# Pràctica 2

## Activitat 2.A

### Exercici 2.1

| # Y = Y^(1<<X)-1);  li $t1,1  # 0x00000001  sllv $t1,$t1,$s0  # 1<<X  subi $t1,$t1,1  # (1<<X)-1  xor $s1, $s1, $t1  # Y = Y^((1<<X)-1) |
| --- |

## Exercici 2.2

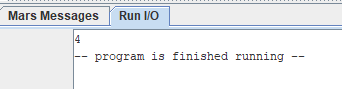
| .data  result: .word 0  num:    .byte '7'      .text      .globl main  main:      # Variables Principals      la $t0,result # &result      la $t1,num      lb $t1,0($t1)  # num        # Constants      li $t3,'a'      li $t4,'z'      # (A&&B)      blt $t1,$t3,oor  # if num<a goto or      bgt $t1,$t4,oor  # if num>z goto or      sw $t1,0($t0)  # result = num      j niam  # Final  oor:      # Constants      li $t3,'A'      li $t4,'Z'      # (C&&D)      blt $t1,$t3,else  # if num<A goto else      bgt $t1,$t4,else  # if num>Z goto else      sw $t1,0($t0)  # result = num      j niam  # Final  else:      # Constants      li $t3,'0'      li $t4,'9'      blt $t1,$t3,elsee  # if num<0 goto elsee      bgt $t1,$t4,elsee  # if num>9 goto elsee      subu $t1,$t1,$t3  # num-'0'      sw $t1,0($t0) # result = num-'0'      j niam  # Final  elsee:      li $t2,-1      sw $t2,0($t0)  # result = -1  niam:      jr $ra |
| --- |



## Activitat 2.C

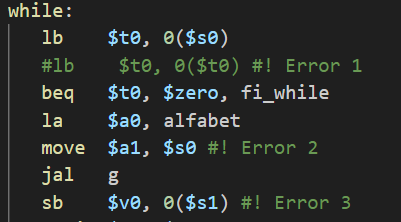
### Exercici 2.3

| .data  w:        .asciiz "8754830094826456674949263746929"  # Aqui hi ha 31 nums. + '\0'  resultat: .byte 0  # Digit ascii mes frequent      .text      .globl main  main:      addiu   $sp, $sp, -4      sw  $ra, 0($sp)      la  $a0, w      li  $a1, 31      jal moda      la  $s0, resultat      sb  $v0, 0($s0)      move    $a0, $v0      li  $v0, 11      syscall      lw  $ra, 0($sp)      addiu   $sp, $sp, 4      jr  $ra  nofares:      li  $t0, 0x12345678      move    $t1, $t0      move    $t2, $t0      move    $t3, $t0      move    $t4, $t0      move    $t5, $t0      move    $t6, $t0      move    $t7, $t0      move    $t8, $t0      move    $t9, $t0      move    $a0, $t0      move    $a1, $t0      move    $a2, $t0      move    $a3, $t0      jr  $ra  moda:      # Bloc d'Activacio "moda"      # ---      # NOTA: Els $sX els guardo en BA perque en algun punt de "moda" els necessitare per guardar-hi contingut de forma segura.      #       De forma segura significa que, DESPRES de fer una crida a una funcio, en sX HI HAURA el MATEIX VALOR QUE ABANS de fer la crida.      #       Aixo comportara en sobreescriure sX i pot ser que el pare (main) tingui un valor que volia preservar despres de fer la crida a "moda".      #       Es per aixo que el guardo en el BA tots els $sX que fare servir i abans d'acabar "moda" el recupero.      #       "El hijo le puede tocar los registros seguros al padre pero este no debe de enterarse :P"      #      # 0($sp)    | histo[10] | 40 B      # 40($sp)   |   s0      | 4 B      # 44($sp)   |   s1      | 4 B      # 48($sp)   |   s2      | 4 B      # 52($sp)   |   ra      | 4 B      # TOTAL: 56 B      # --- Configurar Bloc d'Activacio      addiu   $sp, $sp, -56      sw $s0,40($sp)      sw $s1,44($sp)      sw $s2,48($sp)      sw $ra,52($sp)        # --- Inicialitzar histo a 0      move    $t0,$zero  # k = 0      li  $t3,10  # limit iteracions  moda\_ficar\_zero:      sll     $t1,$t0,2  # i = k\*4      addu    $t2,$sp,$t1  # &histo[i]      sw  $zero,0($t2)  # histo[i] = 0      addiu   $t0,$t0,1  # k++      blt     $t0, $t3, moda\_ficar\_zero  # if k < 10 then goto moda\_ficar\_zero        # --- Trobar la moda      move $s2,$zero  # k = 0  ## IMPO: Guardo en registre segur perque es part del bucle i el puc perdre al fer la crida de funcio      li $t0,'0'  # max = '0'      move $s0,$a0  ## Com que fem crida a una funcio amb parametrers, el valor de $a0 es perdra al fer el "move" del contingut que tindra el primer parametre de la funcio      move $s1,$a1  ## Despres de la crida haig de continuar fent servir a0(vec[0]), a1(max iter) i si no els guardo els perderia.  moda\_trobar\_max:      move $a0,$sp  # $a0 = &histo[0]      addu $t2,$s0,$s2  # &vec[k] . Faig servir el segur perque abans he fet un move del parametre a0 de moda (que era vec[0]) a registre segur      lb $t2,0($t2)  # vec[k]      # Recorda que no existeix subiu o similar aixi que haurem de fer resta amb addiu i el valor de '0' en decimal      addiu $a1,$t2,-48  # $a1 = vec[k]-'0'      addiu $a2,$t0,-48 # $a2 = max-'0'      jal update  # Ara que estan els parametres carregats ja puc fer el salt      addiu $t0,$v0,'0'  # max = '0'+update(histo, vec[k]-'0', max-'0');      addiu $s2,$s2,1  # k++      blt $s2,$s1,moda\_trobar\_max  # if k<num goto moda\_trobar\_max      move $v0,$t0  # return max        # --- Restaurar els valors segurs i ra      lw $ra,52($sp)      lw $s2,48($sp)      lw $s1,44($sp)      lw $s0,40($sp)      addiu   $sp, $sp, 56      jr  $ra  # Tornar a "main"  update:      # Bloc d'Activacio "update"      # ---      # NOTA: Guardo els registres segurs que fare servir per guardar els parametres de la funcio "update" perque es multinivell.      #       "No se" com s'implementa "nofares". Pot ser que aquest, em toqui a0-a3 i jo els necessito mantindre.      #      # 0($sp)    | s0 | 4 B      # 4($sp)    | s1 | 4 B      # 8($sp)    | s2 | 4 B      # 12($sp)   | ra | 4 B      # TOTAL: 16 B      # --- Configurar Bloc d'Activacio      addiu $sp,$sp,-16      sw $s0,0($sp)  # Guardar els registres segurs del pare a la Pila per, abans d'acabar, restaurar.      sw $s1,4($sp)      sw $s2,8($sp)      sw $ra,12($sp)  # Aixo ho guardo perque es multinivell i al fer el salt perdere la direccio de retorn de "update"      move $s0, $a0  # \*h # Moure a registres segurs previament a cridar la funcio      move $s1, $a1  # i      move $s2, $a2  # imax        # --- Crida a funcio      jal nofares      # --- Incrementar comptador i retornar maxim      sll $t0,$s1,2  # t0 = i\*4      addu $t3,$s0,$t0  # &h[t0]      lw $t1,0($t3)  # h[t0]      addiu $t1,$t1,1  # h[t0]++      sw $t1,0($t3)      sll $t2,$s2,2  # $t2= imax\*4      addu $t3,$s0,$t2  # &h[imax]      lw $t2,0($t3)  # h[imax]      blt $t1, $t2, update\_else   # if h[i] < h[imax] then goto update\_else      move $v0,$s1  # return i      j etadpu  update\_else:      move $v0,$s2  # return imax        # --- Restaurar els valors segurs i ra  etadpu:      lw $ra,12($sp)      lw $s2,8($sp)      lw $s1,4($sp)      lw $s0,0($sp)      addiu $sp,$sp,16      jr $ra  # Tornar a "moda" |
| --- |



## Activitat 2.D (Opcional)

### Exercici 2.4



* **Error 1**: Al fer el primer lb ja obtens el valor del parametre a0 (s0) si tornes a fer-ho, estas agafant el valor de la posicio de memoria del valor de s0.
* **Error 2**: Estava guardant el registre segur que no toca. Estava passant s1 (a1 de codifica) pero necesita el s0 (a0 de codifica).
* **Error 3**: Estava guardant el valor de retorn al vector equivocat. S’ha de guardar a pfraseout i ho estava guardant a pfrasein.

