114學年度第1學期 組合語言與嵌入式系統 — Final Project

老師：朱守禮 老師

班組別：資訊二乙-3

學生： 11327214 江竑燁 11327216 陳伊軒 11327217 蔡易勳

一、程式說明

1.name.s 和 id.s 都是沿用期中 project 的程式碼

2.extern 的三大功用

1. 跨檔案使用函式或變數
2. 避免重複定義符號
3. 讓 linker 正確進行符號解析（symbol resolution）

3.

本程式最核心的部分為雙層迴圈結構，用來逐一處理畫面中每一個像素點。外層迴圈 loopwidth 負責控制畫面在水平方向（x 軸）的掃描，從左至右依序處理每一個位置，內層迴圈 loopheight 則負責垂直方向（y 軸）的掃描，從上到下依序計算每個像素。

在每一組 x、y 座標中，程式會先計算對應的初始複數值，接著進入 while 迴圈進行 Julia Set 的反覆運算。此 while 迴圈會根據運算結果是否超出範圍或是否達到最大迭代次數來決定是否跳出。當跳出迴圈後，會依據剩餘的迭代次數計算顏色，並將結果寫入 frame buffer，完成單一像素的繪製。透過此多層迴圈設計，程式能完整繪製整張 Julia Set 圖形。

二、設計重點說明

1.

frame 宣告是：int16_t (*frame)[FRAME_WIDTH]

代表它指向的是一個一個 row，每個 row 有 FRAME_WIDTH 個 int16_t

sizeof(int16_t) = 2 bytes

=> 用 LDRH / STRH

=> FRAME_WIDTH = 640, FRAME_HEIGHT = 480, int16_t = 2 bytes

=> frame 總大小 = 640 × 480 × 2 = 614,400 bytes

=> 轉成 16 進位： 0x96000

假設我frame 起始位置為 0xBEF69498

0xBEF69498 + 0x00096000 = 0xBEFFF498（結束位置）

2.Operand2 格式：

```
mov r5, #1
sub r2, r5, r2
add r0, r0, r5, lsl #1
cmp r5, #0
```

3.非 Branch 指令的 Conditional Execution:

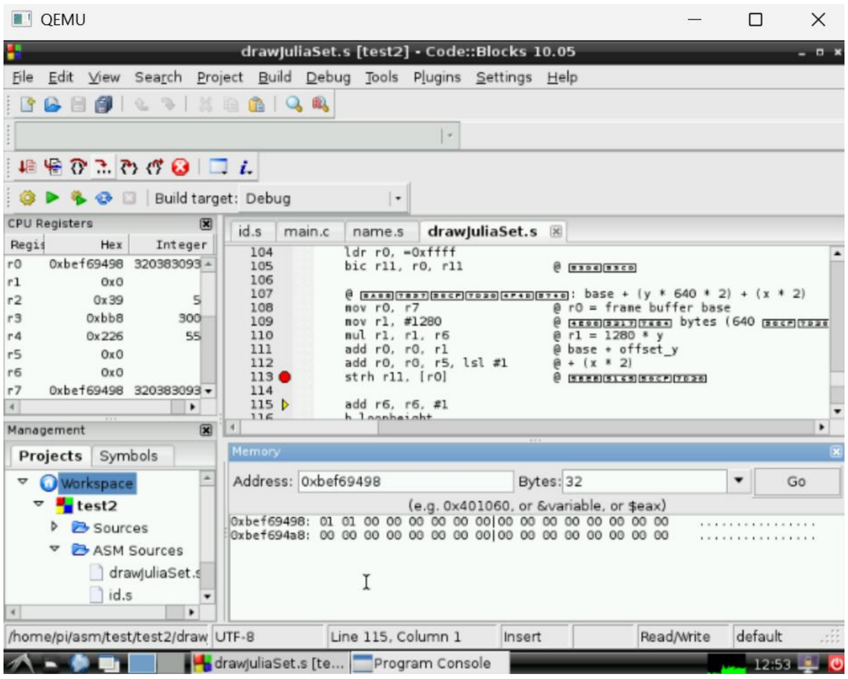
```
movge r5, #0
movne r6, #0
moveq r6, #0
```

4.指定的必執行指令：

adds lr, sp, pc（第四道）直接加上去程式碼不會編譯錯誤

三、程式驗證結果

frame start



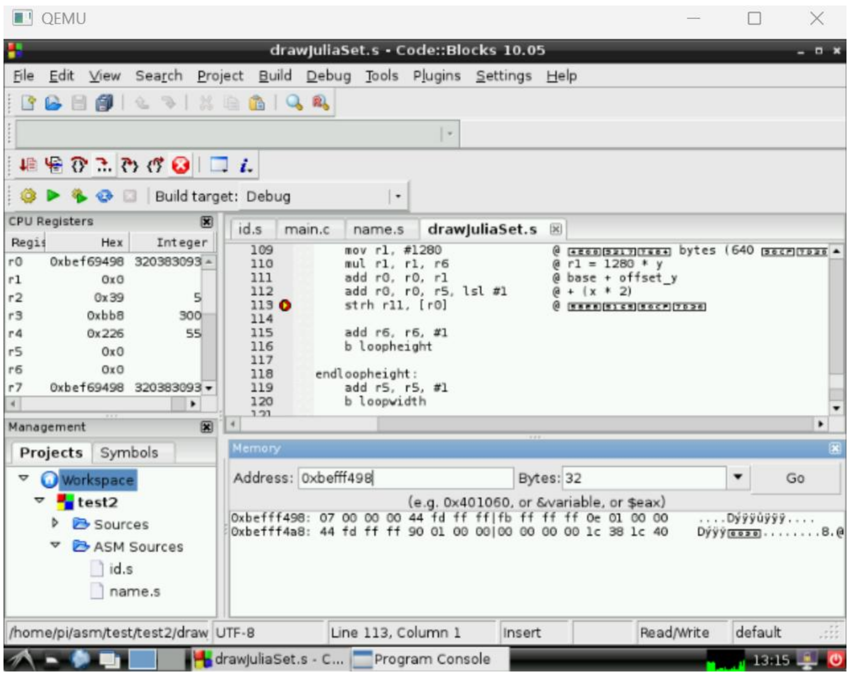
```
        ldr r0, =0xffff
        bic r11, r0, r11                @ [illegible]

        @ [illegible]: base + (y * 640 * 2) + (x * 2)
        mov r0, r7                      @ r0 = frame buffer base
        mov r1, #1280                   @ [illegible] bytes (640 [illegible]
        mul r1, r1, r6                  @ r1 = 1280 * y
        add r0, r0, r1                  @ base + offset_y
        add r0, r0, r5, lsl #1          @ + (x * 2)
        strh r11, [r0]                  @ [illegible]

        add r6, r6, #1
        b loopheight
```

```
r0   0xbef69498  320383093
r1   0x0
r2   0x39        5
r3   0xbb8       300
r4   0x226       55
r5   0x0
r6   0x0
r7   0xbef69498  320383093
```

Address: 0xbef69498 Bytes: 32

```
0xbef69498: 01 01 00 00 00 00 00 00|00 00 00 00 00 00 00 00   ................
0xbef694a8: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00   ................
```

frame end



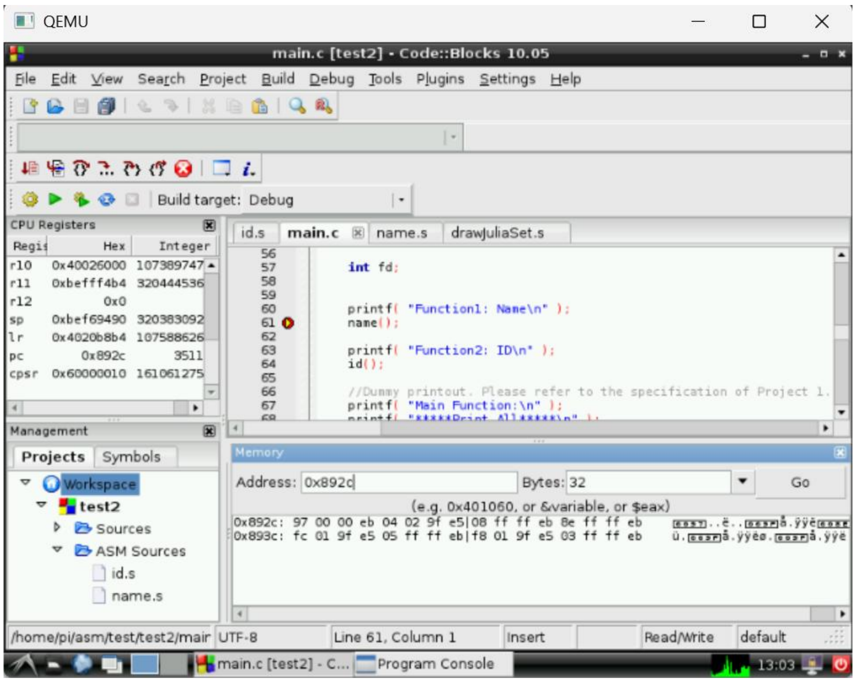
```
        mov r1, #1280                   @ [illegible] bytes (640 [illegible]
        mul r1, r1, r6                  @ r1 = 1280 * y
        add r0, r0, r1                  @ base + offset_y
        add r0, r0, r5, lsl #1          @ + (x * 2)
        strh r11, [r0]                  @ [illegible]

        add r6, r6, #1
        b loopheight

endloopheight:
        add r5, r5, #1
        b loopwidth
```

```
r0   0xbef69498  320383093
r1   0x0
r2   0x39        5
r3   0xbb8       300
r4   0x226       55
r5   0x0
r6   0x0
r7   0xbef69498  320383093
```

Address: 0xbefff498 Bytes: 32

```
0xbefff498: 07 00 00 00 44 fd ff ff|fb ff ff ff 0e 01 00 00   ....Dýÿÿûÿÿÿ....
0xbefff4a8: 44 fd ff ff 90 01 00 00|00 00 00 00 1c 38 1c 40   Dýÿÿ[illegible]........8.@
```

name.s function start



```
CPU Registers
Regis   Hex          Integer
r10     0x40026000   107389747
r11     0xbefff4b4   320444536
r12     0x0
sp      0xbef69490   320383092
lr      0x4020b8b4   107588626
pc      0x892c       3511
cpsr    0x60000010   161061275
```

```c
    int fd;


    printf( "Function1: Name\n" );
    name();

    printf( "Function2: ID\n" );
    id();

    //Dummy printout. Please refer to the specification of Project 1.
    printf( "Main Function:\n" );
```

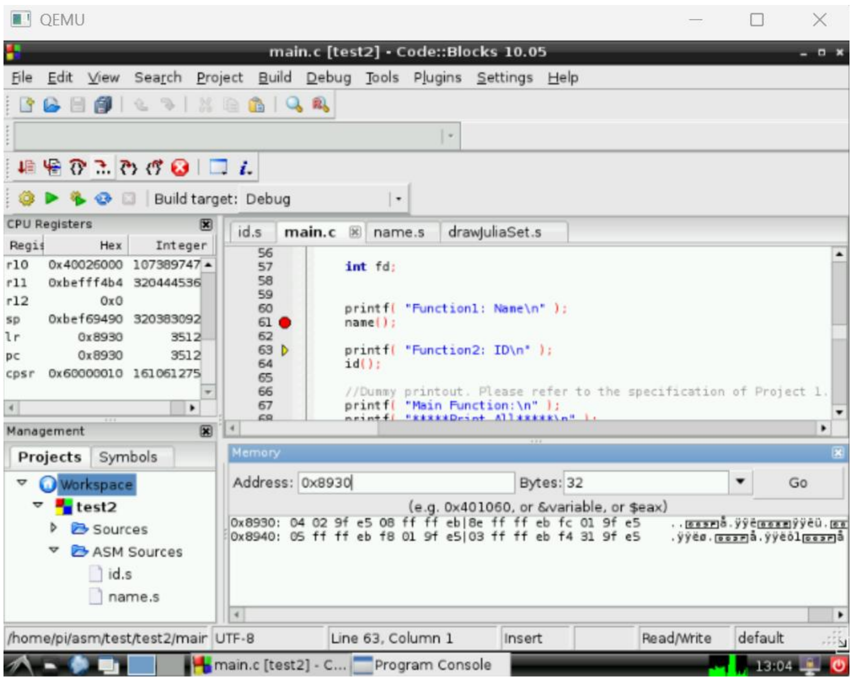
Address: 0x892c Bytes: 32

(e.g. 0x401060, or &variable, or $eax)

```
0x892c: 97 00 00 eb 04 02 9f e5|08 ff ff eb 8e ff ff eb
0x893c: fc 01 9f e5 05 ff ff eb|f8 01 9f e5 03 ff ff eb
```

/home/pi/asm/test/test2/main UTF-8 Line 61, Column 1 Insert Read/Write default

name function end



```
CPU Registers
Regis   Hex          Integer
r10     0x40026000   107389747
r11     0xbefff4b4   320444536
r12     0x0
sp      0xbef69490   320383092
lr      0x8930       3512
pc      0x8930       3512
cpsr    0x60000010   161061275
```

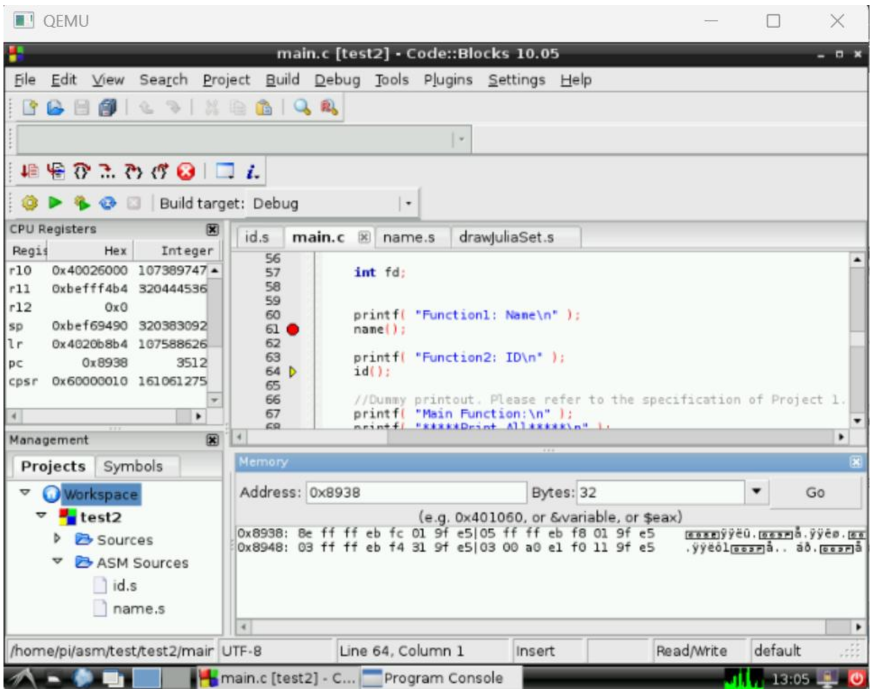
```c
    int fd;


    printf( "Function1: Name\n" );
    name();

    printf( "Function2: ID\n" );
    id();

    //Dummy printout. Please refer to the specification of Project 1.
    printf( "Main Function:\n" );
```

Address: 0x8930 Bytes: 32

(e.g. 0x401060, or &variable, or $eax)

```
0x8930: 04 02 9f e5 08 ff ff eb|8e ff ff eb fc 01 9f e5
0x8940: 05 ff ff eb f8 01 9f e5|03 ff ff eb f4 31 9f e5
```

/home/pi/asm/test/test2/main UTF-8 Line 63, Column 1 Insert Read/Write default id.s function start



```
CPU Registers
Regis        Hex     Integer
r10   0x40026000   107389747
r11   0xbefff4b4   320444536
r12          0x0
sp    0xbef69490   320383092
lr    0x4020b8b4   107588626
pc        0x8938        3512
cpsr  0x60000010   161061275
```

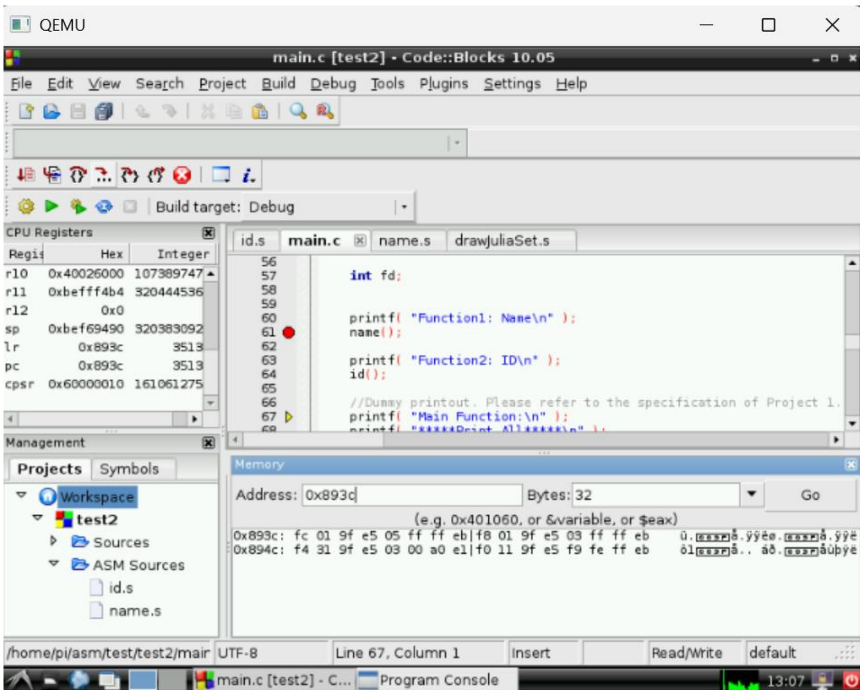
id.s | main.c | name.s | drawJuliaSet.s

```c
    int fd;


    printf( "Function1: Name\n" );
    name();

    printf( "Function2: ID\n" );
    id();

    //Dummy printout. Please refer to the specification of Project 1.
    printf( "Main Function:\n" );
```

```
Memory
Address: 0x8938          Bytes: 32          Go
(e.g. 0x401060, or &variable, or $eax)
0x8938: 8e ff ff eb fc 01 9f e5|05 ff ff eb f8 01 9f e5
0x8948: 03 ff ff eb f4 31 9f e5|03 00 a0 e1 f0 11 9f e5
```

/home/pi/asm/test/test2/mair | UTF-8 | Line 64, Column 1 | Insert | Read/Write | default

id function end



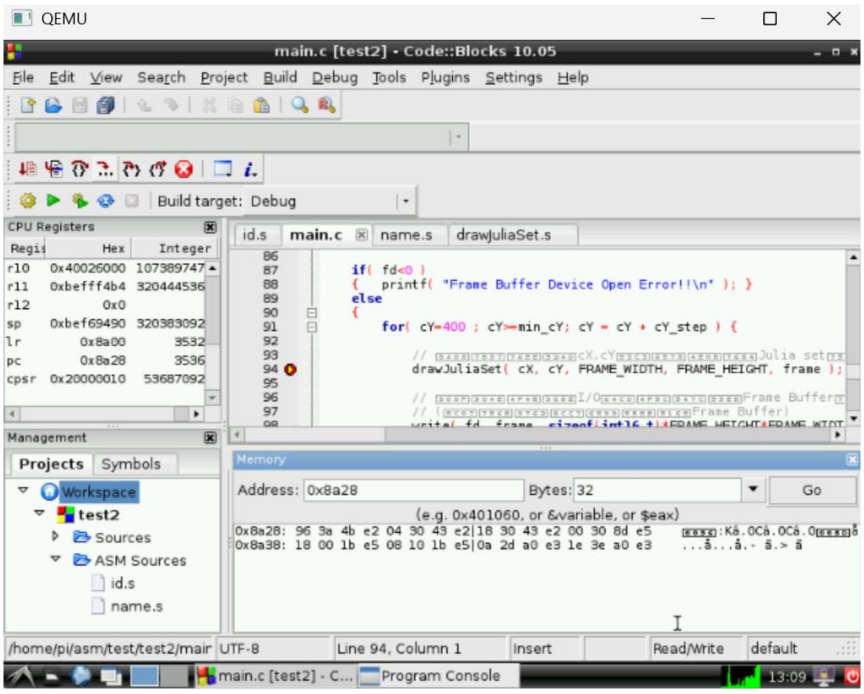
```
CPU Registers
Regis        Hex     Integer
r10   0x40026000   107389747
r11   0xbefff4b4   320444536
r12          0x0
sp    0xbef69490   320383092
lr        0x893c        3513
pc        0x893c        3513
cpsr  0x60000010   161061275
```

id.s | main.c | name.s | drawJuliaSet.s

```c
    int fd;


    printf( "Function1: Name\n" );
    name();

    printf( "Function2: ID\n" );
    id();

    //Dummy printout. Please refer to the specification of Project 1.
    printf( "Main Function:\n" );
```

```
Memory
Address: 0x893c          Bytes: 32          Go
(e.g. 0x401060, or &variable, or $eax)
0x893c: fc 01 9f e5 05 ff ff eb|f8 01 9f e5 03 ff ff eb
0x894c: f4 31 9f e5 03 00 a0 e1|f0 11 9f e5 f9 fe ff eb
```

/home/pi/asm/test/test2/mair | UTF-8 | Line 67, Column 1 | Insert | Read/Write | default drawJuliaSet function start



```
Regis        Hex     Integer
r10   0x40026000   107389747
r11   0xbefff4b4   320444536
r12          0x0
sp    0xbef69490   320383092
lr        0x8a00        3532
pc        0x8a28        3536
cpsr  0x20000010    53687092
```

id.s | main.c | name.s | drawJuliaSet.s

```
    if( fd<0 )
    {   printf( "Frame Buffer Device Open Error!!\n" ); }
    else
    {
        for( cY=400 ; cY>=min_cY; cY = cY + cY_step ) {

            // cX,cY Julia set
            drawJuliaSet( cX, cY, FRAME_WIDTH, FRAME_HEIGHT, frame );

            // I/O Frame Buffer
            // ( Frame Buffer)
```

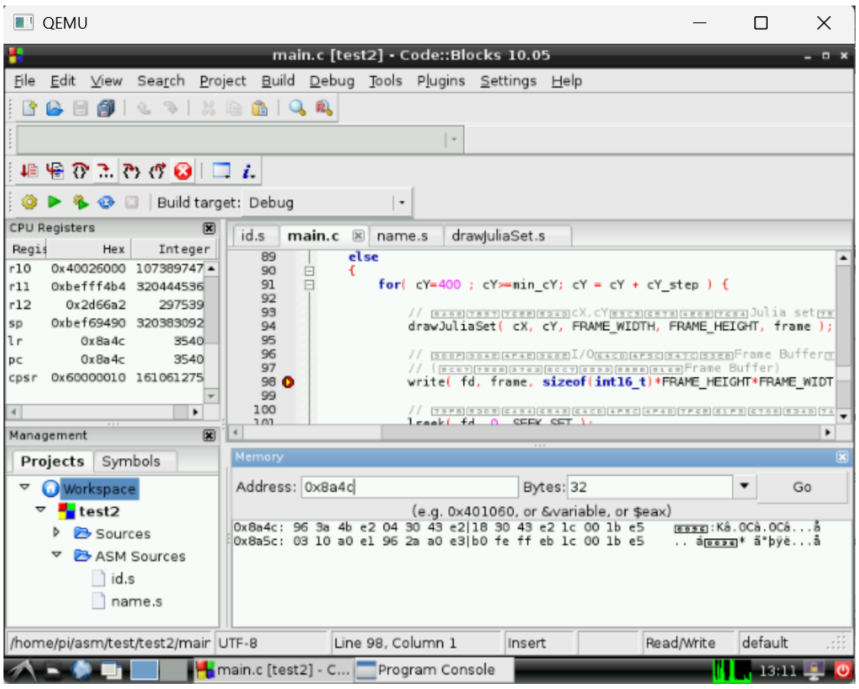
Address: 0x8a28 Bytes: 32

(e.g. 0x401060, or &variable, or $eax)

```
0x8a28: 96 3a 4b e2 04 30 43 e2|18 30 43 e2 00 30 8d e5    :Kâ.0Câ.0Câ.0 å
0x8a38: 18 00 1b e5 08 10 1b e5|0a 2d a0 e3 1e 3e a0 e3    ...å...å.- ã.> ã
```

/home/pi/asm/test/test2/mair | UTF-8 | Line 94, Column 1 | Insert | Read/Write | default

drawJuliaSet function end



```
Regis        Hex     Integer
r10   0x40026000   107389747
r11   0xbefff4b4   320444536
r12     0x2d66a2      297539
sp    0xbef69490   320383092
lr        0x8a4c        3540
pc        0x8a4c        3540
cpsr  0x60000010   161061275
```

id.s | main.c | name.s | drawJuliaSet.s

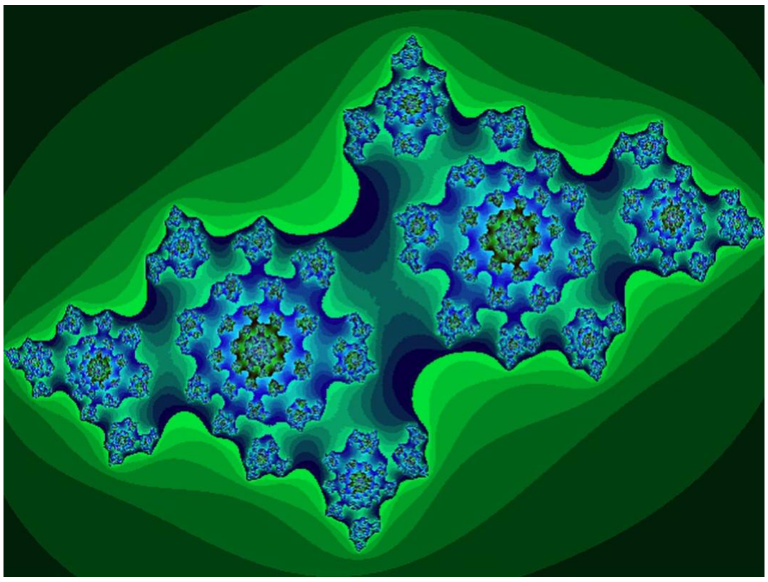
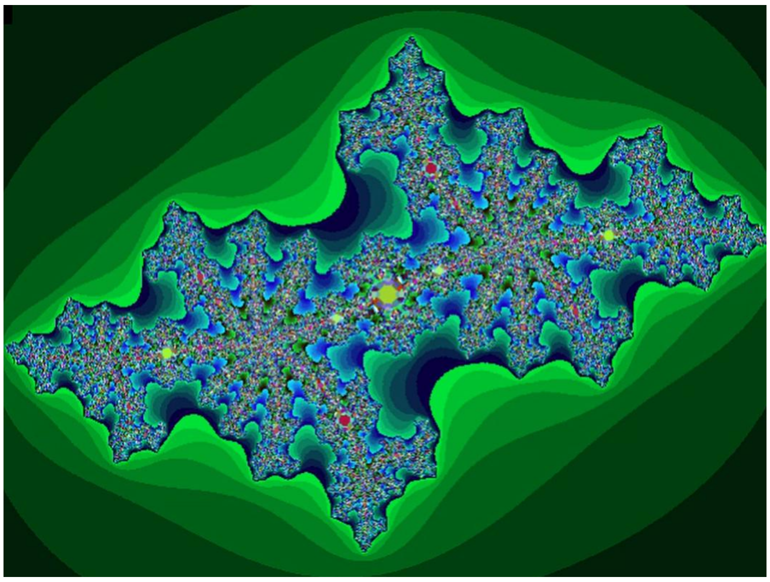
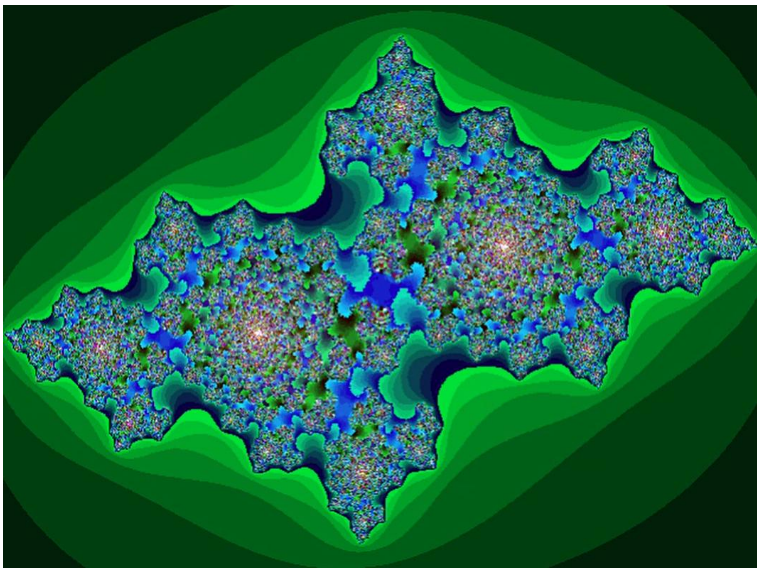
```
    else
    {
        for( cY=400 ; cY>=min_cY; cY = cY + cY_step ) {

            // cX,cY Julia set
            drawJuliaSet( cX, cY, FRAME_WIDTH, FRAME_HEIGHT, frame );

            // I/O Frame Buffer
            // ( Frame Buffer)
            write( fd, frame, sizeof(int16_t)*FRAME_HEIGHT*FRAME_WIDT

            //
```

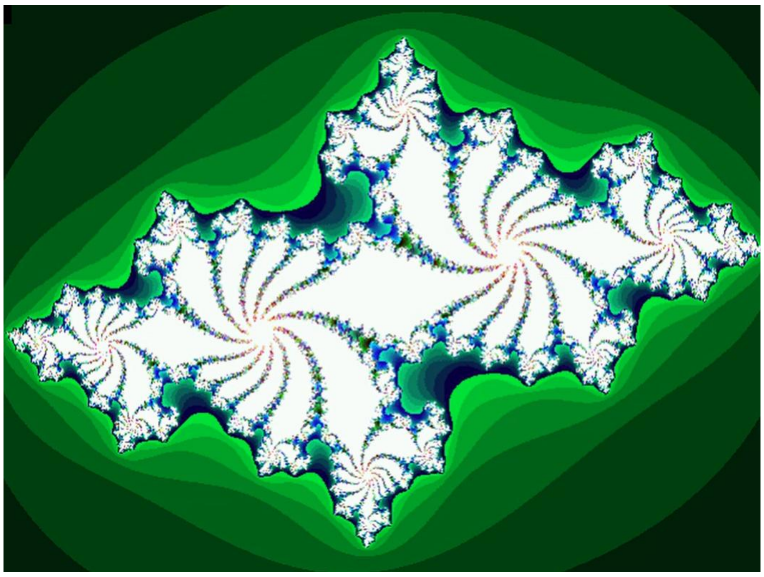
Address: 0x8a4c Bytes: 32

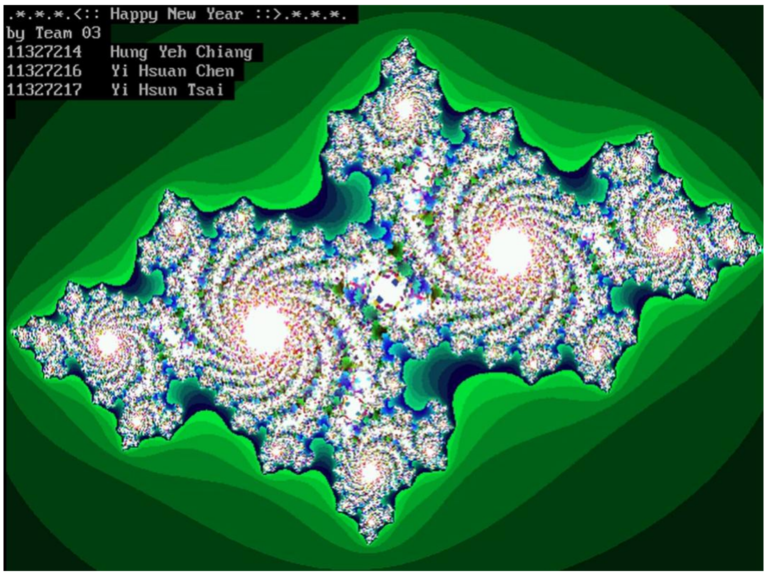
(e.g. 0x401060, or &variable, or $eax)

```
0x8a4c: 96 3a 4b e2 04 30 43 e2|18 30 43 e2 1c 00 1b e5    :Kâ.0Câ.0Câ...å
0x8a5c: 03 10 a0 e1 96 2a a0 e3|b0 fe ff eb 1c 00 1b e5    .. á * ã*þÿë...å
```

/home/pi/asm/test/test2/mair | UTF-8 | Line 98, Column 1 | Insert | Read/Write | default





```
.*.*.*.<:: Happy New Year ::>.*.*.*.
by Team 03
11327214   Hung Yeh Chiang
11327216   Yi Hsuan Chen
11327217   Yi Hsun Tsai
```

四、心得感想

這次的 function 實作相較上一次明顯困難許多，不僅程式中包含更多層的迴圈與條件判斷，也花一些時間在完成指令要求。在組合語言中，迴圈與流程控制需要自行處理每一次比較與跳轉，只要有一個細節出錯就會影響整體結果。實作過程中，常需要反覆測試與除錯，確認暫存器的數值是否正確。

五、各組員分工方式與負責項目

共同完成