Sesión 9 Estructura de Computadores

Ejercicio: Incrementar un programa en Mips que calcule, de forma recursiva, el elemento n-ésimo de la serie de Fibonacci.

```
#Sesión 9. Fibonnaci con Recursividad

.data

cadena: .asciiz "Introduce el número a calcular de la serie de Fibonnaci: "
cadena1: .asciiz "Número: "

.text
.globl main
main: #----------------------------# La rutina main crea su marco de pila (de 32 bytes) en memoria.
# (0)
        subu $sp, $sp, 32 # Nuevo stack pointer (puntero de pila): $sp <-- $sp+32
        sw $ra, 20($sp) # M[$sp+20] <-- $ra. Guarda direcc. de vuelta
        sw $fp, 16($sp) # M[$sp+16] <-- $fp. Guarda $fp antiguo
        sw $s0, 12 ($sp) #Registro Auxiliar para guardar el contenido de fibonacci (n-1)
        addu $fp, $sp, 32 # Nuevo frame pointer (puntero de estructura): $fp <-- $sp+32 (donde estaba sp antes)

        li $v0, 4       #Imprime por pantalla la cadena
        la $a0, cadena
        syscall

        li $v0, 5       #Argumento por consola a calcular
        syscall

        move $a0, $v0
        jal fibo                # Llama a la funcion fibo, almacena en $ra dir. sig. instrucc.
        add $s2, $v0, $zero     # Resultado está en $s2.

        li $v0, 4               #Imprime por pantalla la cadena
        la $a0, cadena1
        syscall


        li $v0, 1               # Se escribe en consola
        add $a0, $s2, $zero
        syscall
        lw $ra, 20($sp)         # Restaura registros
        lw $fp, 16($sp)
        lw $s0, 12($sp)
        addu $sp, $sp, 32

        li $v0, 10
        syscall
#----------------------------------------# Función fibonnaci: cálculo de número
fibo:

        subu $sp, $sp, 32 # Crea marco de pila
        sw $ra, 20($sp) # M[$sp+20] <-- $ra. Guarda direcc. de vuelta
        sw $fp, 16($sp) # M[$sp+16] <-- $fp. Guarda $fp antiguo
        sw $s0, 12($sp) #Registro auxiliar
        addu $fp, $sp, 32 # Nuevo frame pointer (puntero de estructura):
                          # $fp <-- $sp+32 (donde estaba sp antes)
        sw $a0, 0($fp) # Guarda argumento $a0 en marco de pila (n)


        bgt $a0,1, casoN # si n>1, voy a la etiqueta de caso recursivo
        move $v0,$a0
        b fin           #Salto a etiqueta fin

casoN:
        sub $a0, $a0, 1  #Argumento n-1
        jal fibo         #fibo(n-1)
        move $s0,$v0     #guardo en en auxiliar fib (n-1)

        sub $a0,$a0,1   #Argumento n-2
        jal fibo        #fibo(n-2)
        add $v0,$v0,$s0 #en v0, se ejecuta la recursividad: v0 = fibo(n-1)+fibo(n-2)

fin:
        lw $a0, 0 ($fp) #Restuaro los registros del marco de pila de la función
        lw $ra, 20($sp)
        lw $fp, 16($sp)
        lw $s0, 12($sp)
        addu $sp, $sp, 32
        jr $ra # decrement/next in stack

```

Ejecución de Consola para comprobar el funcionamiento del código:

Serie de Fibonacci

Términos	**1**	**1**	**2**	**3**	**5**	**8**	**13**	**21**	**34**	**55**	**89**	**144**	**233**
Posición	1	2	3	4	5	6	7	8	9	10	11	12	13

Parámetro: 5 (caso Recursivo)

```
Introduce el número a calcular de la serie de Fibonnaci: 5
Número: 5
-- program is finished running --
```

Parámetro: 12 (caso Recursivo)

```
Introduce el número a calcular de la serie de Fibonnaci: 12
Número: 144
-- program is finished running --
```

Parámetro: 1 (caso Base)

```
Introduce el número a calcular de la serie de Fibonnaci: 1
Número: 1
-- program is finished running --
```

Parámetro: 0 (caso Base)

```
Introduce el número a calcular de la serie de Fibonnaci: 0
Número: 0
-- program is finished running --
```