康威生命遊戲

組別：第四組

組員：B075020033謝朋潔、B083040020童冉琳、B083040025蔡苡暄、

B083040031邱歆惠、B083040045李亦晴

**1.摘要**

對於現在的21世紀的我們，人工生命似乎還距離我們很遙遠，我們電腦的計算量遠遠不足以去計算以模仿大自然的每個變化，但若就生命繁衍的規則來說，其實在1970年《科學美國人》雜誌上就出現了，是由英國數學家約翰·何頓·康威提出的生命模型。康威生命遊戲是一個風靡學術圈的遊戲，康威因此成為家喻戶曉的人物，它在70年代爆紅，根據當時統計，全世界有四分之一的電腦都在進行這個遊戲，更因為許多上班族在上班時玩這個遊戲而造成巨大的經濟損失，雖然名為遊戲，但它其實沒有任何玩家，只要我們設定好條件，就可以看到生命自行繁衍，而我們這次用C語言寫出了簡單的康威生命遊戲，並用亂數決定初始細胞位子，讓每一次生命的初始位子都不同，以更好觀察康威生命的繁衍。

**2.簡介**

康威生命遊戲是一個在二維世界的生命繁殖遊戲，以一個方格為單位，當空間過度擁擠或寬廣時中間的細胞都會死亡，以下是這個遊戲裡生命繁衍的幾個條件：

(1)細胞有存活跟死亡兩種狀態，會受到周圍八個格子裡生命的狀態影響

(2)當中間細胞存活，周圍若有三個以上細胞存活，中央細胞就會死亡

(3)當中間細胞存活，周圍有兩個或三個存活，中間細胞會維持存活

(4)當中間細胞存活，周圍細胞小於兩個存活，中間細胞則死亡

(5)當中間細胞是死亡狀態，周圍有三個細胞存活，中間細胞會存活，其他裝況則維持原樣。

以上五個條件可以跟現實生命繁衍法則做比對，第(2)條是指過度繁衍，資源不足，第(3)條指生態過於貧瘠，第(5)條則是生命的繁衍。而這些細胞在繁衍過程中會呈現幾種不同的模式，包括：

(1)靜物：如果沒有其他細胞干擾會永遠保持不變的圖形

(2)振盪器：經過一定的週期會回到初始圖形並不停輪迴

(3)太空船：若路徑上沒有其他阻礙，會週而復始向對角線繁衍在模型這部分之後我們會再做更深入的說明。

康威生命的巧妙之處就在於，按照這五條規則，細胞可以一代接著一代繁衍，周而復始，不需要人為中途操控，而這也牽涉到宗教及哲學上的意義：在沒有人為干擾下，生命也能自行的發展出組織。

而我們這次寫的程式中，只運用C語言中最基礎的for和while迴圈以及if-else來做運算，也讓我們發現看似複雜的規則其實邏輯並不困難。

**3.相關研究**

細胞自動機，可以說是人工生命被提出前的一個理論基礎，是由斯塔尼斯拉夫·馬爾欽·烏拉姆(Stanislaw M. Ulam)在1950年代提出的概念。

其主要是將一格一格的格子模擬為自然界的細胞。而每個細胞在每個時刻會存在一種狀態，此狀態是會隨著時間、環境，依據一定的規則改變。所謂環境，即該細胞周圍其他細胞的狀態。而規則，則是模擬自然界的生物依據環境改變可能對應的狀態變化。

整體來說，只要賦予每個細胞一個初始的生存狀態，細胞自動機即能隨其規則演變，直至達成可能的靜態或動態的平衡。

另外，有關細胞自動機的活動空間，可以是一至多維的狀態。其中，本文重點講述的Conway’s game of life即是其中的二維的細胞自動機。

Conway’s game of life，是在細胞自動機的概念被提出數年後，由

數學家 約翰·何頓·康威(John Horton Conway)提出的構想。Conway’s game of life在被提出後不久，即引發多方的注意、許多人加入嘗試，其也使當年較無人重視細胞自動機的概念重新被關注。

John Horton Conway將每個細胞的狀態設定為「生」或「死」，演變規則設定在細胞群落的消長，影響個體的狀態。包含中間細胞及其周圍圍繞的八格視為一個群落，由此八個格子決定中間細胞的生或死。當一個群落細胞太多，模擬的是自然界中，生物數量過多，超過該環境的負載力，而造成的死亡及淘汰。當一個群落只餘一個細胞生存，則細胞會因孤單及缺乏同伴而無法生存，狀態會改為死亡。當環境中的存活數，恰符合環境的最適生存數，則中間細胞維持原本的生存狀態。若中間細胞處於非存活狀態，但群落卻有足夠繁衍生命的個體數，則中間細胞會因此被孕育出生命，細胞的狀態改為生，此乃自然界中的長。

以下是簡單整理出的Conway's game of life的規則:

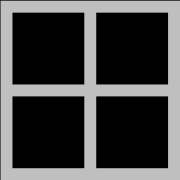
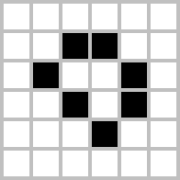
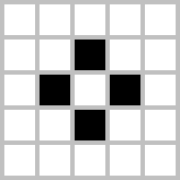
1.中心細胞存活時，偵測出環境存活細胞數低於兩個，不符合生命群聚生長特性，中心細胞死亡

2.中心細胞存活時，偵測出環境存活細胞數高於3個，高於環境負荷度、環境過度擁擠，中間細胞生存狀態改為死亡

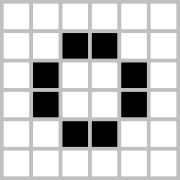
3.中心細胞存活時，偵測出環境存活細胞數為2或3個，剛好形成最適環境，中間細胞維持原先的生存狀態

4.中心細胞死亡時，偵測周圍環境存活數為3個，則中間細胞改為存活狀態，狀似自然界中的繁衍

在長時間演化過後，每個格子相互達成靜態或動態平衡，可能成為以下各式圖樣:

板凳 麵包 花朵

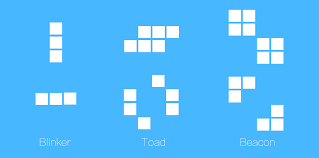


鴨子 航空母艦 池塘

以上是穩定態、靜態平衡多種圖樣的其中六種。穩定態多是由上述的第三種規則構成，相鄰的群落互相提供二或三個存活個數，造成群落中心細胞保持存活狀態。

 信號燈

而震盪狀態，則有信號燈、蟾蜍等等。震盪狀態則是具有一定的週期性，依據上述各種規則搭配形成週期性的圖樣。舉例來說，信號燈就是由第一、三及第四項規則構成。設在空間中，圖形在某一瞬間狀態如上圖，中央橫向的三個連續的細胞處於存活狀態，其餘周圍皆是死亡狀態。依據第一項規則，周圍環境存活數低於二時，則中心細胞死亡，我們可以預測在下一瞬間左邊及右邊細胞會因為規則一(環境生存數低於二)消亡，中央會因為規則三(環境生存數恰為二)保留，中上及中下則會因為規則四(環境生存數為三，中心細胞由死轉生)而繁衍出新的生命。故而下一瞬間，圖形轉為三個生存的細胞，縱向的排成一排。再下一瞬間，又因上述規則，會再度回到橫向三個的生存狀態，形成週期的動態平衡。

 蟾蜍

上圖是另一震盪狀態的經典例子--蟾蜍。其也是呈現二輪週期的震盪，且是由上述的第二、三及四項規則構成。分析上圖，分別偵測上排左一、上排中間、下排右一及中間的周圍八格，發現生存數為4，依據規則二(環境生存數高於3，中心細胞死亡)，可以預測出下一瞬間這些格子的狀態為死亡。而分別以上排右一及下排左一為中心偵測，發現環境生存數皆是2，符合規則三，故而會保持原本的存活狀態。接下來持續觀測，發現原本的存活狀態能使周圍環境生存數恰為三的情況，符合規則四(中心細胞死亡，但環境生存數為三)而孕育出生命。故能推測下一瞬間的狀態為下圖。

接下來分析下圖狀態。有四個格子會因第一項規則，環境中的存活數低於2而滅亡；但同時，會有四隔因周遭環境生存數恰有三個，符合第四項規則而由死轉生。而其中有二格會在環境中的消長下恆符合第三項規則(周圍生存數為2)而一直處於生存狀態。上下兩張圖在不停地迭代中會週期的輪替出現，形成震盪。

另外還有會移動的震盪狀態，舉例來說有滑翔機及太空船。兩者皆是四輪一周期，並穩定的朝向某處移動，直至因觸碰到邊界而改變環境狀態，才會停止。

**4.程式設計方式&討論**

根據Conway’s game of life的規則，我們可以設計出程式模擬出其消長。在20\*20大小的區塊內，簡單模擬一下Conway’s game of life。

以下程式由C語言撰寫，僅節錄部分程式並簡單說明:

int i,j,k,l,keep;

int num[20][20],save[20][20];

char arr[20][20];

for(i=0;i<20;i++)

{

for(j=0;j<20;j++)

{

num[i][j]=0;

save[i][j]=0;

}

}

首先設定變數，重點的部分是陣列的宣告。共有三個陣列，其中兩個是整數型態的二維陣列，儲存值為1或0。1表該位置細胞為存活，0表該位置細胞為死亡。陣列num是存放原情況下的生存狀態，而陣列save是暫時存放下一瞬間細胞改變後的狀態。字元陣列arr則是儲存根據num狀態決定的空格(死亡)或o(生存)。

這邊比較值得一提的是，我們在開始時，將所有num和save中的值都暫設為0，這部分在後面會說明為何有這樣的步驟，以及為何有必要宣告兩個陣列。

srand(time(NULL));

for(i=1;i<19;i++)

{

for(j=1;j<19;j++)

num[i][j]=rand()%2;

}

for(i=0;i<20;i++)

{

for(j=0;j<20;j++)

{

if(num[i][j]==1)

arr[i][j]='o';

else

arr[i][j]=' ';

}

}

for(i=0;i<20;i++)

{

for(j=0;j<20;j++)

printf("%c",arr[i][j]);

printf("\n");

}

隨機設定初始生存狀態並儲存入陣列num中。我們引入函數srand(time(NULL))確保每次的取出亂數皆不同(引入標頭檔<time.h>)，random函數取餘數2，使取出的亂數結果為1(生存)或0(死亡)。再將陣列num中依據亂數出的生存狀態，對應字元陣列arr的位置，若生存(num[i][j]==1)則在該位置儲存‘o’表示該細胞存活，若死亡(num[i][j]==0)則在該位置儲存‘ ’表示該細胞死亡。最後，在畫面中將字元陣列中的儲存值全顯示出來，這邊要注意的是，我們只將空間中間的格子存入隨機狀態，而空間的外圍維持原先初始的0狀態。其主要原因是我們要偵測周圍八個格子的狀態，來決定中心細胞的存活狀態時，若取周圍格子作為中心細胞，則偵測計算時會取到超出在二維陣列中的位置。譬如說，我們取外圍的num[0][0]作為中心細胞，則偵測周圍時，會取到num[-1][0]這樣的值，然而此位址不合理、不存在。故而我們將周圍一圈預設為0(死亡狀態)，視之為「牆」。

for(k=1;k<19;k++)

{

for(l=1;l<19;l++){

keep=num[k-1][l-1]+num[k-1][l]+num[k-1][l+1]+num[k][l-1]+num[k][l+1]+num[k+1][l-1]+num[k+1][l]+num[k+1][l+1];

if(num[k][l]==1)

{

if(keep<2)

save[k][l]=0;

else if(keep==2 || keep==3)

save[k][l]=1;

else if(keep>3)

save[k][l]=0;

}

else if(num[k][l]==0){

if(keep==3)

save[k][l]=1;

else

save[k][l]=0;

}

}

}

結束初始設定，接下來開始進入重頭戲--判斷，決定下一瞬間每個格子的狀態。

進入while迴圈，並將條件設為恆真，使程式可以無限的執行、判斷。進入迴圈後，取合理的中心格子(介於陣列第一至十八間，含一及十八)，將其周圍八個位置生存狀態相加，存入變數keep，以便判斷群落生存數。若原本中心細胞生存(num[i][j]==1)，且若群落生存數小於2(keep<2)，則在save中存入下一瞬間中心細胞的死亡狀態(save[i][j]=0)。若群落生存數小等於2或3(keep==2||keep==3)，則在save中存入下一瞬間中心細胞的存活狀態(save[i][j]=1)。若中心細胞生存，然而不屬於上述任一狀態，進入else，也就是群落生存數大於2(keep>3)，環境過度擁擠，中心細胞死亡，則在save中存入下一瞬間的死亡狀態(save[i][j]=0)。若原本中心細胞死亡(num[i][j]==0)，且若群落生存數洽為3(keep==3)，群落產生繁衍，中心細胞由死轉生，則在save中存入下一瞬間為「生」的生存狀態(save[i][j]=1)。

另外，此區塊可以看出宣告兩個陣列的必要性。因為電腦是一個一個格子去判斷、決定是否改動的。若只使用一個陣列，上面被改動過的數值，會影響下面的判斷。如此一來，上下的格子就相當於不是在同一時刻下的狀態演變，存在時間差。故而需要另一個陣列save去儲存被改動後的狀態，而原本的狀態num則保留下去繼續判斷。

for(i=0;i<20;i++)

{

for(j=0;j<20;j++)

num[i][j]=save[i][j]; }

for(i=0;i<20;i++)

{

for(j=0;j<20;j++)

{

if(num[i][j]==1)

arr[i][j]='o';

else

arr[i][j]=' ';

}

}

system("cls");

for(i=0;i<20;i++){

for(j=0;j<20;j++){

printf("%c",arr[i][j]);

}

printf("\n");

}

Sleep(20);

接下來則是將上個區塊的判斷結果稍做處理，並顯示出來。

首先先將暫存在陣列save中的下一瞬間結果改入陣列num中。並根據num中的生存狀態改變字元陣列arr中存放的顯示生存(‘o’)或死亡(‘ ’)。

引入函式system("cls")，將上衣瞬間的結果清空(該函式被包含在標頭檔stdlib.h中)。最後將字元陣列中的arr生存狀態顯示在畫面中。

最後再引入函式Sleep()(此函式被包含在標頭檔windows.h中)，將此瞬間的圖片稍作停頓，括號中的時間單位在windows中是毫秒(ms)，我們選定20毫秒，故而每幀畫面會定格0.02秒。

**5.結論**

無論由細胞自動機至康威的生命遊戲，這些理論皆提供我們新的方式去看待人工模擬大自然演變的大智慧。透過一定的規則，我們可以將之寫成程式，使程式依規則而行，我們可以知道其中消長變幻、探討其原因、一窺其中的奧秘。

康威的生命遊戲，最大的特徵是能透過一些簡單的規律，創造出龐雜的結果型態。它的用途並不僅止於純粹的觀賞性質、可精密設計的藝術創作，它更是一個可以模擬變動的模型，模擬大自然生物的變遷、粒子的變動，甚至於模擬一些我們社會方面的領域，提供各類學者一個新的途徑去預測及模擬更多結構豐富、龐大的可能性。故而人們能多加探索、開發其用途。

**6.參考資料：**

<https://zh.wikipedia.org/wiki/%E5%BA%B7%E5%A8%81%E7%94%9F%E5%91%BD%E6%B8%B8%E6%88%8F>

<https://pansci.asia/archives/95148>

<https://huanqiukexue.com/a/qianyan/xinxi__nenyuan/2017/0106/26910.html>

<http://www.atlas-zone.com/complex/alife/ca/index.html>