Nro. ord.	Apellido y nombre	L.U.	#hojas

SISTEMAS DIGITALES - **Parcial**

Primer Cuatrimestre 2024

Ej.1	Ej.2	Ej.3	Nota

Correctorx:

Aclaraciones

- Anote apellido, nombre, LU y numere *todas* las hojas entregadas, entregando los distintos ejercicios en hojas separadas.
- Cada ejercicio será calificado con una de las siguientes tres notas: Bien, Regular o Mal. La división de los ejercicios en incisos es meramente orientativa. Los ejercicios se calificarán globalmente.
- El parcial **no es a libro abierto** pero pueden utilizar la cartilla de referencia entregada por la materia.
- **Importante:** Justifique sus respuestas.
- Un resultado sin suficiente justificación equivale a un ejercicio no resuelto.
- El parcial se aprueba con al menos dos ejercicios Bien y uno Regular. Para obtener un Regular es necesario demostrar conocimientos sobre el tema del ejercicio. Para la promoción deben contar con al menos tres ejercicios bien y uno regular.

Ejercicio 1 Se cuenta con cuatro datos de un byte cada uno almacenados en el registro `s0` y queremos saber cuántos de esos datos son impares. Escriba un programa de ensamblador `RISC V` que realice esta operación y almacene el resultado en el registro `a0`.

Ejemplo:

Bits	31-24	23-16	15-8	7-0
s0	0x37	0xA2	0xF0	0x11

Con este dato el registro debería valer 2 ya que el segundo y tercer byte son impares.

Ejercicio 2 Implemente la función factorial en el lenguaje ensamblador `RISC V` de forma recursiva, respete la convención de llamada presentada en la materia, explique el uso que le dará a cada registro y cómo se asegura que sus valores se preservan antes y después de cada llamada a función.

$$fact(n) = \begin{cases} 1, & \text{si } n = 1 \\ n * fact(n-1), & \text{si } n > 1 \end{cases}$$

Guía de resolución (opcional):

- Escriba una versión de pseudocódigo.
- Transforme cada caso a su equivalente de operaciones atómicas (descomponga las operaciones lógicas, aritméticas y llamadas a función).
- Identifique los registros a emplear para cada dato.
- Si debe preservar algún registro para respetar la convención, indique qué mecanismo utilizará.
- Defina un flujo de ejecución tentativo.

Ejercicio 3 Un dispositivo de medición de monóxido de carbono en el aire realiza mediciones periódicas de la calidad del aire y almacena estos datos en su memoria principal. Queremos agregar lógica para detectar cuándo la calidad del aire se ve considerablemente degradada.

Se cuenta con un arreglo `mediciones` de datos de 16 bits **sin signo** empaquetados de forma contigua. El largo del arreglo (en half-words, o sea datos de 16 bits) se define en la constante `largo`.

Escriba un programa que cuente la cantidad de mediciones que se encuentran por sobre el valor `0x0F00`. Si la cantidad de valores que superan este límite es mayor a la mitad del largo debemos poner un 1 en el registro `a0`, en caso contrario debemos poner un 0.

Ejemplo:

Dirección	0x0000	0x0002	0x0004	0x0006
mediciones	0x1100	0x00F0	0xA200	0x1000

En este caso debemos poner un 1 en `a0` porque más de la mitad de los valores valen más que `0x0F00`.

Esqueleto de programa:

```
1 .data:
2 mediciones: .half 0x1100 0x00F0 0xA200 0x1000
3 largo: .byte 4
4
5 .text:
6 # Escribir el programa aca.
```

Ejercicio 4 ¿De qué tamaño son las direcciones de memoria de la arquitectura `RISC V` que vimos en la materia? ¿En cuánto debemos incrementar una dirección si queremos acceder a la próxima palabra? ¿Y al próximo byte?