SUBRUTINAS - ARM

1. Escriba un programa en lenguaje de alto nivel que llame a una función `fact` que calcule el factorial de un número `n` usando un bucle. Traduzca el programa al lenguaje ensamblador del ARM.

RESPUESTA 1.

2. Escribe un programa para el ensamblador del ARM que llame a una subrutina, `swap(int *a, int *b)`, encargada de intercambiar el contenido de dos posiciones de memoria. La subrutina recibirá como parámetros de entrada las posiciones de memoria correspondiente a `a` y `b` y deberá preservar el contenido de todos los registros que se empleen para realizar la operación.

RESPUESTA 2.

3. Escribe un programa para el ensamblador del ARM que cuente el número de 0's de un vector de longitud arbitraria. Emplea para ello una subrutina llamada `cuenta0s` que reciba como parámetros de entrada toda la información necesaria para llevar a cabo la tarea.

RESPUESTA 3.

4. Implementar el algoritmo de ordenación de la burbuja o *bubble sort* en ensamblador. Este sencillo algoritmo ordena los elementos de un vector de menor a mayor por medio de un procedimiento muy sencillo: recorre repetidas veces el vector, intercambiado posiciones sucesivas si `V(i)>V(i+1)`, hasta que no se realiza ningún cambio. Se proporciona como ayuda el siguiente código de alto nivel:

   Nota: definir una constante `N` que almacene el número de elementos del vector `V`.

```
do
  swapped=0
  for i from 0 to N-2 do:
       if V[i] > V[i+1] then
            swap( V[i], V[i+1] )
            swapped = true
       end if
  end for
```

```
while swapped
```

RESPUESTA 4.

5. Supongamos que definimos que un número natural es "bonito" si es menor que cien mil y además su valor puede obtenerse como una suma de números naturales de la forma 1+2+3+4+5+…

Se pide:

1. Escribir un programa en lenguaje ensamblador del ARM tal que dado un número natural N decida si es o no bonito. El programa escribirá en la variable B un 1 si el número es bonito y un 0 en caso contrario.
2. Convertir el código anterior en una subrutina que reciba como entrada un número natural N y devuelva como salida un 1 si el número N es bonito y un 0 si no lo es. Escribir un programa en lenguaje ensamblador del ARM que llame a la subrutina, y tal que dado un vector A de M números naturales sea capaz de hallar cuántos números bonitos hay en el vector. El programa debe almacenar la cantidad de números bonitos hallada en la variable "cuenta_bonitos".

   Nota: Se debe respetar el convenio del ARM visto en clase para llamadas a subrutinas. Además, en ambos apartados se deben incluir las directivas para reservar memoria y declarar las secciones (.data, .bss y .text) correspondientes.

RESPUESTA 5.

6. Responde a las cuestiones suponiendo que el vector V está almacenado a partir de la dirección de memoria 0x0C000000, que el código se encuentra a continuación de los datos y que las pseudo-instrucciones ocupan el mismo espacio que las instrucciones.

```
# Codigo 1
.global start

.data
V: .word 12,21,13,14,5,9
N: .word 6

.bss
CuentaTotal: .space 4

.text
start:              ldr R0,=V
                        ldr R2,=N
                        ldr R1,[R2]
                        mov R2,#0
```

```
                        mov R3,#0
bucle:              cmp R2,R1
                        beq fin_bucle
                        ldr R4,[R0]
                        and R4,R4,#1
                        add R3,R3,R4
                        add R2,R2,#1
                        add R0,R0,#4
                        b bucle
fin_bucle:     ldr R1,=CuentaTotal
str R3, [R1]
b .

.end


#Código 2
.global start

.data
V: .word 12,21,13,14,5,9
N: .word 6

.bss
CuentaTotal: .space 4

.text
start:              mov sp,#0x0C200000
ldr R0,=V
ldr R2,=N
ldr R1,[R2]
bl Cuenta
ldr R2,=CuentaTotal
str R0,[R2]
b .

Cuenta:
PRÓLOGO_1
mov R4,#0
mov R5,#0
mov R6,R0
mov R7,R1
bucle:              cmp R4,R7
beq fin_bucle
ldr R0, [R6]
bl Comprobar
add R5,R5,R0
```

```
add R4,R4,#1
add R6,R6,#4
b bucle
fin_bucle:     mov R0,R5
EPÍLOGO_1
mov pc,lr

Comprobar:
PRÓLOGO_2
mov R4,#1
and R0,R0,R4
EPÍLOGO_2
mov pc,lr

.end
```

1. ¿En qué dirección de memoria del código 1 está almacenada la variable N? ¿Y la primera instrucción del bucle para el código 1? Razona las respuestas.
2. ¿Cuál es el contenido final de la variable de memoria CuentaTotal en el código 1? ¿Y de los registros R0, R1 y R2? Razona la respuesta.
3. Supongamos que queremos estructurar el código con subrutinas, y lo rescribimos en código 2. Completa el prólogo y epílogo de cada una de las dos subrutinas, explicando por qué incluyes cada instrucción.

RESPUESTA 6.

7. Dado un vector V de N componentes se dice que es Melchoriforme si posee al menos un elemento Rubio. Un elemento V[i] es Rubio si satisface la siguiente expresión:

$$\sum_{j=0}^{N-1} v[j] = 2 * v[i]$$

Se pide:

1. Una subrutina SumaVector(V, N) que sume los N elementos del vector V, respetando el convenio de llamadas a subrutinas visto en clase.
2. Un programa que dado un vector V y su dimensión N decida si es Melchoriforme, utilizando la subrutina SumaVector.

RESPUESTA 7.

8. Un vector V de N números naturales es noeliano si es una secuencia monótona creciente y sus elementos suman en total 45. Por ejemplo: 0-1-2-3-4-5-6-7-8-9 es noeliano porque 0123456789 y 1+2+3+4+5+6+7+8+9=45. También: 3-5-5-7-10-15 es noeliano, ya que 35571015 y 3+5+5+7+10+15=45. Se pide:
    1. Escribir una subrutina en ensamblador de ARM Sum45(A, N) que reciba la dirección de comienzo de un vector A como primer parámetro, el número N de elementos del vector como segundo parámetro y devuelva 1 si su suma es 45 y 0 en otro caso. La subrutina debe programarse de acuerdo con el estándar de llamadas a subrutinas que hemos estudiado en clase.
    2. Escribir un programa ARM que, utilizando la subrutina anterior, determine si un vector de entrada es noeliano o no.

RESPUESTA 8.