程式設計大賽於 2004 年起第一次舉辦至今，已為 Openfind 年度重要的盛事及傳統之一。公司內部人員於比賽期間將打破原部門建置，重新編組，依據比賽題目內容，並在有限的時間及資源下，發揮最大的創意及團隊合作，努力達成目標。2010 年 Openfind 程式設計大賽花絮如下：

題目：Openfind Railroad Tycoon(鐵路大亨). 比賽時間：24 小時

競賽規則：Openfind Railroad Tycoon 比賽題目下載
(1) 歐洲地圖上有 10 個都市，各自出產特定資源。
(2) 有 3 台性能不同火車想辦法在時間內載運貨物賺錢。
(3) 單場最高分得 10 分，其他組別按比例計算分數。10 場總分最高者獲勝。

比賽結果名次公布：

- 第一名：**Henry, Shou_lai, Yaoan**
- 第二名：**Mindi, Shawn, Shenqyang**
- 第三名：**Felka, Josh, Tony, Willie**

戰況剪影：

◎ 比賽地圖(藍色數字代表公里數)

◎ 都市出產貨物 及 運送價目表

收購價格/車	Birmingham 伯明罕(0)	London 倫敦(1)	Paris 巴黎(2)	Amsterdam 阿姆斯特丹(3)
Coal(煤)	0	75	250	522
Mail(郵件)	60	0	140	350
Wine(葡萄酒)	200	140	0	180
Tulip(鬱金香)	375	310	160	0
Computer(電腦)	464	396	220	374
Steel(鋼鐵)	672	575	396	200
Beer(啤酒)	560	480	335	457
Wood(木材)	740	670	477	436
Amber(琥珀)	980	805	660	440
Oil(石油)	2430	2230	1830	1456

Frankfurt 法蘭克福(4)	Berlin 柏林(5)	Munich 慕尼黑(6)	Vienna 維也納(7)	Warsaw 華沙(8)	Moscow 莫斯科(9)
577	840	810	1020	1235	2030
395	570	530	744	875	1485
220	395	375	530	670	1250
335	180	455	435	395	935
0	140	120	262	374	960
140	0	262	241	200	720
108	235	0	126	283	785
236	218	126	0	144	603
370	200	315	160	0	440
1335	1008	1225	925	615	0

◎ 火車能力表

	車頭	車廂數	時速
A	TGV	2	360km/hr
B	AMD-103	5	180km/hr
C	Class E111	10	120km/hr

◎ 比賽結果

得分表	1	2	3	4	5	6	7
input01.txt	7.85	6.59	9.45	9.37	10.00	6.19	10.00
input02.txt	7.26	6.80	10.00	8.84	9.38	5.16	9.78
input03.txt	8.51	6.34	8.81	8.50	10.00	6.93	8.86
input04.txt	7.09	5.05	5.42	4.22	10.00	5.71	6.27
input05.txt	7.78	6.73	5.51	5.46	10.00	3.99	8.74
input06.txt	6.60	7.00	8.73	8.45	10.00	7.71	9.10
input07.txt	4.65	6.96	8.12	8.79	10.00	5.32	8.82
input08.txt	6.06	7.07	8.79	8.97	10.00	4.52	9.71
input09.txt	4.08	6.76	6.21	6.93	10.00	5.76	7.08
input10.txt	4.49	8.11	7.63	3.35	10.00	4.88	5.99
Sum	64.37	67.41	78.67	72.88	99.38	56.17	84.35

幕後花絮

Phantom	這次比賽一開始想要出的題目是 Traveling Salesman。 有一些都市讓幾個 salesmen 買賣貨品 (這已經不是原本的 traveling salesman 了)， 後來考慮到 salesmen 走路很辛苦，所以改成開貨車開來開去，而且幾台貨車容量都不同。 在跟我妹妹討論題目的時候她直接說：「這不就是鐵路大亨嗎？」真是一語驚醒夢中人， 所以題目馬上從開貨車的 sales 改成火車，外加貨物價格會隨著時間遞減。 既然題目是鐵路大亨，當然按照我印象中的遊戲設定要有 TGV 車頭，慢慢演變成 三種車頭各自有載運量以及火車車速，地圖本來要找鐵路大亨第一代的起始位置(英國)， 後來為了 TGV 的車速問題把地圖範圍擴大到歐洲 (英國地圖不夠讓高速火車衝來衝 去)。 在跟 Andric 討論題目的時候發現這題真的非常難，理論上要寫好的話比電梯還難， 畢竟三台一模一樣的電梯可以共用同一種策略，但是先天上三種屬性完全不同的火車 應該要用三種不同的策略比較合理。期間我還想說要在鐵軌上面動態出現「寶物」， 車子開過去拿到金幣可以加 gold，拿到加速 buff 可以加速 300%。或是讓貨物的賣價 採用浮動制度。或是讓玩家可以自己花錢鋪設鐵軌。不過鑒於電梯那年的慘烈狀況， 這些奇怪的想法最後通通捨棄了，反而是我跟 Andric 不斷的在想如何降低難度。 當然降低難度最直接的就是 (1)提供 sample codes (2)先給前三關供各組測試。 接下來開始做整個系統，為了抓那些火車的圖樣我還去找了第一代跟第二代的鐵路大亨， 不過最後還是請 Andy 幫我畫了幾張圖解決了我的困擾，專業人士畫的就是不一樣， 你 們大家應該不會想要看到我原本的那些火車圖。 然後當天早上為了比賽的特效我還去抓了火車的音效檔， 本來想要在各位寫程式的時候 一面在旁邊放音效增加一些臨場感， 雖然這怎麼看都是幸災樂禍的行為。 這一題我到現在都還沒想到太好的方法。 到底該用 Greedy? Dynamic programming? DFS? 各種流派眾說紛紜。我個人會選擇往前 preview 一定的 time window， 外加三五種的 heuristics， 沒實驗過不知道最後結果會如何。 比賽過程中我不斷的提醒各組，前三關的結果可以用手工刻出來，不過沒人做到。 星期六早上我起床之後為了證明自己說的是對的， 花了三個多小時手工刻了第一題的結 果，拿下 18542.90 分遙遙領先各組。 不過說真的，手工刻出結果真不是人幹的，尤其在精神狀況不好的要刻出對的結果真的很 難。 比賽最後的結果由 Henry + Shou_Lai + Yaoan 那組以接近滿分的 99.38 拿下冠軍。 而且偷偷告訴各位一個訊息，他們那組 Henry 跟 Yaoan 各寫了一個版本， 單獨拿 Henry 的版本出來也是冠軍， 單獨拿 Yaoan 的版本出來其實也是冠軍， 更不用說兩者取其優。 這次比賽的 10 個考題設計如下。 (01) 手工刻出 (02) 三天, 每天一開始 10 個都市通通長出貨物, 然後一整天沒有其他產出 -routing 能力去把貨物妥善載完

	(03) Rich resource: 每小時 1 ~ 3 個都市產生 3 ~ 20 個資源, 3 days -東西多到載不完, 所以需要精準計算載那個賺最多錢 (04) 極大量, 每 3 小時 1 ~ 3 個都市產生 20 ~ 40 個資源, 6 days (05) Poor resource: 每六小時 1 ~ 2 個都市產生 1 ~ 8 個資源, 6 days -能夠預先判斷把車開過去等, 時間足夠開遠點去賣東西 (06) 忽多忽少: (每 24 小時變換一次 type (多少少多少少共六天)) (07) 出資源頻率高, 但是數量少 (每小時 1 ~ 3 個都市產生 1 ~ 3 個資源) -考驗火車 routing 載貨能力 (10 days) (08) 端點多 (0,1,8,9 機率為其他都市的 5 倍) - 每 3 小時 1 ~ 3 個都市產生 1 ~ 10 個資源, 其中端點機率最高, 6 days (09) 中間多 (2,3,4,5,6,7 為端點都市的 2 倍) - 每 2 小時 1 ~ 2 個都市產生 1 ~ 10 個資源, 中間六的都市機率比端點高兩倍, 6 days (10) Poor generate with huge resource - 每 6 小時 1 ~ 3 個都市產生 7 ~ 18 個資源, 9 days By Phantom, 2010/09/15
Jace	有生以來的第一次程式設計比賽就這樣獻給了 Openfind(羞)，我們這一組採取了最無腦的策略， 因為正常來說要能動態考慮路線還要能同時考慮貨物買賣的最佳化，因此使得這問題變得很複雜。 經過兩個多小時的討論後，決定把路線跟買賣分開來處理的策略，三台火車就依固定的路線來回跑， 看 input 檔裡貨物的分佈，來安排路線，買賣的話也很簡單，因為路線固定了， 當每站的貨物也有個賣價標準，就是這個貨物載去端點站時的賣價，如果比目前車上最便宜的貨物高的話 就賣舊的，載新的，然後在路線端點站時都把貨物賣掉(這是為了解極端 case，像是只有一個點會出貨。 這個方法會使所得下降，而且後來也沒這個 case，哭哭)， 最後公布 4-10 題後，本來想照原定計畫…分析 input，排路線，try 高分的組合， 但是時間一急迫，腦子就轉不太動了(囧興)，所以最後就靠子興的 script， 暴力搜尋一組路線的最佳起始點，這組路線裡三台火車都是從伯明罕到莫斯科， a 是走最短路線，b 是走上面那排城市，c 是走下面那排，我覺得這組應該算萬用解啦，只是遇到第十題的表現就糟了。 星期六臨走前我手動挑了組路線來跑一下第十題，結果分數是比賽交出去的那組 output 的兩倍！！ 所以結論就是…我們這個方法挑路線比挑起始位置還要重要 Orz 如果我能定下心穩穩的把路線選好， 應該拿個第三名沒問題，真是對不起同組的 Heidi 跟 Zishin。
Henry	一開始看到題目是三台火車跑來跑去時，不禁讓我想起之前慘敗的「電梯大王」(詳情請參閱 2008 年比賽結果).....

不過今年的題目似乎比電梯還更複雜一些，三台火車速度容量都不相同，
每個地點間的距離和貨物的價格也有著微妙的差距。

但是好在這次有了 Andric 的 Sample 程式加上 Phantom 圖形化的 Checker，
在開發和 Debug 過程上都省去不少功夫(想不出策略時看看火車跑來跑去也滿好玩的 XD)

有鑑於去年一開始的策略有些漏洞，導致最後的分數沒有理想中的高，
所以這次一開始就花了不少時間測試來擬定出一個大方向(這個遊戲有必勝法?!):
「在最短時間內，得到最大獲利」(好像投資理財廣告的標語)

決定大方向之後就開始來做一些初步的估算:
打開「資源價目表.xls」把「資源價目表」、「都市最短距離」和三台火車的速度容量
用題目規定的計分公式算出 [從 X 城市到 Y 城市的資源可賣的價錢] / [所需的回合數]

結果可以得到幾個重點:
1. 火車 C 在滿載狀態下每回合可賺的錢最多
2. 從端點(伯明罕&莫斯科)運出的貨物平均可賺的比較多
3. 一般來說運越遠賺越多，但是例外的情況也不少

而轉化為策略就變成: 1. 讓每台火車處於滿載狀態
2. 盡可能去端點買賣
3. 應用實際的計分公式算出最佳目的地

有了這些策略後就可以開始實際 coding 了.....
不過一開始在寫火車初始地點時就有點卡關
最後硬是湊了一個算分程式出來，測試的分數也確實有所提升就算 ok 了
(不過結果證明效果還不夠好，造成前兩題比較短的 Input 分數輸了一些)

然後又做了一些基本的改進:
1. 可在同一回合 SELL & LOAD
2. LOAD 時由比較新的貨物先 LOAD
3. 在最後結束前 SELL 貨物
4. MyTrainCityPath 修正
5. 未滿載時從經過都市上貨

做完這些後竟然已經是晚上十點了.....orz(此時開始有人準備回家了)
結果這時才要開始做重點策略.....(抱頭)

總之，最後花了約 3HR 把最重要的一段寫完(效率出奇的高?)
測試成績也比從 Tony 那偷看來的數據好....(逃)
於是乎就安心的回家睡覺了(AM 1:XX)

第二天來到公司時已經有不少人了
一開始寫了一個測試的策略結果分數反而大幅下降
然後也做了避免搶貨物和調整火車順序的功能，不過效果似乎有限
(每次比賽第二天來都會開始越改越爛....@@)
最後就是測試->調參數->Debug->記最佳成績的無窮迴圈.....

	比賽的結果..... Yaoan 的策略在較短時間的 1,2,5 題拿到不錯的分數 我的則在其他題拿下意外的高分(剛好互補 XD) 在小賴整合的結果之下，最終幸運得拿到接近完美的成績 以上(好像有點寫太多了，不過應該沒有去年家民的心得長吧 XD)
Mindi	有鑑於聽說過大家參加電梯比賽的情況， 加上覺得這次的題目跟電梯滿像的 所以一開始設定的目標就是先寫出能動的然後比 Andric 的 sample code 分數高就好 XD 下午和隊友們討論過之後就分頭實作各自的策略 因為我想說火車最主要就是要決定兩件事情 一個是要去哪載貨 一個是要去哪賣貨 所以就想了一個基本的策略就是到最近且有貨的城市去載貨, 載完之後再到可以賺最多錢的城市去賣貨 然後三台火車很偷懶的都用一樣的策略... 然後因為怕弄的很複雜所以沒有作預測， 只有在一開始找啟始位置的時候, 先掃過整個 input 找到前三個有貨的城市然後依照貨物數量放火車 後來再加些調整像是載貨或賣貨的沿途都可以順便載貨, 找賣貨的城市的時候多考慮時間結束前還開不到的就不去 晚上終於把自己的策略實作出來後前三題分數都有比 Andric 高就心滿意足了~ 在調整的過程中感覺到策略的好壞分數真的會差很多 不過也不知道別組跑的情況所以也無從比較&擔心起... 看到自己分數一直有進步就很開心 XD <- 自我感覺良好(?) 第二天早上改了很多晚上回家想到的小調整像是載貨載到中途滿了就改成賣貨, 找載貨城市時考慮該城市的貨物期限等 不過分數一直停滯不前毫無進展...跟前天晚上作出來的結果都差不多 心想這個基本策略的極限也就到此了吧||b 一直到後來用 Phantom 提供的介面觀察火車跑來跑去的情況 看到第三題我一直很心酸的載著過期的灰灰貨物到處跑... 才終於發現我漏掉了一件很重要也很基本的事情~ 我每次都從最舊的貨物開始拿(直接照 call Andric 的函數沒仔細看...) 應該要從最新的貨物開始拿才對(發現 Andric 其實有提供這個函數只是沒有用)! 然後分數終於又有了大躍進 XDD 然後後來聽到 Phantom 第一題用手刻的分數和自己跑出來的也沒差很多心裡有比較安心:p 因為今年只要交 Output 所以就用 megaman 寫出來的幾個版本+我的幾個版本, 都跑跑看拿最好的交 遇到第五題跑不出來的時候還超緊張的 最後跟大家一起看每組跑的結果 有了去年的經驗加上心裡大概知道自己的分數, 所以就沒有去年那麼緊張 然後覺得 Henry 那組實在是太強了~!
Bart	這次比賽我們這組雖然有四個人，可是兩個人從中午拿到題目都有事情在忙， 所以一開始只有新人 Kevin 的可以測，雖然我想說用手刻出。

	結果，後來花了兩小時多，還刻錯，後來只好等 Mike 寫出來他的版本， 如果 Mike 早點寫出來我們就有時間調教了，哈哈，可是大家都很認真， 智中後來也把 sample 的調參數改了一下，雖然拼 24 小時沒贏， 可是好玩就好啦，還是 Henry 默默跑回家默默做厲害。
Tony	以往的程式設計比賽，常常都會想了太多如何逼近完美的做法， 而導致最後程式寫不出來； 這次大家聰明了，先求有，再求好，一隻程式不行的話，再生另一隻出來， 每隻程式有針對各種不同 testcase 的特別解法； 最後 Felka 跟 Willie 做出來接近 10 個板本， Josh 也提供了負責測試的程式，讓我們這組能在最後十分鐘， 用 10 個版本跑完所有題目，並自動的挑出最佳的解法。 不過整夜都沒睡，真的太累了@@
Willie	這想不到才剛進 Openfind 不到兩個星期，就可以參加這一年一度的盛事， 真是備(ㄅㄟˋ)感(ㄏㄢˊ)榮(ㄖㄨㄥˊ)幸(ㄒㄧㄥˋ)。 一開始在討論戰術的時候其實也想了不少策略，但是在寫到 9 點多終於有個可以跑的版本時， 三個 Input 分數都不到 1000，真是悲劇... 在修修改改之後， 好不容易出現個可以跑贏 sample 程式的數據，人已經呈現半入眠狀態。 在出了幾個修正 BUG 的版本之後就掛白旗回家睡覺了，實在是對不起隊友們。 隔天到辦公室時發現別人的分數都是我昨天寫出來的兩倍時，真想找個洞鑽進去。 幸好 Felka 前輩幫忙改了兩個部分之後，分數整個大躍進，打消了我偷溜的念頭(噓)。 這次的比賽實在是學到不少東西，雖然 Felka 前輩一開始跟我說先別管 coding style， 先做出可以跑的版本就好，我也就照著刻出最簡單的版本。 雖然說不這樣刻也許連第一版都生不出來，但是這樣的版本要再增加策略真是找自己麻煩， 更不用說刻完之後神智不清的狀況。 反觀其他人的程式碼，完全可以體會自己刻出來的有多難維護，還是需要再多磨練。
Mike	又到了一年一度的程式設計比賽，往年都是辦在飄月， 今年 Phantom 很好心的等飄月過了再辦，看來難度就不再是鬼神等級， 而是人可以寫得出來的吧? 今年的題目滿好玩的 - 火車大亨。 拿到題目的當下，看著範例程式的火車跑來跑去的可愛模樣， 著實讓我不忍心破壞這唯美的範例程式，所以心中升起要去干擾別組完， 然後回家好好睡一覺，隔天再來干擾別組， 最後把這可愛的範例交出去比賽的念頭。畢竟我們有超 D(編按：超級 Deliver)的智中， 超強的新人 Kevin 以及超強的 QA Bart，這不就是叫我回家睡個好覺了嗎…(灑花) 於是乎，我們這一組的策略就是大家各自寫自己的策略， 最後跑的時候，再拿分數高的出去給別人笑一下。 策略部分從簡，因為我沒什麼策略，所以這部分就看隊友有沒有寫什麼了 XD

	最後我們是一共寫出 6x 個版本去 run，再取最佳的結果。 結果只有倒數第二，看了一下結果，有三、四個 case 是最低分，而且差距頗大， 看來程式上有很多 case 沒有想到，下次要注意。 感謝隊友的不眠不休，至少讓我們沒有殿底，大家辛苦了 ！
Honesty	這次有幸與 Jimmy 及 Venser 同隊奮戰 ， 充分感受到 Jimmy 垃圾話干擾他隊軍心的強大能力，及 Venser 埋頭苦幹的拼勁... 可惜一開始我的野心不小，打算寫個預先算分的架構， 策略設定的頭一目標就是.....完全把 andric 的範例程式擱一邊 ~ Orz 寫著寫著，發現變因實在太多，成效並不特出， 直到最後當初大家說好要準時回家的約定時間將屆， 才回心轉意發現 andric 範例程式雖簡單卻在這次複雜的各項變因中也有不少點可以發揮... 亡羊補牢猶未晚也，幸虧 Jimmy 大大連夜將 andric 範例程式修改成我們的一套取決方式， 第二天一早，我們再針對開始的佈陣下功夫分析， 並根據貨物產生時間地區不同，動態調整各列車負責的城市... 這次有了 console checker，我們也利用 perl 寫了個可以執行多版本程式的架構， 根據不同參數調整的各個版本都跑過一次，選擇一個分數最高的版本交出... 最可惜的是最後才發現一個致命的 bug，沒能及時好好利用 sold 及 load 的一次動作， 並且還有其他想法例如亂數動態配置的版本也就差那麼一點點沒能及時趕出... 雖然這次不幸落敗了，但還會有下次機會的！ XDDD
Jimmy	這次的程式設計比賽跟同隊的人討論得很開心， 想到的解法很好但是太難實做了， 最後時間不夠，不過過程是相當有趣的! BY 宮城良田
Shenqyang	這次程式設計大賽很高興和兩位功力深厚的 RD 一組 也因此在第一次參與比賽就得到第二名的佳績 也借此機會了解到分工合作的重要， 雖然這次比賽我的貢獻不大 但願以後有機會能加強這部份的能力 以成為團隊中的最佳戰力
Zishin	來 Openfind 快一年，第一次參加公司的程式設計比賽覺得真是非常緊張刺激阿 XD 題目一公布，腦袋就開始打結，很明顯是個策略導向的題目， 可參考的變因完全超乎人腦可以負擔的範圍。 跟 Heidi 和 Jace 在會議室腦力激盪了兩個多小時，最後決定化繁為簡， 把很多變因拿掉，並且定義出一個明確的主軸： "每到一個城市如果有貨可以載，就把車上的貨賣掉改載新貨， 目的就要讓車廂一直載有新鮮貨。" 其他的變因(路線、起點城市)就交由人腦控制 XD 決定工作分配的時候也是一件不容易的差事，題目需要實作的部份感覺不太多， 很難有明確的項目可以切開來分頭實作。 所以，最後還是交由 Jace 大大繼續幫我們分析 case 和想策略，Heidi 跟我則開始刻程式。 散會之後，回到位子上開始研究 andric 的 sample code，

	發現整體架構跟我們的策略有一點點的出入。 頓時有種不知道該從哪裡下手的感覺 XD 臨時又找了 Heidi 跟 Jace 討論，最後決定還是要改掉 sample code 的架構， 來達到我們的可以指定路徑的構想。臨時動議後，工作分配又稍微改了一下， Heidi 以 smaple code 出發再想策略，我則是把我們的策略實作出來。 晚餐後就是 coding 時間，這時候 RD 這半邊充滿了各式各樣搞笑的垃圾話， 讓整個比賽氣氛變得很歡樂 XD 我們的第一版在很專心的情況下快速趕出來，我們三個對結果都覺得很滿意， 所以就決議第一天就到此為止，收工回家睡覺明日再戰。 隔天原本我只打算做個暴力的 script，來決定火車的起始位置， 看看可不可以有更好的結果。 不過，第二天就遇到一個奇怪的 bug，花了一點時間小改版，讓整個架構更穩定。 就我們的架構而言，路線的決定跟火車起始位置對於結果的影響程度非常大， 在載貨的策略相同時，同一個 case 有可能可以拿到幾萬分， 也有可能只拿到幾千分。這時候暴力法就是一個不錯的手段 XD 在所剩時間不多的情況下，努力把決定初始位置的暴力法趕出來， 效果果真不錯。可惜實作時，做得不夠彈性， 沒辦法很容易修改火車行進路徑。 最後的測資出來，發現自己搞的暴力法不夠自動化， 在這種時間壓力大的情況下來跑實在太緊張刺激， 邊跑 script 邊按滑鼠鍵盤分析結果的同時，手也一直抖， 還好心臟還夠力，有把結果跑出來 XD 這次的程式比賽，很可惜沒有擺脫傳說中雙數組無法晉級前三名的詛咒 XD 不過，跟隊員們溝通順暢，合作愉快，是一次很愉快的團隊合作經驗。
Shawn	一看到比賽題目，馬上聯想到的是以前電梯那題，傳說有人使用隨機亂數的策略得到不錯的成果。 在寫好框架之後，隨即試著採用隨機策略，雖說是隨機，不過也只有行進方向是亂數決定的 ： 一開始的火車位置會載貨量選擇最適當的起點位置來等待， 產品的賣出則是根據所持有貨物的價格趨勢決策，價格下降之後便賣出。 這個隨機行走策略在老楊的提供的隨機 Input 產生器跑出的結果之下， 沒想到能比一般策略的結果好上不小。然而隨機終究是隨機，結果好壞相差極大， 要跑到好的結果有時候需要跑好幾十分鐘。 由於隨機策略看起來太耗費時間，便開始由隨機策略作改良： 行進的方向以會根據所載運貨物抵達鄰近城市的價格或即將產出的貨物價值來決定， 卸貨時機則判斷該時間點是否已達該商品最大價值，並判斷是否為最後可賣出的機會。 而此策略看來成效也不錯，雖然還可以改良到考量到更多步之後， 不過可惜到此時已經神智不清只能做些微調了。 然而我雖然有兩種策略，最後跑結果的時候，大部分靠的還是 Mindi 在早上突然改出的超強策略， 更不用說隨機策略其中有個最佳的結果，卻因為第五題突如其來的意外導致手忙腳亂沒派上用場。 這次的比賽能夠獲獎主要還是因為有優秀的隊友，過程也是相當有趣的，雖然很累一切都

	是非常值得的啊！