魔術方塊Magic Square

資訊四丙

D0683497

柯利韋

目錄

[一、 題目定義 2](#_Toc53527217)

[1. 作法 2](#_Toc53527218)

[2. 說明 2](#_Toc53527219)

[二、 程式碼 2](#_Toc53527220)

[三、 執行結果 5](#_Toc53527221)

[四、 心得 5](#_Toc53527222)

1. 題目定義
2. 作法

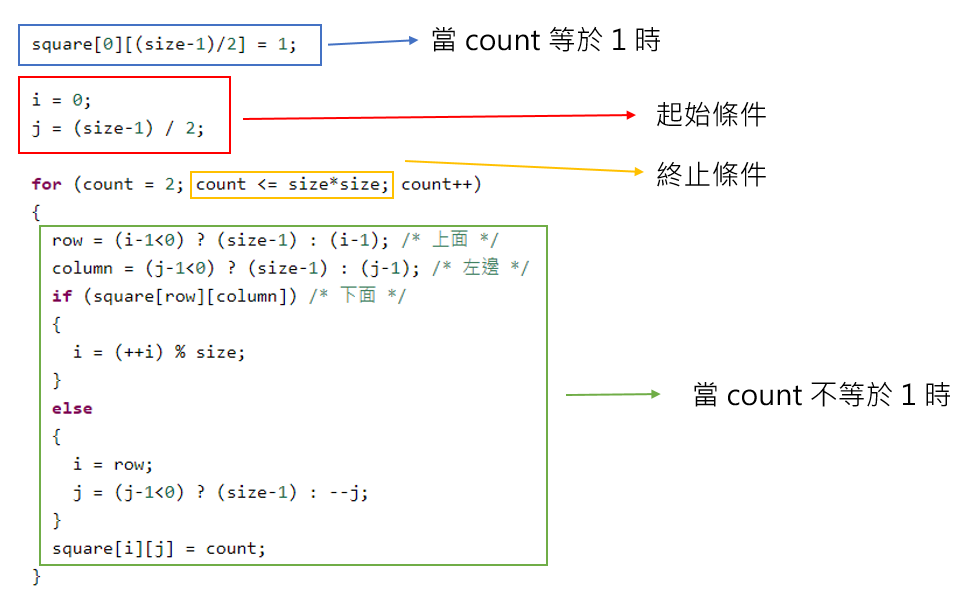
使用課本所提供的範例，並修改方塊邊長的限制條件，然後將迴圈修改成地回方式。

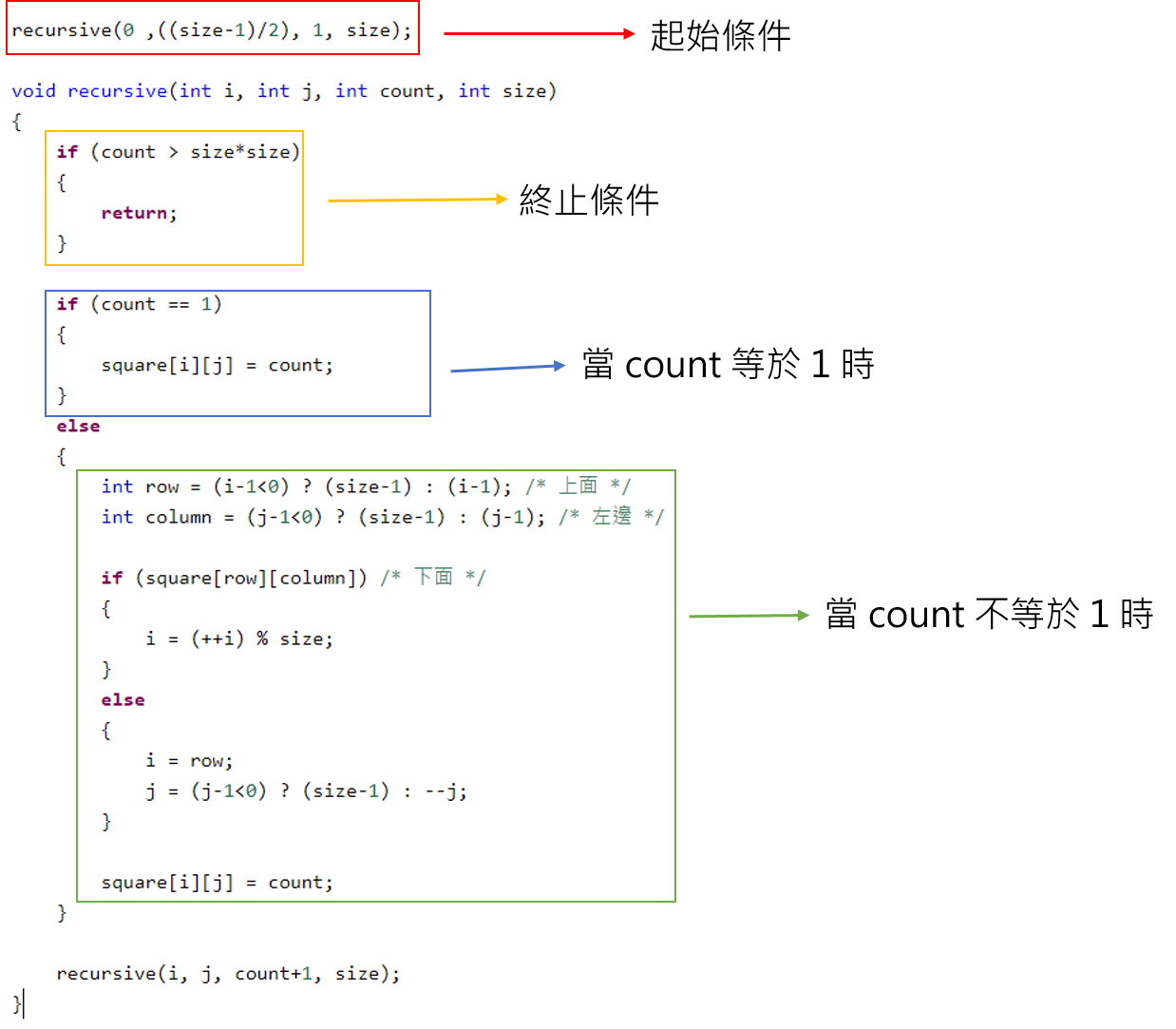
1. 說明

驗證：

因題目限制輸入為5~9因此將原本課本範例 #define MAX\_SIZE 15 改為 #define MAX\_SIZE 9 以及將原本檢查輸入的 if (size < 1 || size > MAX\_SIZE+1) 改為 if (size < 4 || size > MAX\_SIZE+1)。

迴圈改成遞迴：





步數分析：

* + - * 1. 輸入：1
        2. 檢查錯誤的輸入：2
        3. 初始化：n\*n
        4. 遞迴：

終止條件判斷：n\*n

count 是否等於 1：n\*n

* + - * 1. 輸出：n\*n

結論：Θ(n\*n)

1. 程式碼

#include<stdio.h>

#include<stdlib.h>

#define MAX\_SIZE 9 /\* 最大方陣大小 \*/

int square[MAX\_SIZE][MAX\_SIZE];

void recursive(int i, int j, int count, int size)

{

if (count > size\*size)

{

return;

}

if (count == 1)

{

square[i][j] = count;

}

else

{

int row = (i-1<0) ? (size-1) : (i-1); /\* 上面 \*/

int column = (j-1<0) ? (size-1) : (j-1); /\* 左邊 \*/

if (square[row][column]) /\* 下面 \*/

{

i = (++i) % size;

}

else

{

i = row;

j = (j-1<0) ? (size-1) : --j;

}

square[i][j] = count;

}

recursive(i, j, count+1, size);

}

void main(void)

{

/\* 反覆地產生一個大小為 n 的魔術方陣 \*/

int i, j;

int size;

printf("Enter the size of the square:");

scanf("%d", &size);

/\* 檢查錯誤的輸入 \*/

if (size < 4 || size > MAX\_SIZE+1)

{

fprintf(stderr, "Error! Size is out od range\n");

exit(EXIT\_FAILURE);

}

if (!(size % 2))

{

fprintf(stderr, "Error! Size is even\n");

exit(EXIT\_FAILURE);

}

/\* 初始化陣列 \*/

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

{

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

{

square[i][j] = 0;

}

}

recursive(0 ,((size-1)/2), 1, size);

/\* 輸出魔術方陣 \*/

printf("Magic Square of size %d : \n\n", size);

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

{

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

{

printf("%5d", square[i][j]);

}

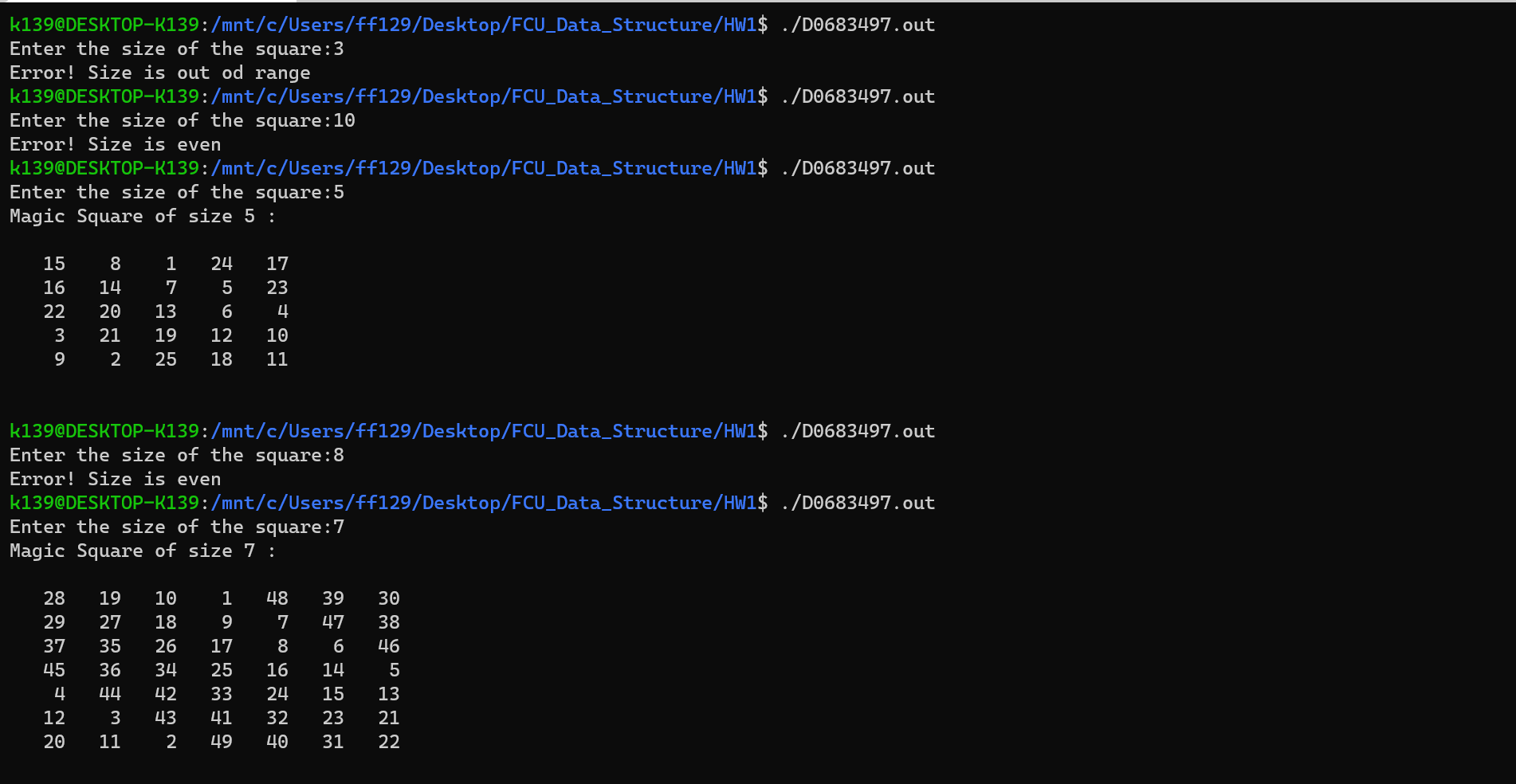
printf("\n");

}

printf("\n\n");

}

1. 執行結果



1. 心得

一開始在理解魔方陣的定義時遇到了一些困難，之後就沒什麼問題了。