Práctica de Laboratorio 1: Secuencia de Fibonacci en MIPS32

Nombre del Estudiante

Julio 2025

1. Implementación de recursividad en MIPS32

- **Mecanismo:** Llamadas anidadas mediante `jal` con gestión manual de contexto
- **Rol de la pila (`$sp`):**
  - Almacena dirección de retorno (`$ra`)
  - Preserva argumentos (ej. `$a0` con valor actual de `n`)
  - Guarda resultados intermedios
  - Maneja crecimiento dinámico: `addi $sp, $sp, -12` (reserva) y `addi $sp, $sp, 12` (liberación)

2. Riesgos de desbordamiento y mitigación

Riesgo	**Causa**	**Mitigación**
Stack overflow	Recursión profunda (`n` ¿1000)	Limitar `n` 40
Integer overflow	`fib(47)` ¿$2^{31} - 1$	Validar `n` 46
Heap overflow	Memoria dinámica con `n` grande	Liberar memoria con `sbrk` negativo

3. Comparación: Iterativo vs. Recursivo

Característica	**Iterativo**	**Recursivo**
Complejidad temporal	$O(n)$	$O(2^n)$
Uso memoria	Heap: $4(n+1)$ bytes	Pila: $12n$ bytes
Registros usados	Temporales (`$tX`)	Preservados (`$sX`)
Legibilidad	Más extenso	Similar a definición matemática

4. Diferencias con ejemplos académicos

- **Código real incluye:**
  - Validación de entrada (`n  0`)
  - Gestión de memoria dinámica (`sbrk`)
  - Mensajes de error específicos
  - Interfaz de usuario completa (prompts, formatos)
  - Visualización de secuencia completa (iterativo)
- **Ejemplos académicos:** Focalizados en algoritmo básico sin manejo de errores

5. Tutorial de Ejecución Paso a Paso en MARS

5.1. Preparación Inicial

1. Descargar e instalar MARS (versión 4.5 o superior)
2. Configurar:
   - `Settings → Memory Configuration → Compact, Data at Address 0`
   - `Settings → Delayed Branches → Disabled`

5.2. Ejecución del Fibonacci Iterativo (n=5)

1. **Cargar y ensamblar:**
   - Abrir `Fibonacci_Iterativo.asm`
   - Ensamblar (F3)
2. **Flujo de ejecución:**
   a) `main`: Muestra prompt y lee entrada (ingresar 5)
   b) Reserva de memoria:
      - `addi $t0, $s1, 1`: $t0 = 6 (n+1)
      - `sll $a0, $t0, 2`: $a0 = 24 bytes
      - `li $v0, 9`: syscall sbrk → $s2 = dirección base (ej: 0x10040000)

   c) `fib_array`:
      - `sw $zero, 0($s2)`: fib[0] = 0
      - `li $t0, 1; sw $t0, 4($s2)`: fib[1] = 1
      - Bucle (i=2 a 5):

   d) Salida:
      - Muestra: .El número Fibonacci F(5) es: 5”
      - Secuencia completa: ”0 1 1 2 3 5”

i	Dirección	Valor
2	0x10040008	1 (0+1)
3	0x1004000C	2 (1+1)
4	0x10040010	3 (1+2)
5	0x10040014	5 (2+3)

5.3. Ejecución del Fibonacci Recursivo (n=3)

1. **Cargar y ensamblar:** `Fibonacci_Recursivo.asm` (F3)
2. **Flujo de ejecución:**

   a) `main`: Lee n=3, llama a `fib_rec`

   b) **Pila durante ejecución:**

Llamada	`$ra`	`$a0`	Resultado
fib_rec(3)	0x00400034	3	?
fib_rec(2)	0x004000A8	2	?
fib_rec(1)	0x004000A8	1	1
fib_rec(0)	-	0	0

   c) **Desarrollo:**

   - `fib_rec(3)`: Guarda $ra=0x00400034, $a0=3. Llama a fib_rec(2).
   - `fib_rec(2)`: Guarda $ra=0x004000A8, $a0=2. Llama a fib_rec(1) y fib_rec(0).
   - `fib_rec(1)`: Retorna 1 → guardado en pila de fib_rec(2).
   - `fib_rec(0)`: Retorna 0.
   - `fib_rec(2)`: Calcula 1+0=1 → retorna 1.
   - `fib_rec(3)`: Recupera resultado fib(2)=1, calcula fib(1)=1 → 1+1=2 → retorna 2.

   d) `main`: Recibe 2, muestra "F(3) es: 2"

6. Justificación de enfoque

Iterativo recomendado cuando:

- $n > 30$ (evita stack overflow)
- Eficiencia crítica ($O(n)$ vs $O(2^n)$)
- Memoria limitada

Recursivo recomendado cuando:

- $n \leq 30$ y claridad ¿rendimiento
- Propósitos educativos

Conclusión: Para MIPS32 el enfoque iterativo es superior en estabilidad y rendimiento.

7. Análisis de resultados

- **Límites prácticos:**
  - Recursivo: $n = 40$ tarda ¿10s
  - Iterativo: $n = 10^6$ en ¡1s (teórico)
- **Consumo memoria:**
  - Recursivo: $\approx$ 12n bytes (pila)
  - Iterativo: $4(n + 1)$ bytes (heap)
- **Optimizaciones identificadas:**
  - Iterativo: Almacenar sólo últimos 2 valores (reduce memoria a O(1))
  - Recursivo: Memoization (caché de resultados)

Conclusiones

La implementación iterativa es preferible para aplicaciones reales en MIPS32 por su eficiencia y estabilidad, mientras la recursiva sirve como herramienta educativa para demostrar el manejo de pila en llamadas anidadas.