112 學年 第一學期 計算機組織與結構 作業報告 2 日期:2023/12/12

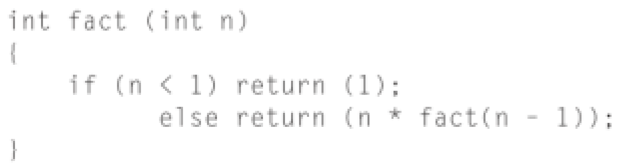
班級:資工三學號:b1043009 姓名:賴韋捷

1. 撰寫 MIPS 程式 (共 2 題，100 分，滿分 100 分)

請於 2023/12/12 前上傳至 M 數位園區作業區繳交

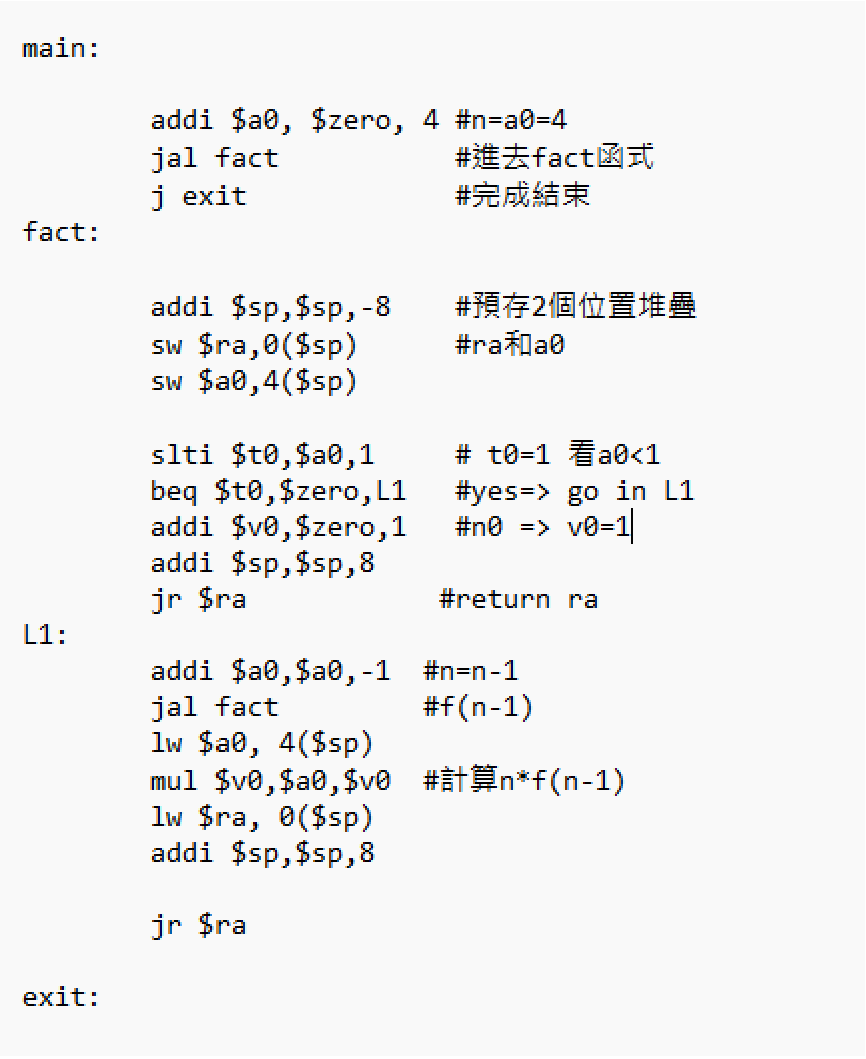
請安裝 QtSpim (http://spimsimulator.sourceforge.net/) 模擬器，並請詳細參考課本第二章及附錄 A 的介紹，於 QtSpim 模擬器環境下，撰寫一完整的 MIPS 核心指令集版本的程式。(需**貼完整程式碼，截圖呈現結果並文字說明**。)

(1) 實作第二章範例階層計算 fact()，須包含呼叫階層計算之主函式，觀察暫存器及記憶體狀態並說明程式之運作。(50 分)



```
int fact (int n)
{
    if (n < 1) return (1);
        else return (n * fact(n - 1));
}
```

程式碼如下



```
main:

        addi $a0, $zero, 4 #n=a0=4
        jal fact           #進去fact函式
        j exit             #完成結束
fact:

        addi $sp,$sp,-8    #預存2個位置堆疊
        sw $ra,0($sp)      #ra和a0
        sw $a0,4($sp)

        slti $t0,$a0,1     # t0=1 看a0<1
        beq $t0,$zero,L1   #yes=> go in L1
        addi $v0,$zero,1   #n0 => v0=1
        addi $sp,$sp,8
        jr $ra            #return ra
L1:
        addi $a0,$a0,-1  #n=n-1
        jal fact         #f(n-1)
        lw $a0, 4($sp)
        mul $v0,$a0,$v0  #計算n*f(n-1)
        lw $ra, 0($sp)
        addi $sp,$sp,8

        jr $ra

exit:
```

直接在 main 設定 n=4

並且執行 f(4)

在 fact 上

堆疊分配 2 個值的空間

返回地址 ra 和參數 a0 保存在裡面

如果 n<=1 值 v0=1

那如果大於 1 就進入 L1function

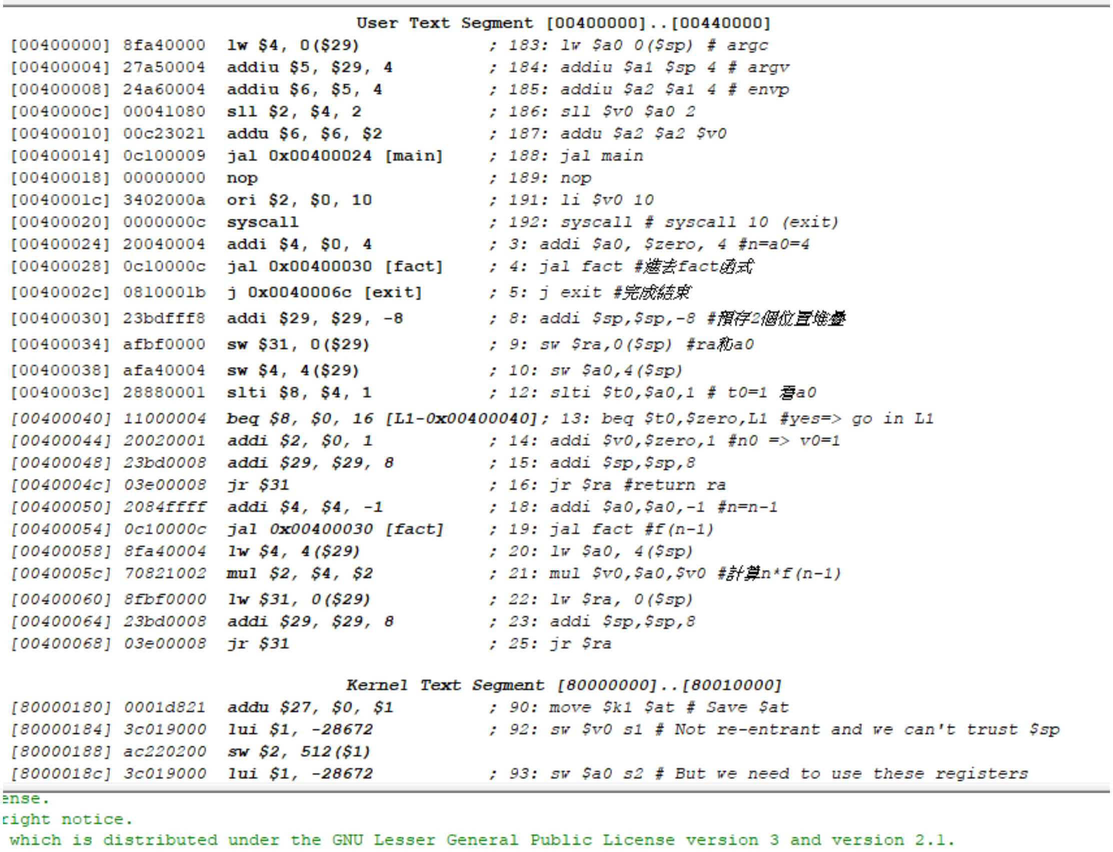
L1:

n=n-1

fact(n-1)

計算 n*fact(n-1)

返回結束



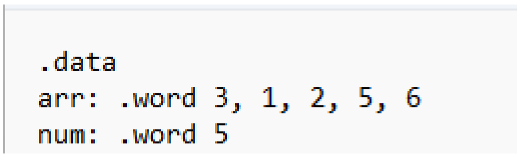
```
                              User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)           ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4        ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4         ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2           ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2         ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]   ; 188: jal main
[00400018] 00000000  nop                     ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10          ; 191: li $v0 10
[00400020] 0000000c  syscall                 ; 192: syscall # syscall 10 (exit)
[00400024] 20040004  addi $4, $0, 4          ; 3: addi $a0, $zero, 4 #n=a0=4
[00400028] 0c10000c  jal 0x00400030 [fact]   ; 4: jal fact #進去fact函式
[0040002c] 0810001b  j 0x0040006c [exit]     ; 5: j exit #完成結束
[00400030] 23bdfff8  addi $29, $29, -8       ; 8: addi $sp,$sp,-8 #預存2個位置堆疊
[00400034] afbf0000  sw $31, 0($29)          ; 9: sw $ra,0($sp) #ra和a0
[00400038] afa40004  sw $4, 4($29)           ; 10: sw $a0,4($sp)
[0040003c] 28880001  slti $8, $4, 1          ; 12: slti $t0,$a0,1 # t0=1 看a0
[00400040] 11000004  beq $8, $0, 16 [L1-0x00400040]; 13: beq $t0,$zero,L1 #yes=> go in L1
[00400044] 20020001  addi $2, $0, 1          ; 14: addi $v0,$zero,1 #n0 => v0=1
[00400048] 23bd0008  addi $29, $29, 8        ; 15: addi $sp,$sp,8
[0040004c] 03e00008  jr $31                  ; 16: jr $ra #return ra
[00400050] 2084ffff  addi $4, $4, -1         ; 18: addi $a0,$a0,-1 #n=n-1
[00400054] 0c10000c  jal 0x00400030 [fact]   ; 19: jal fact #f(n-1)
[00400058] 8fa40004  lw $4, 4($29)           ; 20: lw $a0, 4($sp)
[0040005c] 70821002  mul $2, $4, $2          ; 21: mul $v0,$a0,$v0 #計算n*f(n-1)
[00400060] 8fbf0000  lw $31, 0($29)          ; 22: lw $ra, 0($sp)
[00400064] 23bd0008  addi $29, $29, 8        ; 23: addi $sp,$sp,8
[00400068] 03e00008  jr $31                  ; 25: jr $ra

                             Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1        ; 90: move $k1 $at # Save $at
[80000184] 3c019000  lui $1, -28672          ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
[8000018c] 3c019000  lui $1, -28672          ; 93: sw $a0 s2 # But we need to use these registers
```

```
ense.
right notice.
 which is distributed under the GNU Lesser General Public License version 3 and version 2.1.
```

執行結果

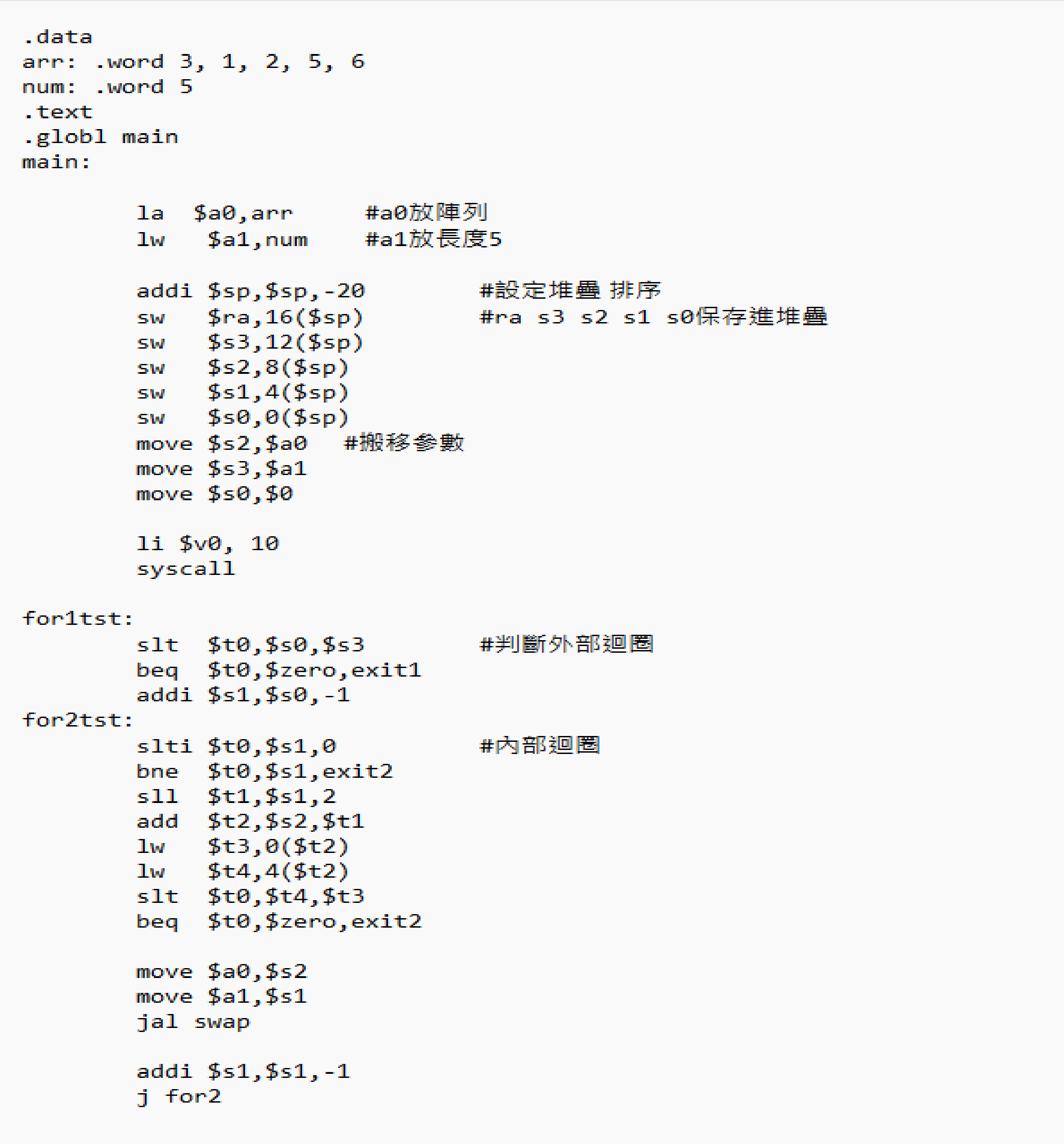
(2) 實作第二章範例泡泡排序演算法，須包含 sort()函式及 swap()函式，觀察暫存器及記憶體狀態並說明程式之運作。(50 分)

根據投影片的提示，我只需要設定初始的值以及 main 的部分
初始的陣列我設定是 3,1,2,5,6



```
.data
arr: .word 3, 1, 2, 5, 6
num: .word 5
```

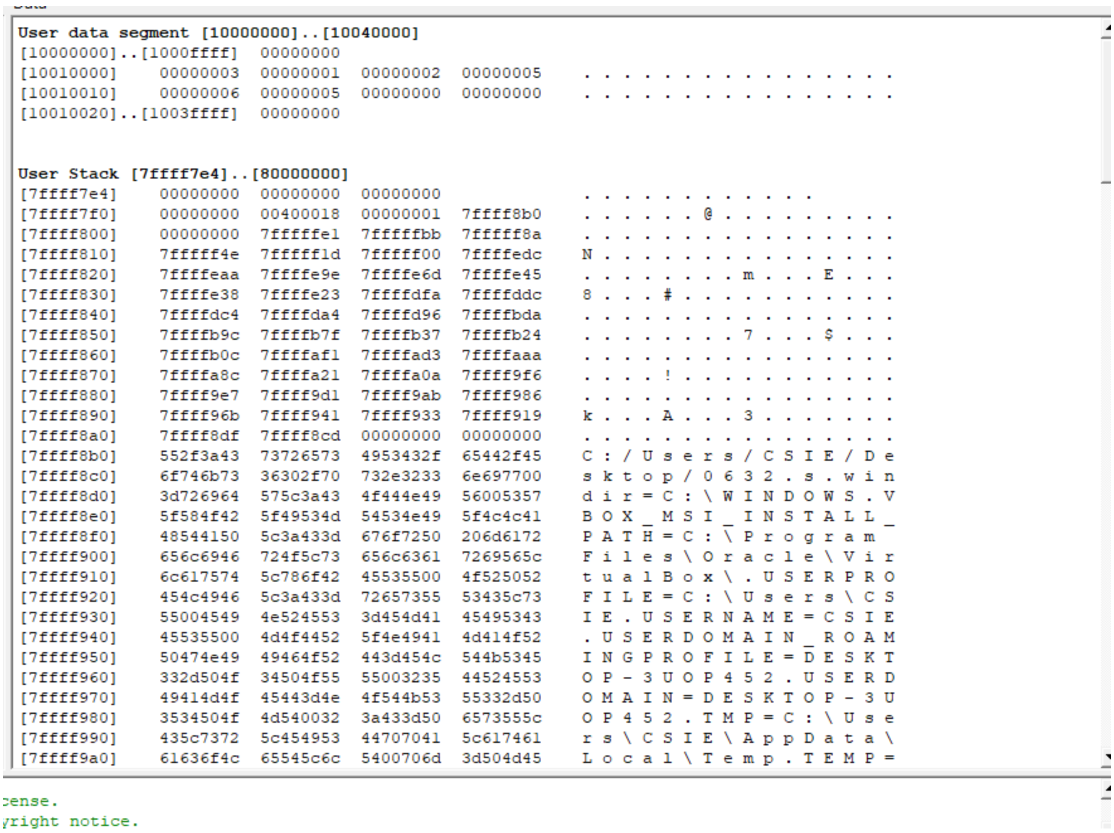
預設結果會是 1,2,3,5,6
下圖程式碼



```
.data
arr: .word 3, 1, 2, 5, 6
num: .word 5
.text
.globl main
main:

        la   $a0,arr      #a0放陣列
        lw    $a1,num     #a1放長度5

        addi $sp,$sp,-20         #設定堆疊 排序
        sw   $ra,16($sp)         #ra s3 s2 s1 s0保存進堆疊
        sw   $s3,12($sp)
        sw   $s2,8($sp)
        sw   $s1,4($sp)
        sw   $s0,0($sp)
        move $s2,$a0   #搬移參數
        move $s3,$a1
        move $s0,$0

        li $v0, 10
        syscall

for1tst:
        slt  $t0,$s0,$s3         #判斷外部迴圈
        beq  $t0,$zero,exit1
        addi $s1,$s0,-1
for2tst:
        slti $t0,$s1,0           #內部迴圈
        bne  $t0,$s1,exit2
        sll  $t1,$s1,2
        add  $t2,$s2,$t1
        lw   $t3,0($t2)
        lw   $t4,4($t2)
        slt  $t0,$t4,$t3
        beq  $t0,$zero,exit2

        move $a0,$s2
        move $a1,$s1
        jal swap

        addi $s1,$s1,-1
        j for2
```

```
exit2:
        addi $s0,$s0,1
        j for2

exit1:
        lw   $ra,16($sp)
        lw   $s3,12($sp)
        lw   $s2,8($sp)
        lw   $s1,4($sp)
        lw   $s0,0($sp)
        addi $sp,$sp,20
        jr   $ra


swap:
        sll $t1,$a1,2 #$t1=k*4
        add $t1,$t1,$a1 #$t1=v+(k*4)
        lw  $t0,0($t1)  #$t0(temp)=v[k]
        lw  $t2,4($t1)  #$t2=v[k+1]
        sw  $t0,4($t1)  #v[k]=t2
        sw  $t2,0($t1)  #v[k+1]=t0
        jr  $ra
```

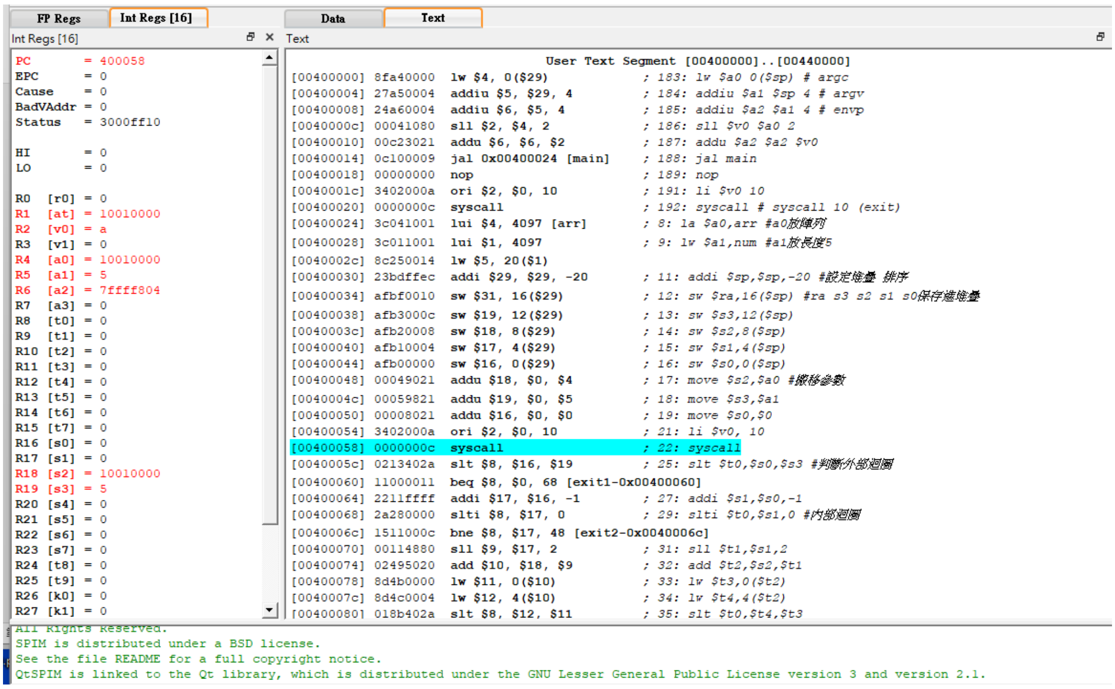
這個氣泡排序法
他會有兩個迴圈 所以我用 for1tst 和 for2tst 表示
由小排到大
所以在外部迴圈 她可以一步一步確定最小的排在第一個 依序排
而內部迴圈是為了要解決大小的關係，如果目標數值大於就交換往後排
這樣一來一往最後就可以解決排序的問題



```
User data segment [10000000]..[10040000]
[10000000]..[1000ffff]  00000000
[10010000]    00000003  00000001  00000002  00000005    . . . . . . . . . . . . . . . .
[10010010]    00000006  00000005  00000000  00000000    . . . . . . . . . . . . . . . .
[10010020]..[1003ffff]  00000000


User Stack [7ffff7e4]..[80000000]
[7ffff7e4]    00000000  00000000  00000000              . . . . . . . . . . . .
[7ffff7f0]    00000000  00400018  00000001  7ffff8b0    . . . . . . . @ . . . . . . . .
[7ffff800]    00000000  7ffffe1  7ffffbb  7ffff8a       . . . . . . . . . . . . . . . .
[7ffff810]    7fffff4e  7fffff1d  7fffff00  7ffffedc    N . . . . . . . . . . . . . . .
[7ffff820]    7ffffeaa  7ffffe9e  7ffffe6d  7ffffe45    . . . . . . . . . m . . . E . . .
[7ffff830]    7ffffe38  7ffffe23  7ffffdfa  7ffffddc    8 . . . # . . . . . . . . . . .
[7ffff840]    7ffffdc4  7ffffda4  7ffffd96  7ffffbda    . . . . . . . . . . . . . . . .
[7ffff850]    7ffffb9c  7ffffb7f  7ffffb37  7ffffb24    . . . . . . . . 7 . . . $ . . .
[7ffff860]    7ffffb0c  7ffffaf1  7ffffad3  7ffffaaa    . . . . . . . . . . . . . . . .
[7ffff870]    7ffffa8c  7ffffa21  7ffffa0a  7ffff9f6    . . . . ! . . . . . . . . . . .
[7ffff880]    7ffff9e7  7ffff9d1  7ffff9ab  7ffff986    . . . . . . . . . . . . . . . .
[7ffff890]    7ffff96b  7ffff941  7ffff933  7ffff919    k . . . A . . . 3 . . . . . . .
[7ffff8a0]    7ffff8df  7ffff8cd  00000000  00000000    . . . . . . . . . . . . . . . .
[7ffff8b0]    552f3a43  73726573  4953432f  65442f45    C : / U s e r s / C S I E / D e
[7ffff8c0]    6f746b73  36302f70  732e3233  6e697700    s k t o p / 0 6 3 2 . s . w i n
[7ffff8d0]    3d726964  575c3a43  4f444e49  56005357    d i r = C : \ W I N D O W S . V
[7ffff8e0]    5f584f42  5f49534d  54534e49  5f4c4c41    B O X _ M S I _ I N S T A L L _
[7ffff8f0]    48544150  5c3a433d  676f7250  206d6172    P A T H = C : \ P r o g r a m
[7ffff900]    656c6946  724f5c73  656c6361  7269565c    F i l e s \ O r a c l e \ V i r
[7ffff910]    6c617574  5c786f42  45535500  4f525052    t u a l B o x \ . U S E R P R O
[7ffff920]    454c4946  5c3a433d  72657355  53435c73    F I L E = C : \ U s e r s \ C S
[7ffff930]    55004549  4e524553  3d454d41  45495343    I E . U S E R N A M E = C S I E
[7ffff940]    45535500  4d4f4452  5f4e4941  4d414f52    . U S E R D O M A I N _ R O A M
[7ffff950]    50474e49  49464f52  443d454c  544b5345    I N G P R O F I L E = D E S K T
[7ffff960]    332d504f  34504f55  55003235  44524553    O P - 3 U O P 4 5 2 . U S E R D
[7ffff970]    49414d4f  45443d4e  4f544b53  55332d50    O M A I N = D E S K T O P - 3 U
[7ffff980]    3534504f  4d540032  3a433d50  6573555c    O P 4 5 2 . T M P = C : \ U s e
[7ffff990]    435c7372  5c454953  44707041  5c617461    r s \ C S I E \ A p p D a t a \
[7ffff9a0]    61636f4c  65545c6c  5400706d  3d504d45    L o c a l \ T e m p . T E M P =
```

cense.
yright notice.

紅色框起來得部分 可以知道我設定的陣列 和長度都有在上面



```
PC       = 400058
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 10010000
R5  [a1] = 5
R6  [a2] = 7ffff804
R7  [a3] = 0
R8  [t0] = 0
R9  [t1] = 0
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 10010000
R19 [s3] = 5
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
```

```
                              User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)                ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4             ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4              ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2                ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2              ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]        ; 188: jal main
[00400018] 00000000  nop                          ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10               ; 191: li $v0 10
[00400020] 0000000c  syscall                      ; 192: syscall # syscall 10 (exit)
[00400024] 3c041001  lui $4, 4097 [arr]           ; 8: la $a0,arr #a0放陣列
[00400028] 3c011001  lui $1, 4097                 ; 9: lw $a1,num #a1放長度5
[0040002c] 8c250014  lw $5, 20($1)
[00400030] 23bdffec  addi $29, $29, -20           ; 11: addi $sp,$sp,-20 #設定堆疊 排序
[00400034] afbf0010  sw $31, 16($29)              ; 12: sw $ra,16($sp) #ra s3 s2 s1 s0保存進堆疊
[00400038] afb3000c  sw $19, 12($29)              ; 13: sw $s3,12($sp)
[0040003c] afb20008  sw $18, 8($29)               ; 14: sw $s2,8($sp)
[00400040] afb10004  sw $17, 4($29)               ; 15: sw $s1,4($sp)
[00400044] afb00000  sw $16, 0($29)               ; 16: sw $s0,0($sp)
[00400048] 00049021  addu $18, $0, $4             ; 17: move $s2,$a0 #搬移參數
[0040004c] 00059821  addu $19, $0, $5             ; 18: move $s3,$a1
[00400050] 00008021  addu $16, $0, $0             ; 19: move $s0,$0
[00400054] 3402000a  ori $2, $0, 10               ; 21: li $v0, 10
[00400058] 0000000c  syscall                      ; 22: syscall
[0040005c] 0213402a  slt $8, $16, $19             ; 25: slt $t0,$s0,$s3 #判斷外部迴圈
[00400060] 11000011  beq $8, $0, 68 [exit1-0x00400060]
[00400064] 2211ffff  addi $17, $16, -1            ; 27: addi $s1,$s0,-1
[00400068] 2a280000  slti $8, $17, 0              ; 29: slti $t0,$s1,0 #內部迴圈
[0040006c] 1511000c  bne $8, $17, 48 [exit2-0x0040006c]
[00400070] 00114880  sll $9, $17, 2               ; 31: sll $t1,$s1,2
[00400074] 02495020  add $10, $18, $9             ; 32: add $t2,$s2,$t1
[00400078] 8d4b0000  lw $11, 0($10)               ; 33: lw $t3,0($t2)
[0040007c] 8d4c0004  lw $12, 4($10)               ; 34: lw $t4,4($t2)
[00400080] 018b402a  slt $8, $12, $11             ; 35: slt $t0,$t4,$t3
```

```
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.
QtSPIM is linked to the Qt library, which is distributed under the GNU Lesser General Public License version 3 and version 2.1.
```