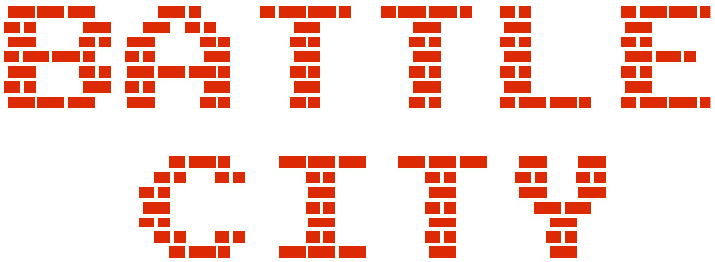
OOP 期末專題製作報告



戰車大戰

9714076 電物系大四盧立偉
9714080 電機系大四鄭祐晨

(1) 題目描述：

這戰車大戰就是指紅白機的戰車大戰，操縱戰車，將敵人通通轟掉，完成這個關卡。

小時後完紅白機時，不能輸入金手指，沒辦法自由控制遊戲，現在剛好有這個 OOP 專題機會，我們就仿製一個戰車大戰，自己可以完全控制程式碼，做到比輸入金手指更完整的控制。

(2) 系統架構和分析

一剛開始，我們思考這遊戲的雛型該怎麼定義，我們認為，需要有：戰車、子彈、地圖(map)，實體(body)。

有實體(body)的原因是因為，畫在 winBGI 的圖形就像影子一樣，那只是顯示給我們看的，電腦根本不知道子彈有沒有打到磚塊，有沒有打到戰車，所以就將 winBGI 視窗 416x416 個像素，都用陣列 body[416][416]，每個像素目前是誰佔據著，都會紀錄在這陣列裡，之後電腦要判斷這塊區域中有沒有障礙，子彈有沒有打到磚塊，哪台戰車被打到，都可以直接從陣列判斷。

雖然已經有 body[416][416]這個陣列了，另外還有 map[26][26]這陣列，因為每種地形(磚塊、海…)都是 16x16 的方塊，所以 416x416 像素，可以分成 26x26 個陣列，map[26][26]這陣列主要有兩個功用，一個是畫圖時，就讀這個陣列，把相對應的地形都印在 winBGI，另一個是可以直接利用檔案 IO，從外部編輯地圖(.txt)，所以不用再手動排地圖。

```
void setactaulbarrier(int y,int x,int type)
{
    for(int j=(16*y);j<(16*y+16);j++)
    {
        for(int k=(16*x);k<(16*x+16);k++)
        {
            body[j][k]=type;
        }
    }
}
```

有一個函式 setactaulbarrier(int y,int x,int type)，將 body[416][416]與 map[26][26]聯結起來。以 16x16 像素為一單位，將 map[26][26]上紀錄的編號(代表不同地形)，映射到 body[416][416]。

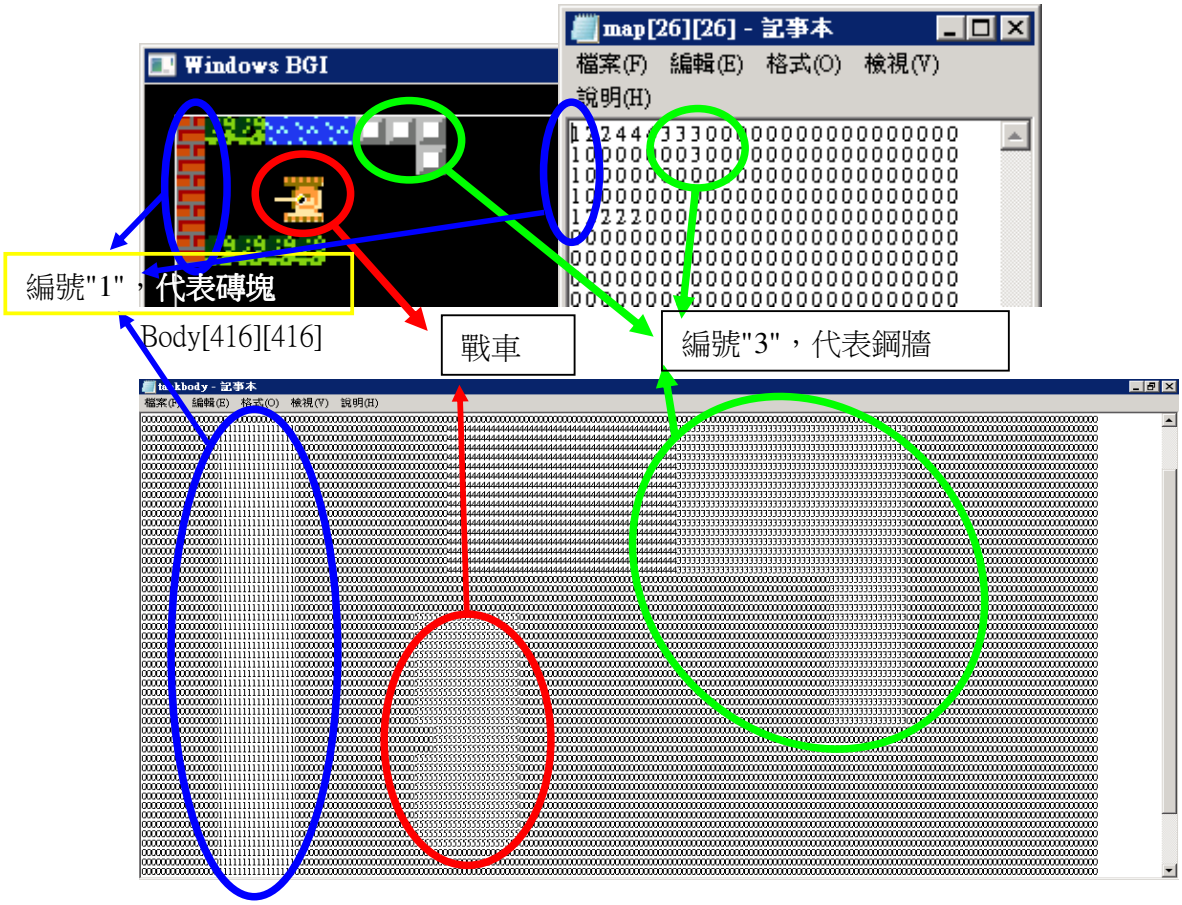
```
void tankbarrier(item tank)
{
    int y1,x1,x2,y2;
    y1=(tank.y)-29;
    x1=(tank.x)-29;
    y2=(tank.y+3);
    x2=(tank.x+3);
    if(tank.no==0)
    {
        for(int j=y1;j<=y2;j++)
        {
            for(int k=x1;k<=x2;k++)
            {
            body[k][j]=0;
            }
        }
    }
    else{

    for(int j=(y1+4);j<=(y2-4);j++)
    {
        for(int k=(x1+4);k<=(x2-4);k++)
        {
            if(k<448&&j<448)body[k][j]=tank.no;
        }
    }
...
```

另一函式 tankbarrier(item tank)，將戰車也紀錄到 body[416][416]，戰車的存在對別台戰車來說，跟其他地形一樣，也是一種障礙，有障礙的話就無法前進；這樣才不會有兩個戰車的位置重疊，而且子彈才能判斷 他打到的是哪一台戰車。

戰車編號如果是”0”的話，就代表死亡，在 body[416][416]上寫入”0”，就不會對其他戰車造成障礙。

下面有一個例子，說明陣列 map[26][26]、陣列 body[416][416]與 WinBGI 上的對應關係。



Body[416][416]上的數字代表：

1：地形,磚塊

2：地形,草棚

3：地形,鋼牆

4：地形,海洋

50~70：代表各台戰車

```
Struct item
{
      char path[16][40];
      int x;
      int y;
      int direction;
      int launch;
      int no;
      int team;
      int live;
      int effect;
      int bv;
};
```

我們定義了一個 struct item 這是給遊戲中的戰車用的，紀錄讀圖的路徑、戰車目前的座標、方向、戰車的編號、有幾條命、子彈的速度、屬於哪一隊，有什麼特殊效果(有防護罩時，或是被打到會噴道具)。

```
struct position
{
      int x;
      int y;
      int direction;
      int launch;
};
```

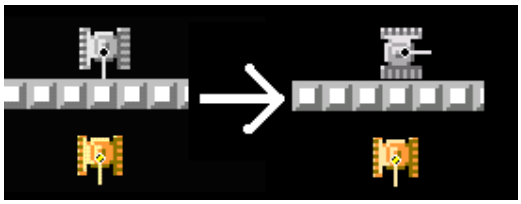
另一個 struct 是給子彈用的，紀錄子彈座標、方向、是否在發射狀態

```
While(1)
 {
        animation(tanka,bulleta,explode,star,onfield,appear,&shield,prop);
        if(tanka[0].no!=0)\\編號 0 代表死掉
          {
               itemctrl(&tanka[0],keyboard(bulleta[0]),bulleta[0],&explode[0],&star[0],onfield,&shield,prop);
               tankbarrier(tanka[0]);
          }
          for(int i=1;i<onfield;i++)
          {
               if(tanka[i].no!=0) )\\編號 0 代表死掉
               {
                    itemctrl(&tanka[i],(*comfunc[rand()%13])(&tanka[i],8+rand()%8,tanka[0]),bulleta[i],&explode[i],&star[i],onfield,&shield,prop);
               }
               tankbarrier(tanka[i]);
          }
}
```

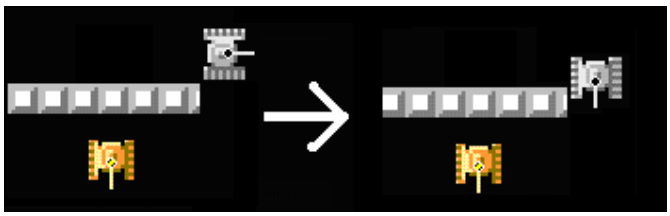
上面是程式主要部分，有兩個主要的函式，一個叫作 animation()，是專工畫圖的函式，這函式的功用就是控制圖案的輸出。

另一個叫作 itemctrl()，整個遊戲的規則都是由這控制的，專門控制圖案的位置，子彈的位置、速度，戰車的位置、速度，磚塊是否被打到，被打到後會怎樣，都是這個函數控制的。

最後一個重要的部分是敵方如何追蹤玩家戰車，我們把每個階段想到的方法都留下來，總共有 13 個，編號越後面，考慮的條件會越多，也會聰明一些。
追蹤的方法是這樣的：
假設往上下左右四個方向各走一步，看哪個方向離目標最近，就選哪個方向。
如果算出來是 下 最近，但是前方剛好有磚塊過不去，該怎麼辦呢?是磚塊的話，就直接打穿前面的磚塊往前走，如果是鋼牆的話，這招就行不通了，程式會紀錄戰車上一次的位置，如果上次位置與這次位置一樣，表示他無法打穿這個障礙物，所以就山不轉路轉，決定繞路!



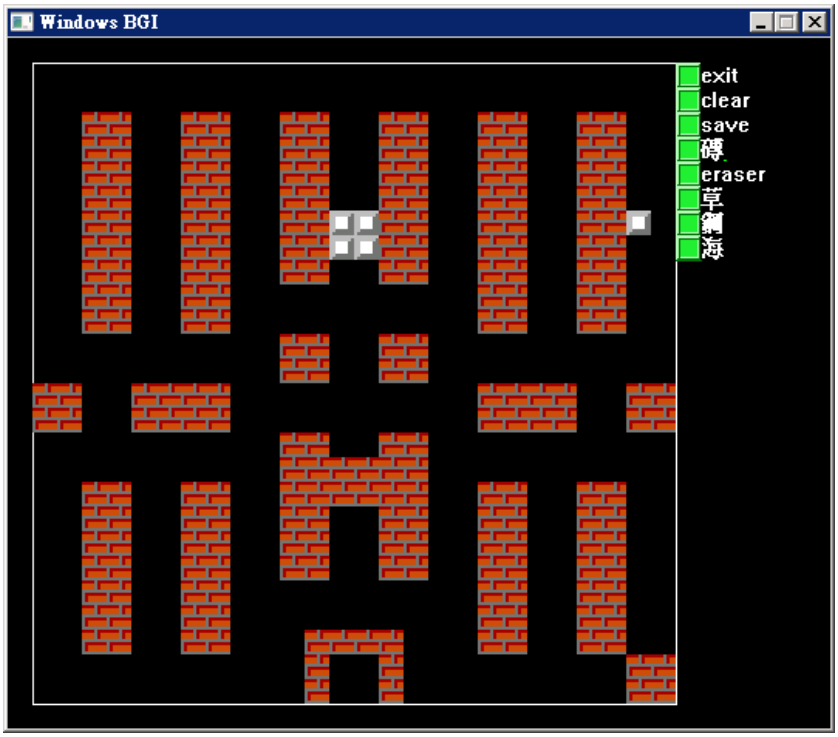
程式會紀錄 上下左右 哪個方向是不能走的，像上面那情況，他的”下”是不能走的，所以他就會選擇往左或右，一直走，直到偵測到原本不能走的”下” ，現在可以走了，這樣就繞過障礙物了



(3)執行結果

程式一開始，有說明控制方式，選單有兩個選項，1 是進入遊戲，2 是編輯地圖
控制方式:方向鍵
Z:發射普通子彈
X:發射後，還能控制方向的子彈
C:在戰車周圍旋轉的子彈
V:會自動追蹤敵人的子彈

進入編輯地圖
點擊右邊的選項，可以用滑鼠編輯，利用檔案 IO 使兩個程式能編輯同一張地圖



進入遊戲

把敵方戰車全殺光就算贏；生命值耗盡

這些是全部的道具(期末 demo 後才加上去的)

 子彈速度+1，而且可以打穿鋼牆

 老鷹周圍出現鋼牆圍著

 生命值+5

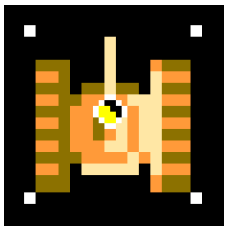
 戰車周圍出現防護罩

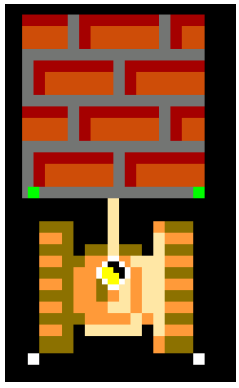
(4)心得報告 (遇到的問題，如何解決或結果分析)

遇到的問題：

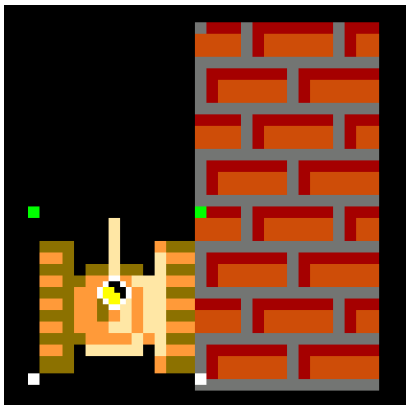
第一個遇到的問題是，戰車常常被磚塊卡住，發現原因是這樣的：

因為戰車是由四個頂點的座標偵測是否前方有障礙物，



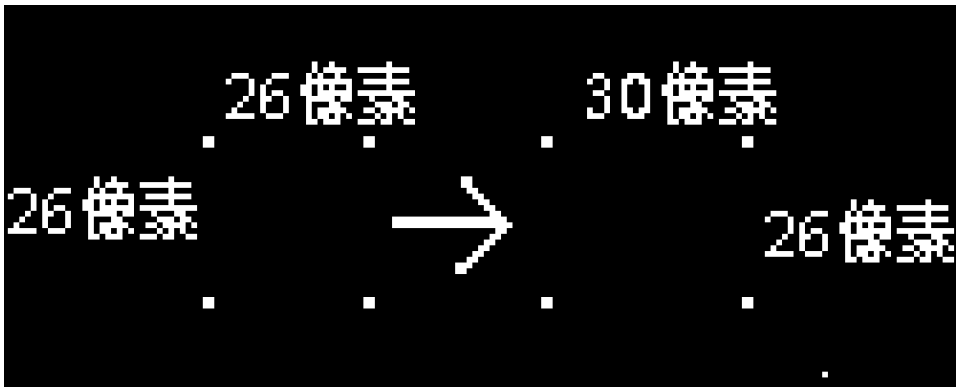


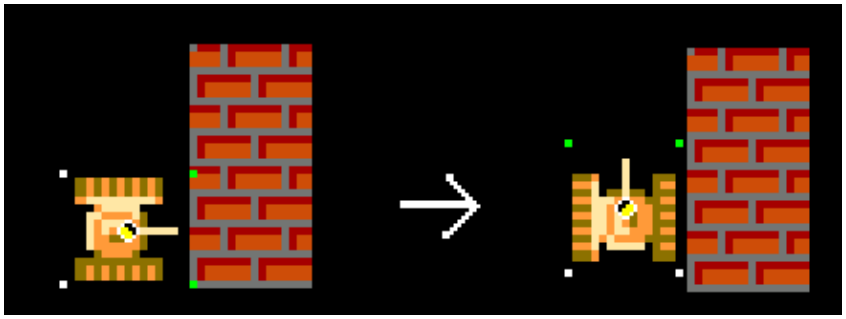
像左圖這種情況，現在戰車方向是"上"，所以上方兩個頂點(綠色)，會在陣列 body[416][416]偵測到前方已經有磚塊了，所以不能在往前了。



不過像這種情況，戰車想往上走，明明前方沒障礙物，但是右上方那點已經偵測到磚塊，所以戰車就會無法前進，感覺像是被磚塊黏住。

知道了原因，我們的解決方法就是，把四個頂點的距離改成不再都是 26 像素





這樣一來，因為偵測頂點的距離從 26 像素加大到 30 像素，當戰車往右走時，右邊有障礙物，他會較早就被障礙物擋住，不能太接近，此時，方向轉上的話，右上方的點就不會偵測到障礙物，就能順利地前進了。缺點是要犧牲左右方向的自由度，不能過分接近障礙物。

另外一個較大的問題是，有時會存取超出範圍的陣列，導致程式當掉。因為戰車每移動一次，就會在陣列 body[416][416]，在相對應的座標寫入戰車的編號，紀錄位置，戰車的速度是 3 像素，假如目前戰車的座標是(414,414)，現在他要往右走，照理說座標要變為(417,414)了，但是 body[417][414]已經超出陣列定義的範圍了，所以程式就當掉了。

還有另一個類似的問題，子彈打到邊界的磚塊，因為沒考慮周全，運算出來是 map[26][10]這個磚塊必須被消除，其實 index 26 已經超過原本定義的大小了(char map[26][26]，index:0~25)，但這時後程式反而不會當掉，但是電腦會去存取 map[0][10]的資料，所以就造成了：明明打到的是右邊，卻是左邊的磚塊被消除掉，難道 map[26][10] == map[0][10]?，這我們也摸不清。

為了解決上述超出邊界的問題，就是把邊界變成不是邊界，陣列 body[416][416]加大成 body[448][448]，map[26][26]變成 map[28][28]，然後我們使用 body[16~431][16~431]與 map[1~26][1~26]當作戰車可以出現的範圍，所以就算偶而超出邊界，也不會超出定義的陣列大小，造成當機。

心得：

寫專題感覺很有趣，感覺就像是玩模擬市民、模擬城市一樣，可以自己動手創造，做出任何自己想要的效果，雖然只是個小小的程式，別人眼中或許不怎樣，但自己玩起來還是頗有成就感。

感覺我們太晚教到 class 了，對 class 的使用也不太熟悉，不然使用 class 的話程式應該可以寫得更簡潔明瞭。

有時遇到問題時，感覺跟數學很像，一直想就是想不通，怎樣 debug 也找不出錯，今天放棄後，明天換個心情重新看一次，竟然就看出問題在哪了

我們專題題目太晚定下來了，改了兩次最後才決定做坦克大戰，如果早點決定，時間就可以充裕些了。

(5)參考資料

上課講義

http://www.cs.colorado.edu/~main/bgi/doc/ (查詢 graphic.h 的用法)

http://msdn.microsoft.com/en-us/library/windows/desktop/dd743680(v=vs.85).aspx (查詢 playsound()的用法)

http://caterpillar.onlyfun.net/Gossip/CGossip/CGossip.html

(6)程式列表

使用 dev C

完成的程式：battlecity.exe、tankmapeditor.exe

程式碼 在壓縮檔裡 battlecity.cpp、tankmapeditor.cpp