

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。2008 年 Openfind 程式設計大賽花絮如下：

題目：Openfind 電梯大王. 比賽時間：24 小時

競賽規則：[Openfind 電梯大王比賽題目及考題下載](#)

比賽結果名次公布：

- 第一名：Heidi, Michael, Bart
- 第二名：Mike, Jonas, Tony
- 第三名：Berit, Wallace, Lex

戰況剪影：

	Round 1		Round 2		Round 3		Round 4		Round 5	
	M-S	N-S	M-S	N-S	M-S	N-S	M-S	N-S	M-S	N-S
Team1	0	0	0	0	0	0	0	0	0	0
Team2	4	7	3	7	7	7	4	7	3	7
Team3	0	0	0	0	0	0	0	0	0	0
Team4	7	5	4.5	3.5	6	5	5	5	4	4
Team5	0	0	4.5	3.5	5	4	0	0	7	3
Team6	6	4	6	5	3	3	7	4	5	5
Team7	5	6	7	6	4	6	6	6	6	6

Round 6		Round 7		Round 8		Round 9		Round 10		Total	
M-S	N-S	M-S	N-S	M-S	N-S	M-S	N-S	M-S	N-S		
0	0	0	0	0	0	0	0	0	0	0	
4	7	3	7	3	7	3	7	7	7	49.7	(2)
0	0	0	0	0	0	0	0	0	0	0	
6	5	6	4	4	3	4	3	3	3	46.8	(3)
0	0	7	5	7	6	7	5	5	5	39.2	
7	4	4	6	5	5	6	6	4	4	50.9	(1)
5	6	5	3	6	4	5	4	6	6	54.4	NPC

幕後花絮

Phantom	Dear all, 今年的題目是一棟 18 層大樓有三台電梯，寫程式去控制三台電梯讓使用者等待時間最短，讓能量耗損最短。 依照往例請各參賽人員於一週內寄送 幕後花絮 給 Phantom，永久將這些紀錄保存於公司網站上面。 不管結果如何，這些都是大家的心血跟奮鬥的歷程.^_^ 由於題目實在太變態，遠超過我的估計。自己下去寫了以後才發現題目難過頭了。經過 24 小時的 廝殺以後(幾乎大部分的人都留在公司沒睡覺)，比賽結果如下。 NPC 隊伍不算的話。 冠軍隊 => Heidi, Michael, Bart. Heidi 以新人之姿寫了一個最標準的電梯程式，就做掉了各支隊伍。Michael 躲在家裡做原子彈， 果然程式一鳴驚人，Bart 拿著精心製作的 test cases 到處追殺躲在家裡的兩位隊友，順便在公司收集情報。原來什麼策略都不用，只要正常電梯開來開去，有人就去載，這樣就可以第一名了. @@ 亞軍隊 => Mike, Jonas, Tony 這組說：因為三台電梯會出 bug，所以我們只用了一台電梯，然後其他兩台休息..... 什麼跟什麼， 只開一台電梯就拿下第二名。只以些微比數落後給冠軍隊 季軍隊 => Berit, Wallace, Lex 這隊伍更猛，三個人寫了三個版本，而且有的策略很極端。居然是用 random!! 用亂數也可以拿下第三名。 這個世界還有天理嗎？ 當然賭盤來說的勝利者是第七隊 NPC，由 Andric 跟 Phantom 組成。我們本來想了六種 Greedy 的策略， 但是由於 Andric 忙著寫裁判程式還有幫各隊伍教學，Phantom 這星期已經是第三天睡在公司體力不濟。 最後早上六點才開始動工(十二點要交卷)，六種 Greedy 只夠時間寫出一種。僥倖小幅領先贏得外圍賭盤。 最可惜的是第五組 Josh, Eric Chang, Dean。他們十個 test cases 只完成七個。不過那七個都拿下超高分。 如果十個都有完成的話，連 NPC 隊伍都遠不是他們的對手。 各位辛苦了。我下次會出「更」簡單一點的題目。 Phantom
---------	---

Bart	幕後花絮: 本次能勝出都靠黑馬 Heidi 做出個穩定的電梯，讓我們這組能勝出，可惜的是 Michael 的原子彈沒做出來，可是他也發揮了嚇唬各組的功能，而我在前半段完全不知道他們兩個做的怎樣，只能想想 TestCase 和看看每組做的怎樣，看來我和 Michael 都剛剛好發揮心理作用，最後讓 Heidi 安安靜靜的完成了他的穩定版，而讓我們些微之差贏，有點僥倖，不過也看到各組努力合作的精神，大家其實都很拼，下次禁止NPC參加賭盤，大家努力才不會白費。
Lex	以下為心得報告內容: 大家好，我是第四組的 Lex。 剛來公司第一個月就參加了程式設計比賽，讓人有點慌了手腳。不過我還是盡力生了一個能運作的程式出來。 一開始本來打算用 C++ 寫的。但是由於題目充滿了麻煩的條件，光是要記錄每個電梯的 State 就花了我不少時間，而且由於自己不熟悉 C++ STL，浪費了不少時間走冤枉路。 到了剩下 12 小時的時候，眼看連一個能動的程式都生不出來，於是心一橫，改用自己拿手的 Ruby 來寫，沒幾個小時就完成一個能通過驗證的程式(雖然還是很醜)。Ruby 是我最喜歡的一個程式語言之一，有興趣的人一定要試試看。由於時間不足的緣故，我的程式採用了隨機演算法，在無法下定論的時候程式會隨機採取一個行動。因此電梯會隨機決定要不要開門，沒人的時候會隨機決定要向上還是向下。跑個幾百遍再選一個最好的結果。 這個策略證明是可行的，只是通常結果不怎麼好看。最後結算的時候，大部分的 TestCase 我的程式都只能給出非常差的結果，只有 Case 8 我的程式給了一個比所有人略好的結果，因此整場比賽我可以說沒幫上什麼忙。不過這也說明在某些極端的狀況下，隨機搜尋會比固定的策略來的有威力。 這次比賽真的很累也很開心，能和這麼多厲害的高手切磋真的是很難的機會，也期待下次比賽的來臨。謝謝。
Heidi	這真是個爆肝的比賽！XD 第一次參加，其實有點誠惶誠恐，心想：我什麼都不懂，只能靠另外兩個組員安排了 XD Bart 說要做他拿手的測試，Michael 說按照往例，寫出來會動就贏一半哩～ 所以為了分散風險，最後決定 Michael 和我各自去寫基本最笨但會跑的程式 原先估計晚上應該可以先寫出來最笨的吧！

	結果當然不是這樣嘍～是比賽前沒多久才真正寫出來 XD 本來以為針對笨笨的方法已經想的差不多了 真正在 coding 的時候才又發現有很多細節還沒有考慮到 修修補補，回家之後也是想啊想、寫啊寫的 天都發白了才去眯一下，隔天早早出門去公司再繼續奮鬥= =+ 感謝 Bart 的測試，陸陸續續發現一些 bug 漸漸解掉之後，程式看起來就有比較正常點了～哈哈 比賽的時候俺還驚呼：竟然沒有當掉耶～XD Michael 也很辛苦，應該整晚都沒睡吧！ 這次能得到名次著實是意料之外啊～(沒想到這樣也能贏 XDD) 感謝兩位隊友正確的決策和測試上完整的支援 讓我們有不錯的成果～很感恩的啦！:D ----- 以上就是心得嘍～ 謝謝～ by Heidi
Henry	心得: 慘敗，這次當作學個經驗，明年再來吧。 by Henry
Jimmy	第一次參加程式設計比賽，雖然成績不理想，不過很好玩、很有趣。明年有機會一定要參加!!
Jonas	當聽到公司要舉辦今年的程式設計比賽，心理就一陣興奮... 沒錯！又可以再來一次那種跟隊友一起腦力激盪，分工合作把東西衝出來的感覺啦！ 自從高中畢業之後就很少有這種團隊競賽的感覺了，沒想到到了 Openfind 還可以重溫這種快感... 當然啦依照慣例，程式設計比賽沒有在睡的啦！反正大家都還那麼年輕 (?)，撐一個晚上不睡 OK 的。 看到今年的題目是電梯大王之後，直覺反應是...哇咧電梯系統不是老問題嗎？ 不過想當然爾，Openfind 程式設計比賽不會那麼遜的，依照慣例一樣有一堆參數還一些怪怪的變化條件。 Phantom 賽前一直再重複：『這次的題目那麼簡單， 20 分鐘之內就可以寫完的啦！』，不過大家都知道一定不是那回事。 果然看到題目之後... 20 分鐘要寫出一個結果 OK 的版本好像有點拼咧？ 話說比賽到一半，Phantom 從旁邊經過還在碎碎唸：『靠怎麼那麼難寫...』喂喂喂...不是 20 分鐘寫完嗎？ 在當天中午公佈題目之後，我還想說下午認真工作，下班之後再來衝比賽。 不過下午到了一點半...兩點...兩點半... 哇咧？人怎麼都不見了啊！？ 原來是我太天真了，各隊幾乎都中午就開始衝了 Orz 所以我們這隊就趕快招集起來，開始衝比賽啦！ 賽間不時有人一直放風聲『這次大概跑得出來結果的話應該就有前三名吧！』... 話是這樣說的嗎 XD 剛開始的時候我們思考了一些方式，找了一些相關 Paper 來看，不過看看比賽時間和題

目複雜度，還是 Greedy 最直接吧！

反正先用最笨的 Greedy 跑出結果 (跑得出來就有前三名是吧?)，再慢慢調整一些 Greedy 方式應該就夠了。

搞複雜演算法的話，到時候時間不夠跑不出來半個 OK 的結果的話，豈不就應驗別人說的『跑得出來就有前三名』了 XD

這次真是辛苦 Mike 了，在我們討論出來方法之後，他就負責主要部分的 coding，但是他似乎平常沒有訓練熬夜，到接近天亮之後已經近乎神智不清了，所以這故事告訴我們...平常偶爾熬點夜還是有點幫助的 :P

而 Tony 不愧是 QA，早就準備好 test cases：『你們有可以跑的版本可以測了嗎？可以測了嗎？可以測了嗎？』

oh my god，沒想到比賽也會被討債，『抱歉，你先咪一下等我們 de 完 bug 吧 Orz』

好不容易第一個可以跑出結果的版本是天要亮的時候才出來的，用題目的 sample case 下去跑，嗯嗯看起來不錯喔！

再放其他幾個 test cases 下去跑... 挖勒 checker 怎麼一直跟我說結果錯誤啊 Orz

之後 de 掉幾個 bugs 之後，剩下一個 FIFO 的問題搞不定，最後想到是邏輯上有點缺陷，可能會造成違反 FIFO 原則的問題 Orz

看了看剩下的時間，如果要修好的話時間上有點危險，所以只好祭出大絕招：

拿三個版本，各跑一個電梯，兩個電梯，三個電梯，最後在三個結果內選一個 checker 驗證過且結果最好的來用 XD

其實當初看到這個比賽評審方式的時候就有想過：

我們其實可以寫 100 個版本各用不同的演算法，到時候拿到 input cases 的時候就用 script 下去跑所有的程式，

最後自動把結果正確且最好的選出來交出去就好了。

甚至有些決策點如果想不出好方法，或是判斷太複雜，或是會讓程式跑很久的，那邊乾脆就用來個 random，

只要讓程式跑得快，到時候就跑個 1000 次選個最好的結果就好啦！

其實這也算是個不錯的方法啦，不過很偷雞就是了，產品這樣寫的話會被砍頭的 XD

總之到了最後評分階段，因為有的 test case 我們只用了一台電梯或兩台電梯，

所以有些 case 所耗費的時間，真是高到自己看了都會不好意思 XD

不過我們這組的電梯整個是『節能減碳型』，雖然時間上輸的蠻多的，但是能源花費部分在每個 test case 都是最優的 :P

不過你要問我為什麼會比較省電我是不知道啦，不過我可以給你一個猜測：

如果公司兩台電梯關掉其中一台，那大樓電費支出可能會少一點 (不過你可能上個四樓要等 20 分鐘電梯) ...

基於環保訴求的『節能減碳電梯系統』也因此讓我們拿到了第二名，喔喔喔喔喔...這算是賽到嗎？

總之，這次的故事告訴我們：

- (1) Openfind 程式設計比賽很好玩，請大家多多參加不要請假
- (2) 平常偶爾訓練熬點小夜，總有一天會派上用場的
- (3) 節能減碳還算蠻重要的，請大家多多響應
- (4) 沒事不要罵電梯很笨，電梯也是有他的苦衷的

Raul	這次的程式設計比賽，很遺憾的，本組完成的不夠確實，雖然程式可以正確無誤的執行，可是跑出來的結果確沒有一個可以確實的拿到分數。之後自己在檢討時，敗因可以歸納成以下兩點： 謀事不臧：在一開始討論時，關於要怎麼讓三台電梯協同合作即沒有一個很明確的結論，最後我們是決定採用瀑布式的開發法，先寫一個較粗糙的版本出來後，再慢慢改正，怎料，光是這個粗糙的版本就已經用掉了我們所有的比賽時間，根本就沒有改進的時間了。 力分致敗：因為本組採用 Java 來實做這個題目，但是只有我跟 Jimmy 比較熟悉這個語言，後來在實作時，Calvin 協助 Jimmy 一同開發，我這邊則是自己埋著頭苦幹，到最後，我這邊的版本一直有缺陷，無法順利的跑出結果，而 Jimmy 那邊，雖然程式可以正確的執行，可是跑出來的結果一直不能盡如人意。 也因為以上兩點，導致我們的比賽結果非常的難勘，最後雖然有交卷，可是無法得到任何的分數，這其實跟沒交卷是一樣的。 雖然這次的程式設計比賽結果不盡如人意，可是對於我來說，透過這次的比賽，一方面認識了更多的同事，一方面也是對程式能力的加溫。而 Phantom 在之後的電子報對於這次的比賽情形更是分析的鞭辟入裡，也讓我受益良多。雖然比賽已經結束了，可是我還是會找時間，打敗電梯大王這個大魔王，並且，我也開始期待下一次的程式設計比賽了。
Berit	一年一度的程式設計比賽終於來了.比賽之前整個公司謠言滿天飛,傳說今年的題目超簡單,簡單到出題的 Phantom 只需要花二十分鐘就可以解決,不過大家都不驚訝,因為 Phantom 似乎每年都說超簡單,事實卻常常不是這樣... Wallace 跟 Lex 還有我三個人同一組,我們原本的計畫是先花兩小時各自寫個簡單的版本,再一起討論較為複雜的作法與不同策略的優缺點.但真正開始實作之後才發現問題比想像中要來的複雜,於是開始了三人的閉門會議.經過一番討論,將程式流程中較複雜的部份清楚的畫在白板上,並擬定作戰策略,我們決定多個版本不同策略同時進行.實作之後,Lex 的 random 在實測上似乎都可以得到相當不錯的結果.原本信心滿滿的我們,卻在最後變的有些手忙腳亂,因為比賽的 test cases 不太適合在 random 版本上跑.好在我們版本多,一個不行還有一堆可以用.最後名次揭曉得到第三名,辛苦總算是有了代價. 今年的程式設計比賽是我第一次參加比賽,感覺相當緊湊且有趣,過程中也學到了很多.
Tony	這次比賽的題目，真如 phantom 所說非常"簡單"啊，各組的人幾乎都開心的在公司渡過了一晚，從以往程式設計比賽的經驗，我們認為只要可以動，基本上就會有機會。不過題目實在太難了，最後準備的幾個複雜的 testcase 都沒派上用場，因為跑基本的都有問題@@。最後結果是由 jonas 所研發的以一部電梯進行載運，就可以順利跑完；最後感謝兩個辛苦的隊友 mike, joans，在這次的比賽中，辛苦的寫程式。

Mike	Sorry for late.....囧 這一次的程式設計比賽，題目滿有趣的。不過似乎大家一開始在 phantom 的"刻意誘導"下，都真的覺得一天內似乎可以寫得出來，但是在深入討論後並實作後，才發現這是個大黑洞...囧。 所以所有的組別都在公司待了一夜。比賽結束時，大家的眼袋都比眼睛還大。XD 我們這一組因為擁有上帝的頭腦之稱的 Jonas，故最後雖然我這一版本的有 bug，還是靠著 Jonas 一人之力，硬是奪得榜眼。而 Tony 哥也很辛苦，硬是一整夜未闔眼在寫 test case. 想來想來覺得自己真的是太弱啦，看來要好好的操練一下，免得成為零戰力。 程式設計比賽還真的滿有趣的，不過，如果題目可以"人性化"一點，不要每次都這麼變態，就更好了
Michael	這次的題目第一直覺就是很麻煩，而且如果真的要最佳化的話也有相當的難度 -- 如果是 online （無法預先知道乘客數量、何時到）的話反而簡單許多。 因為不容易想到好方法，所以決定 Bart 先想幾個 test case，我和 Heidi 則先各自寫一個版本，有餘力再尋求改進，事後證明這也是個不錯的策略：由於這次比賽要交的是 output，所以分頭寫出多種版本、比賽時挑出最好的結果會是不錯的方法。（還可以降低風險） 雖然計畫是先做個最簡單、幾近 online 版的，不過真正動手前時還是想太多，為了之後修改演算法預留彈性、為了可以人工修正.....，後來自己訂出了一個中間層--定義出比較抽象的電梯動作（相較於題目輸出用的電梯動作狀態碼）以及從抽象動作輸出成題目要求的輸出格式的 function。做這件事的目的是希望能讓電梯演算法的輸出簡化（不需要處理 clock by clock detail），而且有人類易讀/寫的中間格式，必要時可以以人工介入修改結果--比起直接修改最終輸出簡單。事後想想，這個動作除了增加不少工作量，也是 online 版演算法難產的原因--既然是 online，直接考慮 clock by clock 就好了，為了輸出成抽象動作，反而得做部分的 look-ahead--結論就是為了不見得寫得出來的「較佳演算法」搬了塊大石頭砸自己的腳。 最終就只有中間層轉輸出這部分可以用，本來打算如果有小的 input case 就用人工排程硬做，可惜並沒有夠小的 case。 幸好其他兩位隊友沒有發生這種搞笑的狀況，順利完成原訂的初版，還做了改進版本，測試穩定而且還有不錯的結果，就如 Phantom 所說：關起來做原子彈--只做出彈殼--的情況下靠著隊友得到這次的成績。