較靠前端的位置，所以紅外線偵測到要修正的時候，車子可能已經跑出軌道太多了，會有點修正不回來，所以在7/4以前會再試著修正倒退循跡。

轉彎：

轉彎則是我們處理時間可能僅次於硬體設備的部分了，一開始在測試轉彎的時候，發現他只要一轉就會一直無止盡的原地自轉，後來發現是因為sensor的0和1設反，這個看似很小的問題花了很多時間才終於發現是這個只需動動手指花個30秒就解決的問題。之後我們一開始是寫成走到node之後，直接用固定的轉速轉固定的delay，但後來發現可能欸因為地圖每個node的地形不同，有些地方會有嚴重的凹凸不平，而且有時候如果循跡不小心走的不夠好，那轉固定的秒數就會讓他一樣歪甚至更歪，所以之後改成轉了一些秒數之後，讓他用紅外線偵測的方式去看他到底有沒有轉到好的位置，中間一度覺得這方法還是有點無法克服地圖凹凸不平，所以我們有試著透過在轉彎的時候，印出每5ms紅外線感測器的讀值，進而判斷是否可以利用可能某幾個燈的1和0的交替變化去修正轉彎，後來發現有時候這個方法會受到node黑線的限制而受影響，所以最終我們還是採用轉一些秒數之後再用紅外線判斷，再把那些秒數和偵測的方法做進一步的探討怎樣是最好的。

分工：

演算法：陳泓叡

循跡：（大部分：謝兆和)、全部人都有參與討論修正

藍芽：鐘民憲

RFID：余昇翰