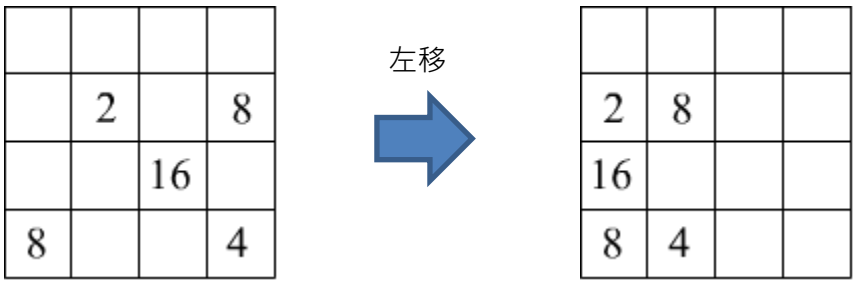
## 題目說明

你的任務是撰寫一套會自動玩 2048 的程式(簡稱 bot), 2048 遊戲規則如下:

1. 遊戲由一個正方形矩陣組成, 每個格子中可能裝有數字方塊, 也可能為空格.

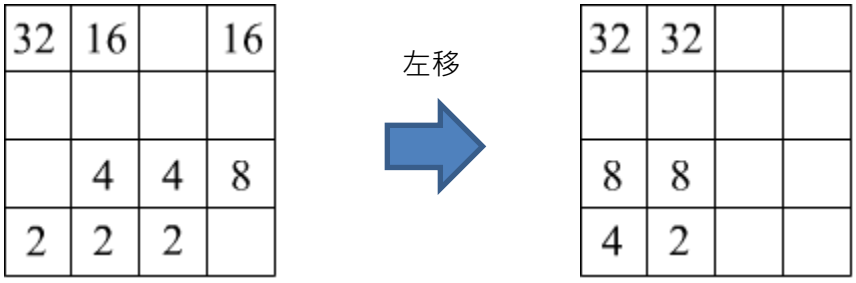
2. 每回開始時, 玩家必須指定一個方向. 每個數字方塊將沿著此方向滑行, 直到遇上邊界或另一個方塊才會停下.

   例.




3. 當數字方塊是被與自己同數字的方塊停下時, 兩個方塊將合併成一個新的方塊, 數字為兩方塊的數字和.
   合併而成的方塊不得在同回合中再與其他方塊合併. 合併的順序與滑行方向相反. 例如, 當玩家指定左移, 所有方塊將從右往左移動, 則左邊的方塊將優先於右邊方塊合併.
   每當兩方塊合併, 你將得到和新方塊上的數字相同的分數.

   例.



4. 若至少一方塊在此回合被移動或合併, 則遊戲將在隨機空格內產生一個新的數字方塊. 新的方塊可能為普通方塊(數字為 2)或外卡方塊(數字為 4).

5. 當沒有一個方塊可再被移動或合併, 則遊戲結束.

## [ 競賽內容概要 ]

遊戲本身利用一個 class 來實現, 提供了六個基本的 public function 讓你的 bot 可以與之溝通.

- reset() –
  呼叫此函式等同於遊戲中的 " 重新開始" 鍵.

- getCurrentGrid() –
  此函式將會把遊戲中使用的矩陣拷貝到你所提供的矩陣中. 給予 index 即可讀取矩陣中的數字方塊, index 與矩陣格子位置的對應圖如下:

| | | | |
|---|---|---|---|
| 0 | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 |
| 12 | 13 | 14 | 15 |

- insertDirection() –
  你的 bot 可透過此函式指定方向滑行遊戲內的方塊

- isGameOver() –
  此函式可用來判斷當前的遊戲是否已經結束

- getScore() –
  你的 bot 可利用此函式得到遊戲內當前的分數

- getMaxScore() –
  你的 bot 可利用此函式得到目前為止達到過的最高分數

當 destructor 被呼叫, 遊戲將輸出一個紀錄了遊戲場次數、最高分數、平均分數、最大出現過的數字方塊、出現過 2048 以上數字方塊的場次數、所花時間以及有效動作數的文字檔. 此文字檔將作為 bot 的排名依據.

## [ 如何開始 ]

首先, 你必須從官方網站的下載區內下載程式碼封包, 檔名為”2048_cpp.zip”.
封包內包含了以下三個檔案:

1. 2048.h
   此檔案提供了兩個 class, Game 及 Grid 的介面與部分 inline function 的定義. 開放給參賽者使用的函式包含了前述 6 個基本函式與一些 Grid 的簡易功能.

2. 2048.cpp
   此檔案包含了主要的函式定義. 必須注意的是, 此檔案中部份函式定義將與最後用來編譯你的主程式的正式版本有些微差異. 這些差異存在的目的是為了保障競賽的公平性, 避免參賽者直接取用特定資訊. 詳細的說明請見”保密資訊”一章節.

3. main.cpp
   此檔案為一簡單的範例, 示範如何運用上述兩個檔案內的功能實現一款用 AWSD 鍵操作的 2048 遊戲. 此檔僅具參考功能, 將不會用來編譯你最後繳交的程式.

當你下載完這個封包, 便可以開始利用 C/C++撰寫屬於自己的 bot. 最後的成績將完全取決於遊戲最後輸出的文字檔. 要成功地輸出此文字檔, 你必須 include 封包內的標頭檔、利用會呼叫 constructor 的方式產生一個型別為’Game’的物件, 並保證程式結束前呼叫到 destructor.

最後, 當你的程式都準備好了, 你必須將程式碼壓縮成一 zip 封包並到官方網站上的檔案上傳區提交. 我們將替換掉封包內的 2048.h 檔及 2048.cpp 檔並統一由官方的 makefile 編譯你的程式. 記住, 封包內只需要包含程式碼, 編譯好的程式及 makefile 不需要提交.

## [ 計分排名 ]

每位參賽者的 bot 表現將由連續玩 100 場遊戲的結果來評比. 你必須設計讓你的程式用同一個 Game 物件玩恰好 100 場遊戲, 並讓它輸出文字檔. 為作業方便, 我們將不輸入任何參數執行你的程式, 請確保這樣便能達到上述所提及之功能.

上一個章節中提到了, 你的 bot 之排名將完全取決於最後輸出的文字檔, 那我們將如何解讀此文字檔呢?

- **正確性**

  第一個被考慮的要素為正確性. 正確性由下列四個項目決定:

  1. 你的程式碼必須能夠被成功編譯
  2. 編譯出來的執行檔不得發生 crash 或 hang 的情況
  3. 文字檔中記錄的遊戲場次必須恰為 100
  4. 程式的執行速度必須在平均每秒 100 個有效動作以上

  只有正確的程式將被視為有效提交的作品.

  所有有效提交作品將再由四項指標決定排名.

- **最高分數**

  100 場內的最高分數將用來當作排名的第一項指標.

- **平均分數**

  100 場玩下來的平均分數將作為排名的第二項指標.

- **最大數字方塊**

  100 場遊戲中曾經出現過的最大數字將會被用來當作排名的第三項指標.

- **破關率**

  出現過 2048 方塊的場次將被當作排名的最後一項指標. 當此指標達成平手, 則取 4096 場次數來決定勝負, 再平手則取 8192 場次數決定, 依此類推.

下表內為各指標排名所對應的積分. 我們將取積分排名前三名為優勝隊伍, 第四名至第三十三名為佳作隊伍.

<table>
<tr><th colspan="2">最高分數</th><th colspan="2">平均分數</th><th colspan="2">最大數字方塊</th><th colspan="2">破關率</th></tr>
<tr><td>排名</td><td>積分</td><td>排名</td><td>積分</td><td>排名</td><td>積分</td><td>排名</td><td>積分</td></tr>
<tr><td>1</td><td>10</td><td>1</td><td>10</td><td>1</td><td>10</td><td>1</td><td>10</td></tr>
<tr><td>2</td><td>9</td><td>2</td><td>9</td><td rowspan="3">2</td><td rowspan="3">5</td><td>2</td><td>9</td></tr>
<tr><td>3</td><td>8</td><td>3</td><td>8</td><td>3</td><td>8</td></tr>
<tr><td>4</td><td>7</td><td>4</td><td>7</td><td>4</td><td>7</td></tr>
<tr><td>5</td><td>6</td><td>5</td><td>6</td><td rowspan="6">3</td><td rowspan="6">2</td><td>5</td><td>6</td></tr>
<tr><td>6</td><td>5</td><td>6</td><td>5</td><td>6</td><td>5</td></tr>
<tr><td>7</td><td>4</td><td>7</td><td>4</td><td>7</td><td>4</td></tr>
<tr><td>8</td><td>3</td><td>8</td><td>3</td><td>8</td><td>3</td></tr>
<tr><td>9</td><td>2</td><td>9</td><td>2</td><td>9</td><td>2</td></tr>
<tr><td>10</td><td>1</td><td>10</td><td>1</td><td>10</td><td>1</td></tr>
</table>

## [ 保密資訊 ]

如前面所提到的, 2048.h 檔及 2048.cpp 將於繳交後被替換成正式版. 參賽者版本與計分排名用版本的差異詳列如下:

1. 輸出文字檔檔名

   最後用來決定排名用的輸出文字檔檔名將被更改, 且完全保密.

2. 亂數產生器

   Game 中提供的亂數產生器函式將在正式版中被改寫, 使得決定新方塊產生位置的隨機變數、決定要產生普通方塊或外卡方塊的隨機變數以及參賽者在任何地方可能使用的隨機變數彼此互相獨立. 另外, 評分時每一組所使用的亂數序列將完全相同.

3. 輸出與偵錯

   除最後輸出文字檔, 所有與輸出、偵錯相關之函式將被停用.

## [ 遊戲參數 ]

競賽所使用的遊戲參數列於下表:

| Parameter | Value | Description |
|---|---|---|
| INITIAL_TILE_NUM | 2 | 遊戲開始時矩陣內的方塊數字個數 |
| EMPTY | 0 | 從 Grid 中讀取到某一格子數字為 EMPTY 表示其為空格 |
| NORMAL_TILE | 2 | 普通方塊上的數字 |
| WILD_CARD_TILE | 4 | 外卡方塊上的數字 |
| TILE_GEN_RATIO | 9 | 產生普通方塊與外卡方塊的機率比 |

## [ 環境規格 ]

Platform: Ubuntu 8.04 LTS OS, Intel Xeon @ 2.5GHz * 8, 26G Ram

Compiler: gcc version 4.2.4

Libraries: Standard C/C++ Library

註: 本競賽不開放使用 Multi-thread、Multi-core Programming