Sesión V. Cálculo de potencias

Escribe una subrutina (`potencia`) que, dados dos números x y n (con $x > 0$ y $n \geq 0$), calcule el valor de x elevado a la n-ésima potencia.

Escribe otra subrutina (`potenciaVector`) que, dados un vector A de m elementos ($m > 0$, $A[i] > 0$) y un número n (con $n \geq 0$), calcule el valor de cada elemento de A elevado a la n-ésima potencia. Para el cálculo esta subrutina llamará a la subrutina desarrollada en el punto anterior (anidamiento de subrutinas).

El programa principal se muestra a continuación. Observa que los parámetros que se le pasan a la subrutina son: $a0=A, $a1=m y $a2=n. No olvides incluir un segmento de datos con los valores asociados a las etiquetas A, m y n. Hay que salir del programa si se produce desbordamiento.

```
# Sesión V
#
# programa potencia

.text
.globl main

main:
   la $a0,A              # dirección de A en a0
   la $a1,m
   lw $a1,0($a1)         # contenido de m (nº els en A) en a1
   la $a2,n
   lw $a2,0($a2)         # contenido de n (potencia)

   jal potenciaVector

   slt $t0,$v0,$0
   bne $t0,$0,fin        # si no hubo desbordamiento salida normal

   la $a0,err
   addi $v0,$0,4
   syscall               # mensaje de desbordamiento

fin:                     # salimos del programa
   addi $v0,$0,10
   syscall


potencia:
# subrutina para elevar un número a una potencia
# entradas: $a0: base  $a1: potencia
# salida: $v0: $a0^$a1

   add $t0,$a1,$0        # iteramos: $t0=$a1 hasta 0
   addi $v0,$0,1         # resultado inicial: 1 = $a0^0
```

```
  for_potencia:          # for($t0=$a1; $t0!=0; $t0--)
    beq $t0,$0,fin_potencia
    mult $v0,$a0
    addi $t0,$t0,-1      # $t0- -

    mflo $v0
    slt $t1,$v0,$0       # si lo<0 => desbordamiento
    bne $t1,$0,desborda

    mfhi $t2
    beq $t2,$0,for_potencia # si hi!=0 => desbordamiento

  desborda:
    lui $v0,0x8000       # devolvemos valor especial para desbordamiento

  fin_potencia:
    jr $ra               # fin subrutina potencia


potenciaVector:
# subrutina para recorrer un array y llamar a la subrutina
#   potencia para cada elemento
# entradas: $a0: array  $a1: nº els array  $a2: potencia
# salida: ninguna, se dejan valores actualizados en memoria

  addi $sp,$sp,-24       # volcar a pila: $ra,$a0,$a1 y reg. salvados a usar
  sw $ra,0($sp)
  sw $a0,4($sp)
  sw $a1,8($sp)
  sw $s0,12($sp)
  sw $s1,16($sp)
  sw $s2,20($sp)

  # uso reg. salvados para asegurar que se preservan su valor
  # entre llamadas a otras subrutinas
  add $s0,$a0,$0         # dirección del elem del array a usar (ini: $a0)
  add $s1,$0,$0          # nº de elementos del array recorridos (ini: 0)
  add $s2,$a1,$0         # nº total de elementos del array (m en $s2)

  add $a1,$0,$a2         # exponente a $a1 para llamar a subrutina potencia

  for_potenciaVector:    # for($s1=0; $s1<$s2; $s1++)
    slt $t0,$s1,$s2
    beq $t0,$0,fin_potenciaVector
    lw $a0,0($s0)        # obtengo primer elemento del array
                         #  y se lo paso en $a0 a subrutina potencia
```

```
    jal potencia          # llamo a subrutina, para obtener $a0^$a1 en $v0

    slt $t0,$v0,$0
    bne $t0,$0,fin_potenciaVector # ha habido desbordamiento?

    sw $v0,0($s0)         # almaceno resultado en lugar de elemento original

    addi $s1,$s1,1        # actualizo nº elem recorridos: $s1++
    addi $s0,$s0,4        # paso a siguiente elemento en el array

    j for_potenciaVector # sigo iterando

  fin_potenciaVector:
    lw $ra,0($sp)         # restauro los valores guardados en la pila
    lw $a0,4($sp)
    lw $a1,8($sp)
    lw $s0,12($sp)
    lw $s1,16($sp)
    lw $s2,20($sp)
    addi $sp,$sp,24       # restauramos puntero de pila a su posición

    jr $ra                # fin subrutina potenciaVector


.data:
A: .word 1,2,3,4,5
m: .word 5
n: .word 3
err: .asciiz "Overflow!"
```