



2018

程式設計
沒有加強班

程式設計與實習(二)

BY 孫茂勛
筆電修好了YA

Email:JOHN85051232@GMAIL.COM

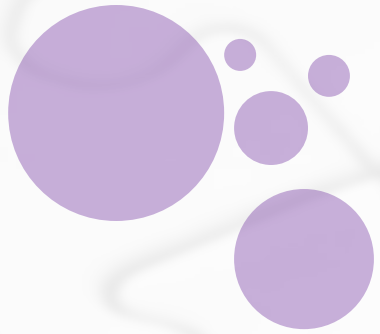


記得喔！！

🔵 5/8(二)大一同學(學號A10655__)記得要來進行APCS試測，非大一同學不用參加~



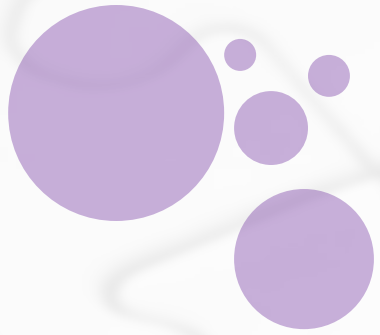
記得喔



Backtracking

Backtracking(回溯法、窮舉法、暴力法...)

- 在搜尋一個問題的解答的時候，如果沒有任何策略，最直接的方式就是把所有可能的答案都判斷一遍。
- 在程式中常使用遞迴來窮舉所有可能答案的解，直到找到正確答案(終止條件)。



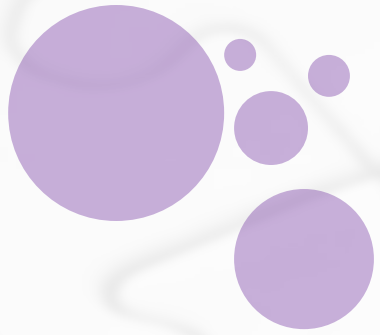
Backtracking

窮舉所有可能答案的解?

🔲那你就得先理解問題

Q1:請列出所有由{1,2,3}三個數字組合而成的三位數(數字可重複)?

🔲答案總共有27種，why?



Backtracking

如何找出所有可能的解？

1 1 1

1 1 2

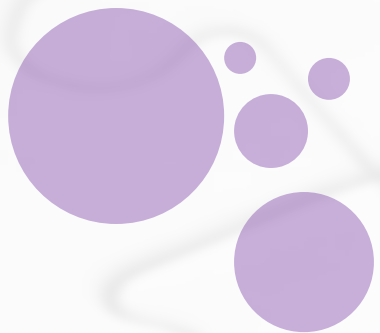
1 1 3

1 2 1

1 2 2

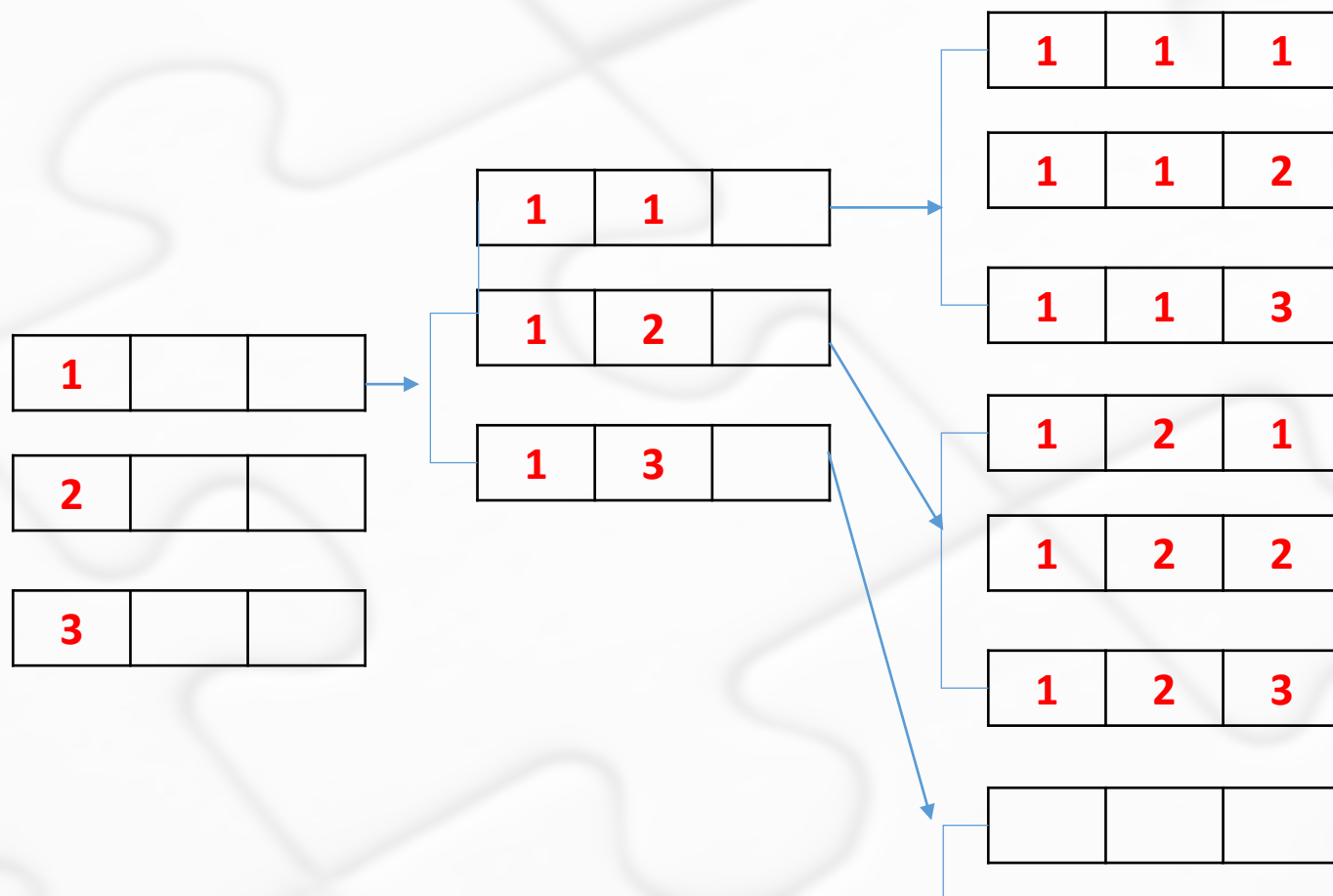
1 2 3

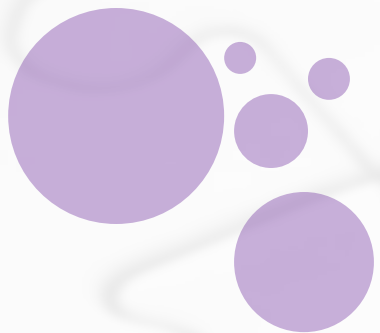
...



Backtracking

如何找出所有可能的解？





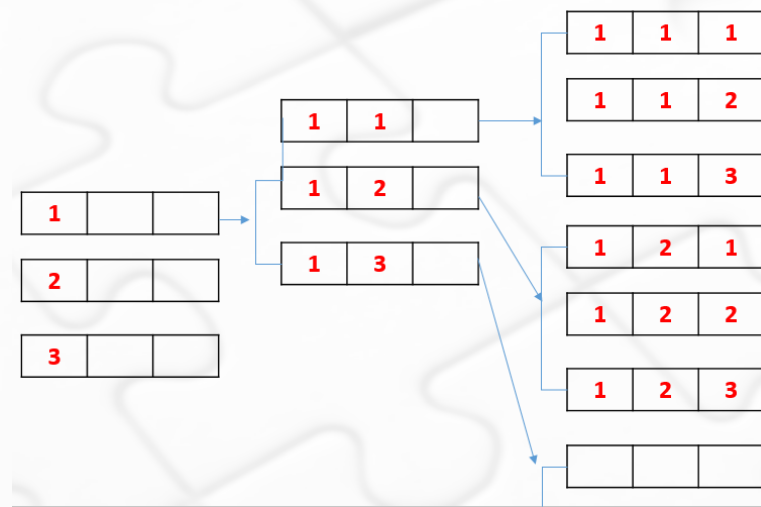
Backtracking

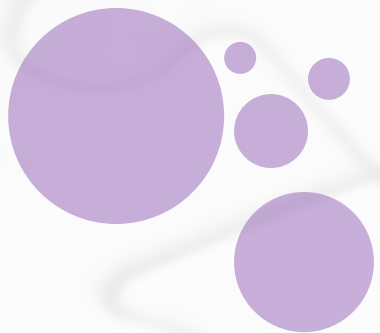
Algorithm:

Let array[4] = {0}, i = 0

Void Backtracking(i):

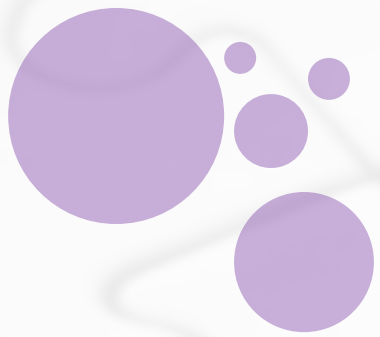
1. if i == 4 then return; //3個位置都找完了
2. array[i] = 1
3. backtracking(i+1) //遞迴i+1個位置找解答
4. array[i] = 2
5. backtracking(i+1)
6. array[i] = 3
7. backtracking(i+1)





Backtracking

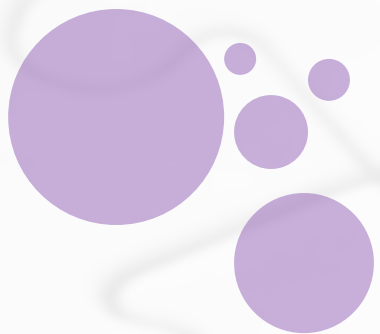
```
int ans[4] = {0}; // 不用第0個位置
void print_ans() //把解答印出來
{
    for (int i = 1; i < 4; i++)printf("%d ", ans[i]);
    printf("\n");
}
```

Backtracking

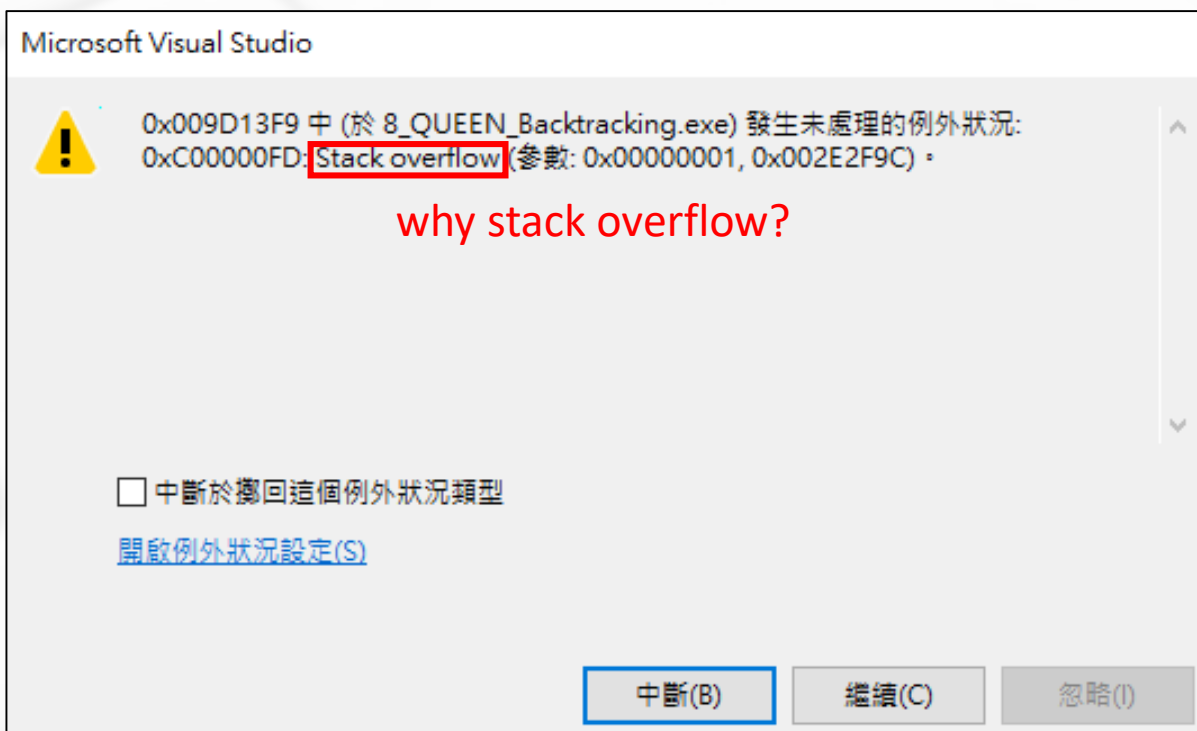
🧩 給第*i*個位置不同的值，並遞迴*i*+1個位置找解答

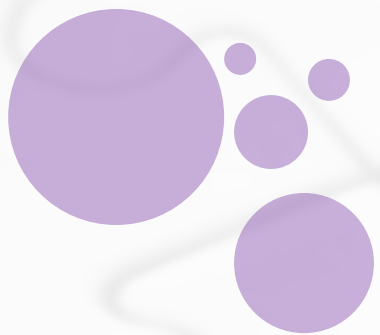
```
void backtracking(int i) //i代表現在在遞迴(枚舉)答案的第幾個位置
{
    if (i == 4) //枚舉完三個位置的答案後就結束遞迴
    {
        print_ans();
        return;
    }
    ans[i] = 1;
    backtracking(i + 1);
    ans[i] = 2;
    backtracking(i + 1);
    ans[i] = 3;
    backtracking(i + 1);
}
```



複習一下

- 上學期教遞迴時強調過的，遞迴的終止條件很重要。
- 一個弄不好你就會收到這個

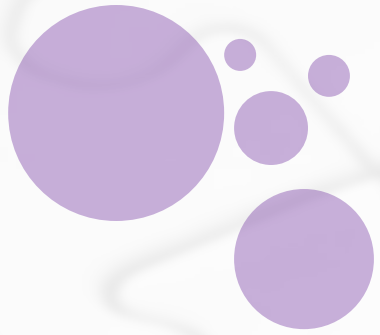




Backtracking

寫成迴圈

```
void backtracking(int i) //i代表現在在遞迴(枚舉)答案的第幾個位置
{
    if (i == 4) //枚舉完三個位置的答案後就結束遞迴
    {
        print_ans();
        return;
    }
    for (int j = 1; j < 4; ++j)
    {
        ans[i] = j;
        backtracking(i + 1);
    }
}
```

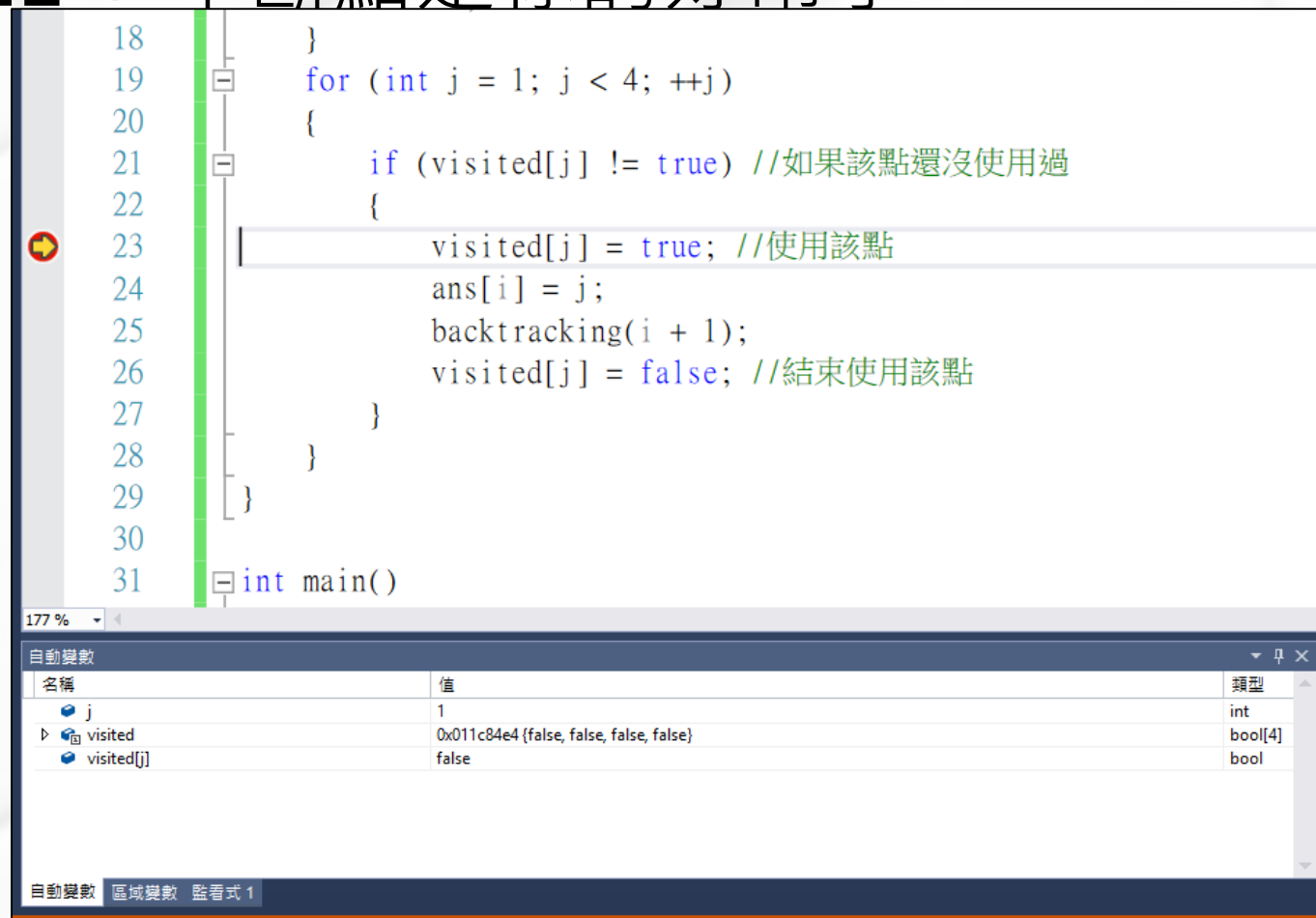


Backtracking

```
int main()
{
    backtracking(1); //從第一個位置開始枚舉
    system("pause");
    return 0;
}
```

遞迴看不懂現在執行到哪一步了？

🔧 F10 + F11 + 中斷點是你的好幫手

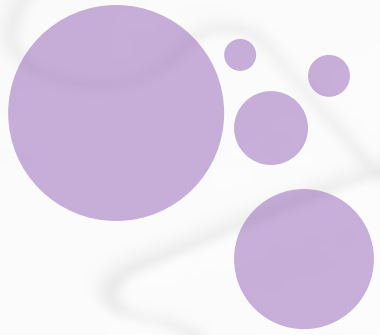


The screenshot shows a C++ IDE with a code editor and a variable watch window. The code is a recursive function `backtracking` that explores a path. A red arrow points to line 23, where `visited[j] = true;` is executed. The variable watch window at the bottom shows the current state of variables: `j` is 1, `visited` is an array of four false values, and `visited[j]` is false.

```
18     }
19     for (int j = 1; j < 4; ++j)
20     {
21         if (visited[j] != true) //如果該點還沒使用過
22         {
23             visited[j] = true; //使用該點
24             ans[i] = j;
25             backtracking(i + 1);
26             visited[j] = false; //結束使用該點
27         }
28     }
29 }
30
31 int main()
```

名稱	值	類型
j	1	int
visited	0x011c84e4 {false, false, false, false}	bool[4]
visited[j]	false	bool

自動變數 區域變數 監看式 1



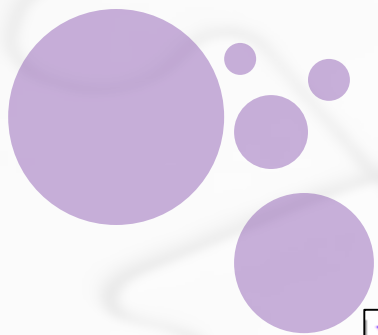
Backtracking

Q2:請列出所有由{1,2,3}三個數字組合而成的三位數(數字不重複)?

💡答案總共有6種，why?

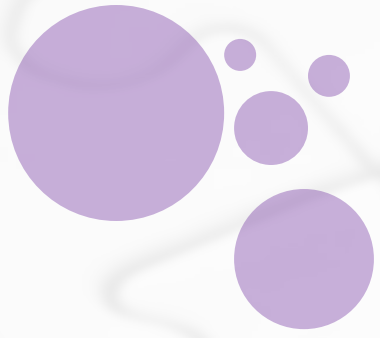
我們需要多一個資料結構來記錄那些數字我們使用過了！

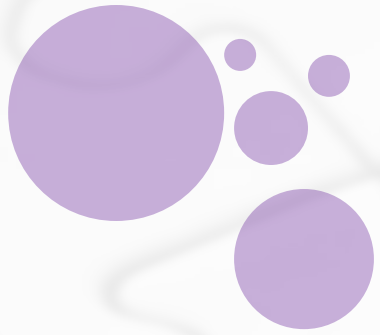
```
bool visited[4]; // 全域的bool初始值都是-1
```



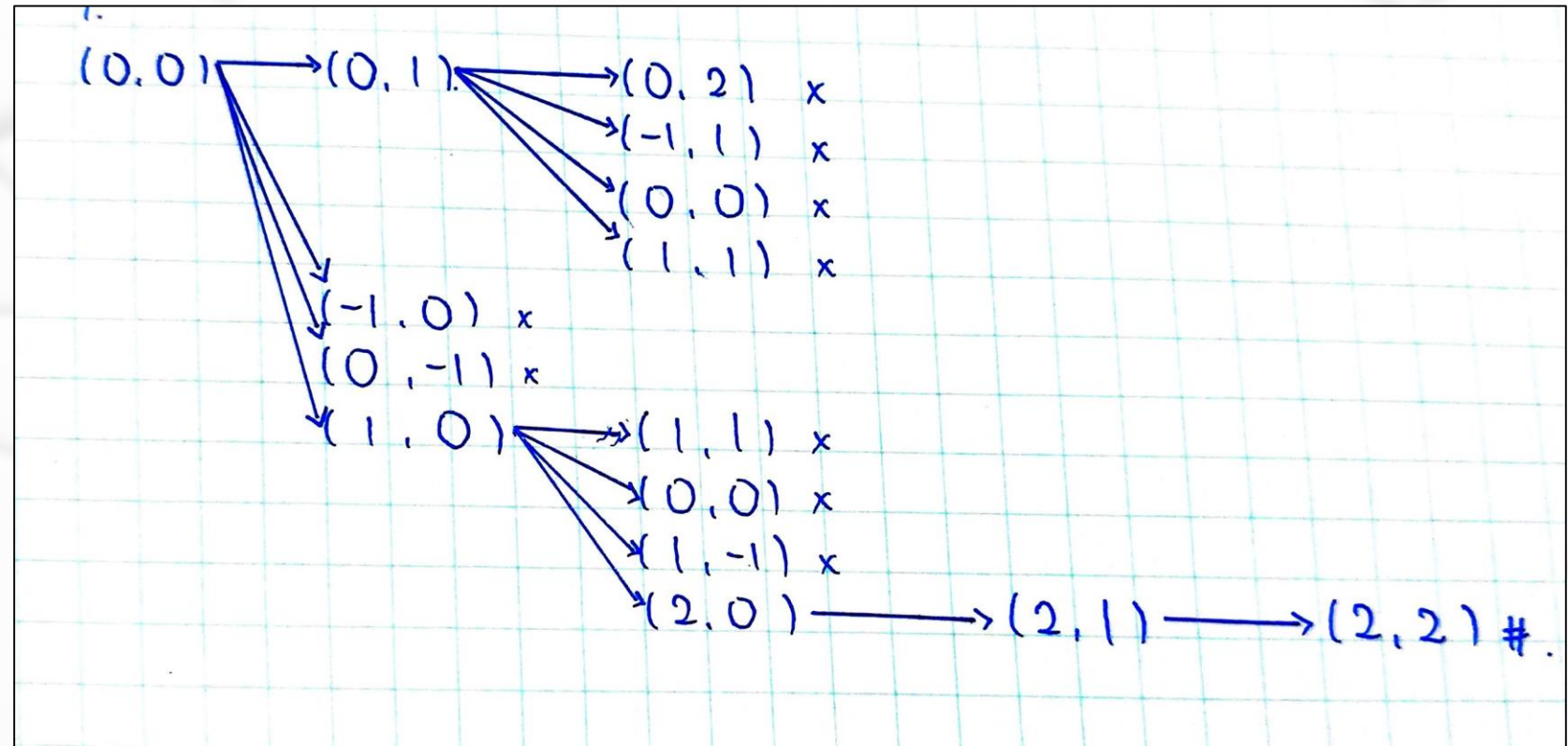
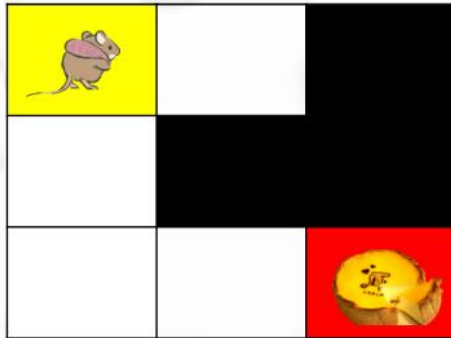
Backtracking

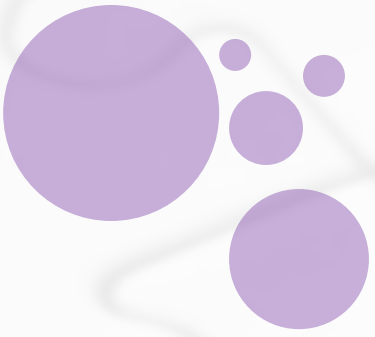
```
void backtracking(int i) //i代表現在在遞迴(枚舉)答案的第幾個位置
{
    if (i == 4) //枚舉完三個位置的答案後就結束遞迴
    {
        print_ans();
        return;
    }
    for (int j = 1; j < 4; ++j)
    {
        if (visited[j] != true) //如果該點還沒使用過
        {
            visited[j] = true; //使用該點
            ans[i] = j;
            backtracking(i + 1);
            visited[j] = false; //結束使用該點
        }
    }
}
```





老鼠走迷宮

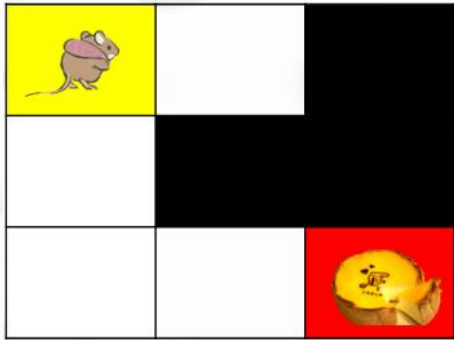




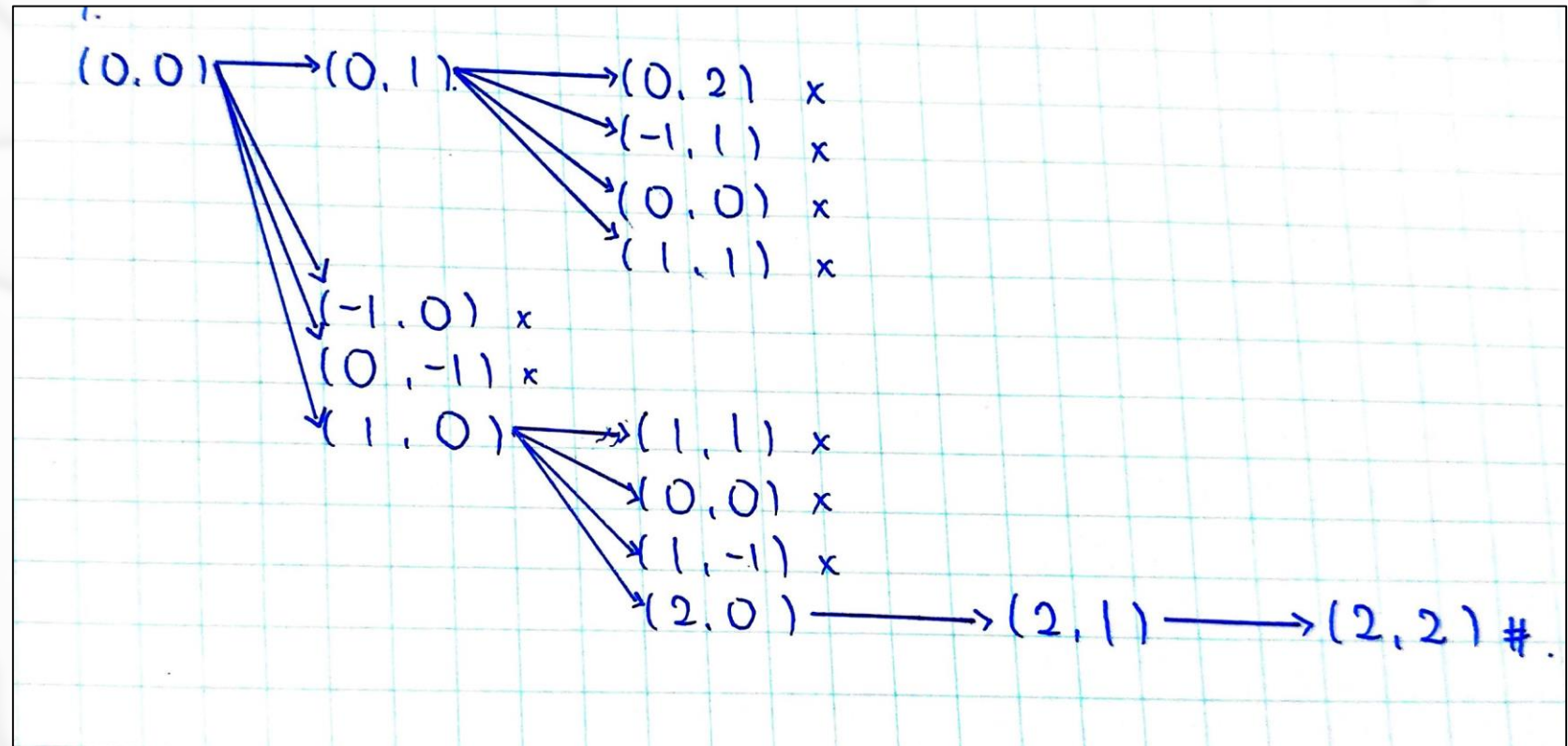
用stack的觀點來看老鼠走迷宮

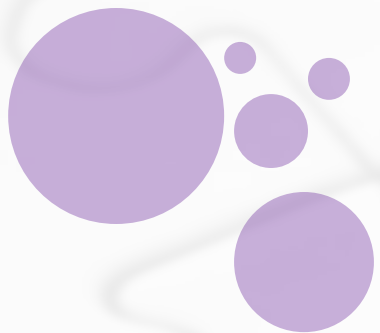
Q：下面stack狀態中，假設(0,2) 已經沒有路了怎麼辦？

A：pop掉top，在push(0,1)向上的stack



(0,2)
(0,1)
(0,0)





走迷宮搜

- 🔲 設計 2 ~ 3 個你自己的地圖吧
- 🔲 (第一列的兩個數字是行跟列大小)
- 🔲 (數字用空格隔開)
- 🔲 設計完先放在筆記本裡

```
未命名 - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

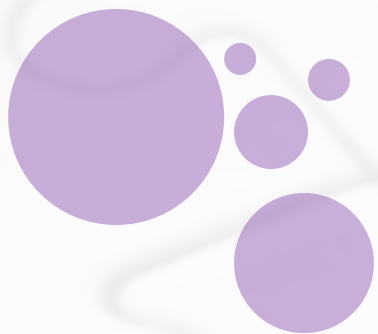
10 10
0 0 1 1 1 1 1 1 1 1
1 0 1 0 1 1 1 0 0 1
1 0 0 0 0 0 1 0 1 1
1 1 1 1 1 0 0 0 1 1
1 1 0 0 0 0 1 1 1 1
1 0 0 1 0 1 1 1 0 0
1 1 1 1 0 0 0 0 0 1
1 1 1 1 1 1 0 1 1 1
1 1 1 1 1 1 0 0 1 1
1 1 1 1 1 1 0 0 1 1
1 1 1 1 1 1 1 0 0 0

4 4
0 0 1 1
1 0 0 1
1 1 0 0
1 1 1 0

3 3
0 0 1
0 1 1
0 0 0
```

走迷宮搜

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 100
int col,row;
int map[SIZE][SIZE] = {0};
bool is_end = false;
//2:走過的路線 1:牆壁 0:走道
int main()
{
    scanf("%d %d",&col,&row);
    for(int i = 0 ; i < col ; ++i)
    {
        for(int j = 0 ; j < row ; ++j)
        {
            scanf("%d",&map[i][j]);
        }
    }
    int start_x = 0,start_y = 0;
    int end_x = col-1,end_y = row-1;
    visit(start_x,start_y,end_x,end_y);
    system("PAUSE");
}
```



走迷宮搜

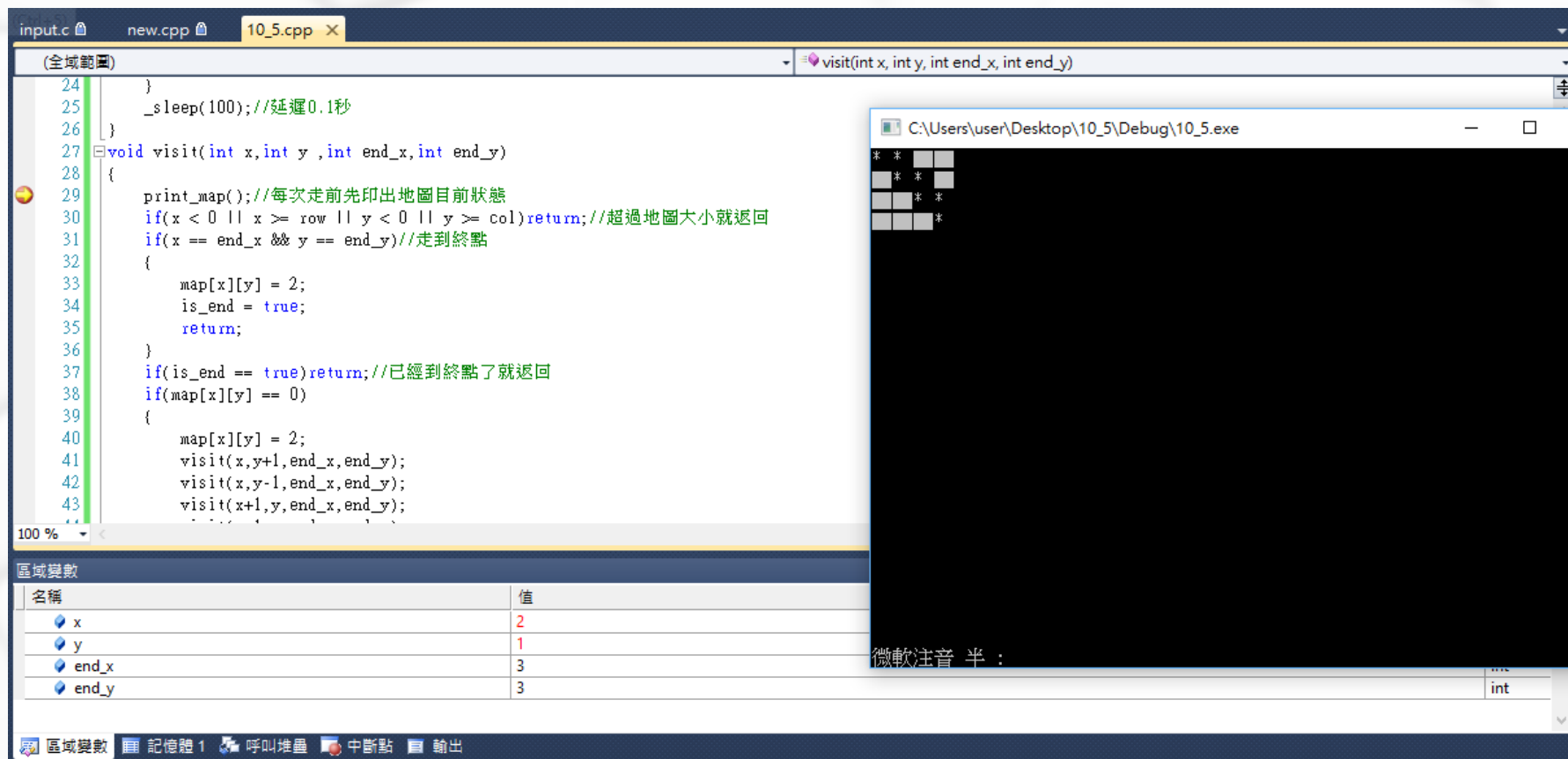
```
void print_map()  
{  
    system("cls");  
    for(int i = 0 ; i < col ; ++i)  
    {  
        for(int j = 0 ; j < row ; ++j)  
        {  
            switch(map[i][j])  
            {  
                case 0 :printf("  ");break;  
                case 1 :printf("■");break;  
                case 2 :printf("* ");break;  
            }  
        }  
        printf("\n");  
    }  
    _sleep(100); //延遲0.1秒  
}
```

走迷宮搜

```
void visit(int x,int y ,int end_x,int end_y)
{
    print_map();//每次走前先印出地圖目前狀態
    if(x < 0 || x >= row || y < 0 || y >= col)return;//超過地圖大小就返回
    if(x == end_x && y == end_y)//走到終點
    {
        map[x][y] = 2;
        is_end = true;
        return;
    }
    if(is_end == true)return;//已經到終點了就返回
    if(map[x][y] == 0)
    {
        map[x][y] = 2;
        visit(x,y+1,end_x,end_y);
        visit(x,y-1,end_x,end_y);
        visit(x+1,y,end_x,end_y);
        visit(x-1,y,end_x,end_y);
    }
}
```

還是看不懂怎麼執行的？

記得，F10/F11真的很好用



The screenshot shows a C++ IDE with a debugger window open. The code is a recursive function `visit` that explores a grid. The debugger window shows the current state of the program, with the `visit` function selected. The console window shows the output of the program, which is a grid of characters.

```
24     }
25     _sleep(100); //延遲0.1秒
26 }
27 void visit(int x, int y, int end_x, int end_y)
28 {
29     print_map(); //每次走前打印出地圖目前狀態
30     if(x < 0 || x >= row || y < 0 || y >= col) return; //超過地圖大小就返回
31     if(x == end_x && y == end_y) //走到終點
32     {
33         map[x][y] = 2;
34         is_end = true;
35         return;
36     }
37     if(is_end == true) return; //已經到終點了就返回
38     if(map[x][y] == 0)
39     {
40         map[x][y] = 2;
41         visit(x+1, end_x, end_y);
42         visit(x-1, end_x, end_y);
43         visit(x, y+1, end_x, end_y);
44         visit(x, y-1, end_x, end_y);
45     }
46 }
```

區域變數

名稱	值
x	2
y	1
end_x	3
end_y	3

100 %

C:\Users\user\Desktop\10_5\Debug\10_5.exe

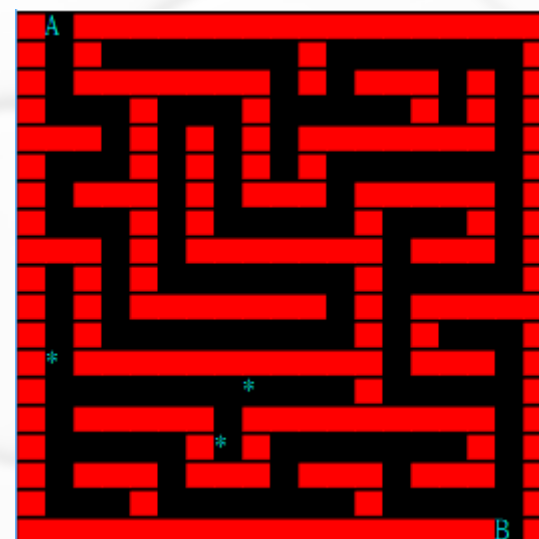
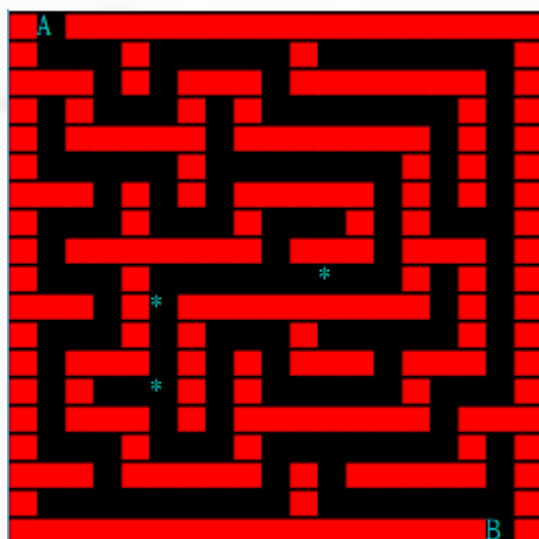
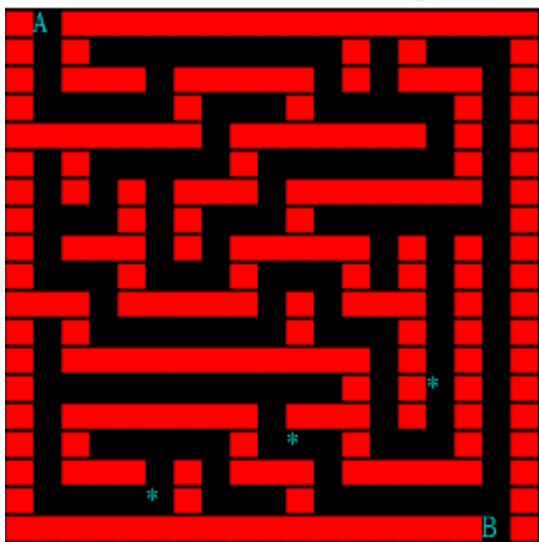
```
* * *
* * *
* * *
* * *
```

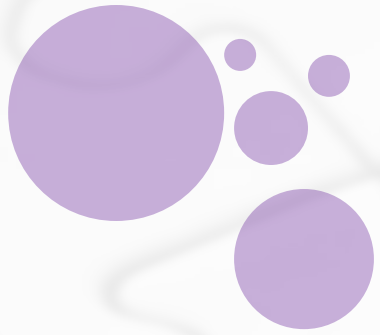
微軟注音 半 :

上學期最後一份作業還留著嗎

❖其實，隨機迷宮就是老鼠走迷宮的應用

❖最簡單的方式就是開一個全部都是牆壁的地圖，使用 **backtracking** 隨機朝著某方向前進，經過的點讓他變成路徑，直到走到終點就會是一個隨機迷宮了。





課堂練習

Q3:請列出 $\{1,2,3\}$ 的所有子集合(包含空集合)?

🔷答案總共有8種，why?

🔷做完後請將專案壓縮上傳至E平台當作這次點名～



THANK YOU