# Universidad Simón Bolívar

Departamento de Computación y Tecnología de la Información

**CI3641 – Lenguajes de Programación I**

Septiembre–Diciembre 2025

# Examen 3

(20 puntos)

A continuación encontrará las instrucciones del examen 3 de CI3641. Sea lo más detallado y preciso posible en sus razonamientos y procedimientos.

**La primera parte del examen** es una parte teorica. Tienen que hacer una presentacion, en forma de video (puede ser .mp4), de max 30 min sobre los siguientes puntos del libro de la materia (Scott — Programming Languages Pragmatics):

- Capítulo 7 — Sección 7.1
- Capítulo 7 — Sección 7.2
- Capítulo 7 — Sección 7.4
- Capítulo 8 — Sección 8.1 a 8.2
- Capítulo 8 — Secciones 8.4 a 8.6

Esta parte teorica vale mucho, su valor es de 10 puntos. Su valor es debido a que deben leer, analizar y comprimir lo que aprendieron en solo 30 min. Deben leer con cuidado cada seccion y pensar que es lo mas importante de cada seccion, eso es lo que deben exponer.

Luego, vendria **la segunda parte del examen**. En esta parte deben hacer desarrollo escrito. En algunas preguntas, se usarán las constantes X, Y y Z. Estas constantes debe obtenerlas de los últimos tres números de su carné. Por ejemplo, si su carné es 09-40325, entonces X = 3, Y = 2 y Z = 5.

En aquellas preguntas donde se le pida decir qué imprime un programa, incluya los pasos rele- vantes de la ejecución del mismo con los cuales usted pudo alcanzar su conclusión.

En aquellas preguntas donde se le pida implementar un programa, mantenga su código en un repositorio git remoto (preferiblemente Github) y coloque un enlace al mismo en lugar de su respuesta. Todo su código debe ser legible y estar debidamente documentado.

La entrega se realizará por correo electrónico a `flovera@usb.ve` y deben entregar la presentacion en video ademas del pdf y del repositorio.

**Parte escrita**

**1.** Tomando en cuenta las definiciones de X, Y y Z planteadas en los párrafos de introducción del examen, considere las siguientes definiciones de variables:

$$\begin{aligned} L_1 &= \text{mín}(X, Y) & U_1 &= \text{máx}(X, Y) + 1 \\ L_2 &= \text{mín}(X, Z) & U_2 &= \text{máx}(X, Z) + 1 \\ L_3 &= \text{mín}(Y, Z) & U_3 &= \text{máx}(Y, Z) + 1 \end{aligned}$$

$$I = \left\lfloor \frac{L_1 + U_1}{2} \right\rfloor \qquad J = \left\lfloor \frac{L_2 + U_2}{2} \right\rfloor \qquad K = \left\lfloor \frac{L_3 + U_3}{2} \right\rfloor$$

Considere también la siguiente declaración:

$$M : \text{array } [L_1..U_1] \text{ of array } [L_2..U_2] \text{ of array } [L_3..U_3] \text{ of } T$$

Suponiendo que $M$ inicia en la dirección cero (0) y que el tamaño del tipo $T$ es cuatro (4), se desea que calcule:

(a) La dirección de $M[I][J][K]$ si las matrices se guardan en *row–major*.

(b) La dirección de $M[I][J][K]$ si las matrices se guardan en *column–major*.

**2.** Se desea que modele e implemente, en el lenguaje de su elección, un programa que simule un manejador de tipos de datos. Este programa debe cumplir con las siguientes características:

(a) Debe saber manejar tipos atómicos, registros (`struct`) y registros variantes (`union`).

(b) Una vez iniciado el programa, pedirá repetidamente al usuario una acción para proceder. Tal acción puede ser:

I. `ATOMICO <nombre><representación><alineación>`

Define un nuevo tipo atómico de nombre `<nombre>`, cuya representación ocupa `<representación>` bytes y debe estar alineado a `<alineación>` bytes.

Ejemplo: `ATOMICO char 1 2` y `ATOMICO int 4 4`

II. `STRUCT <nombre>[<tipo>]`

Define un nuevo registro de nombre `<nombre>`. La definición de los campos del registro viene dada por la lista en `[<tipo>]`. Nótese que los campos no tendrán nombres, sino que serán representados únicamente por el tipo que tienen.

Ejemplo: `STRUCT foo char int`

III. `UNION <nombre>[<tipo>]`

Define un nuevo registro variante de nombre `<nombre>`. La definición de los campos del registro variante viene dada por la lista en `[<tipo>]`.

Ejemplo: `UNION bar int foo int`

IV. `DESCRIBIR <nombre>`

Debe dar la información correspondiente al tipo con nombre `<nombre>`. Esta información debe incluir tamaño, alineación y cantidad de bytes desperdiciados para el tipo, bajo distintas estrategias:

- Sin empaquetar.
- Empaquetado.
- Reordenando los campos de manera óptima.

V. `SALIR`

Debe salir del simulador.

El programa deberá pedir la siguiente acción al usuario después de cada ejecución. Además, debe incluir pruebas unitarias con cobertura mayor al 80 %.