



## PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL

### FORMATO GUÍA DE APRENDIZAJE

Se informa a los aprendices que deben realizar los ejercicios solicitados utilizando el lenguaje de programación **Java** dentro de la herramienta **Apache NetBeans**. Como evidencia esperada, cada aprendiz deberá crear un proyecto por ejercicio, verificar que compile correctamente, incluir comentarios básicos y subir todos los proyectos al repositorio asignado en **Git**.

El enlace para la entrega y verificación del trabajo compartir enlace repositorio **Git**:  
<https://github.com/cjsarasty456/examen-ficha-3145556>

#### **Crear carpeta:**

**Con su primer nombre y primer apellido, ejemplo  
PedroPerez, dentro de esta carpeta subir el trabajo, solo cuando el instructor autorice.**

**La fecha y hora límite de entrega**

**19/11/2025**

**Duración: 4 horas.**

**Hora cargue 5:10-5:525 , por fuera horario se toma como nulo.**

Es obligatorio mantener una estructura organizada dentro del repositorio y cumplir con los lineamientos establecidos en clase.

#### **1. Ahorcado básico (juego de texto)**

Crea un programa que seleccione una palabra secreta al azar y permita al usuario adivinarla letra por letra. Finaliza cuando adivine toda la palabra o se quede sin intentos.

Conceptos: cadenas, bucles, condicionales, control de estados.

#### **2. Calculadora de edad exacta**



Pide al usuario su fecha de nacimiento (día, mes, año) y calcula su edad actual exacta (años, meses y días).

Estructuras: manejo de fechas, condicionales, operaciones aritméticas.

### 3. Conversor de grados Celsius a Fahrenheit

**Descripción:**

Pide al usuario una temperatura en Celsius y conviértela a Fahrenheit usando:

$$F = C \times \frac{9}{5} + 32$$

**Puntos para practicar:**

Manejo de variables

Operaciones matemáticas simples

### 4. Ordenar por frecuencia

Dado un arreglo, ordénalo por **frecuencia de aparición** (de mayor a menor).

Si dos números tienen la misma frecuencia, ordénalos de menor a mayor.

**Entrada:**

[4,4,1,2,2,2,3]

**Salida:**

[2,2,2,4,4,3,1]

**Conceptos:**

Ordenamientos compuestos

Construcción de nuevo arreglo

### 5. Rotación de matriz (90°)

**Descripción:**

Escribe una función que reciba una matriz NxN y retorne su versión rotada 90° a la derecha.

Datos ejemplo:

**Entrada:**

[1,2,3],

[4,5,6],



[7,8,9]

**Salida:**

[7,4,1],

[8,5,2],

[9,6,3]

## 6. Suma de matrices

La **suma de matrices** consiste en sumar elemento a elemento dos matrices que tienen **la misma dimensión** (mismo número de filas y columnas).

Si A y B son matrices de tamaño  $m \times n$ , entonces:

$C = A + B$ , donde

$$C[i][j] = A[i][j] + B[i][j]$$

### Ejemplo 1: Suma de matrices 2x2

A =

$$\begin{pmatrix} 1 & 3 \\ 2 & 5 \end{pmatrix}$$

B =

$$\begin{pmatrix} 4 & 2 \\ 6 & 1 \end{pmatrix}$$

C = A + B =

$$7. \begin{pmatrix} 1+4 & 3+2 \\ 2+6 & 5+1 \end{pmatrix} = \begin{pmatrix} 5 & 5 \\ 8 & 6 \end{pmatrix}$$

Mostrar al usuario la matriz c resultante.

## 8. Clase Estudiante

Crear una clase **Estudiante** con atributos privados:

nombre

nota1



nota2

Métodos:

Constructor sin parámetros

Constructor con parámetros.

Getter y Setter.

**promedio()** → retornar el promedio

**aprobo()** → retornar “Aprobó” si el promedio  $\geq 3.0$ , de lo contrario “Reprobó”

**Tarea:**

Crear un estudiante y mostrar su promedio y si aprobó.

## 9. Descripción del Ejercicio: Piedra, Papel o Tijera

El aprendiz debe desarrollar un programa en Java que simule el clásico juego “**Piedra, Papel o Tijera**”. El programa debe permitir que el usuario seleccione una de las tres opciones posibles y, de manera aleatoria, el sistema debe generar la opción de la máquina. Luego, el programa debe comparar ambas elecciones y determinar el resultado de la partida, indicando si el jugador **gana, pierde o empata**.

- El ejercicio requiere que el aprendiz implemente:
- Lectura de datos desde teclado.
- Generación de valores aleatorios.
- Uso de condicionales para determinar el ganador.
- Un método adicional que permita convertir los valores numéricos en nombres (piedra, papel o tijera).
- Buena organización del código, siguiendo principios básicos de programación.
- El propósito del ejercicio es reforzar la lógica de decisión, el manejo de entradas del usuario, el uso de funciones y la estructura general de un programa interactivo en Java

## Criterios Requeridos para la Actividad

1. **Lenguaje y herramienta obligatorios**
  - Los ejercicios deben realizarse únicamente en **Java**.
  - Deben ser desarrollados en **Apache NetBeans** sin excepciones.
2. **Organización de los proyectos**
  - Crear **un proyecto independiente por cada ejercicio**.
  - Cada proyecto debe tener su **clase principal** correctamente definida.
  - Estructurar paquetes y clases de manera ordenada.
3. **Funcionamiento del programa**



- Todos los proyectos deben **compilar sin errores**.
  - Los ejercicios deben ejecutarse y cumplir exactamente con lo solicitado.
4. **Calidad del código**
- Incluir **comentarios básicos** que expliquen partes clave del código.
  - Utilizar nombres claros y significativos para clases, métodos y variables.
  - Aplicar las buenas prácticas vistas en clase.
5. **Evidencia en Git**
- Subir todos los proyectos al repositorio asignado.
  - Mantener el repositorio ordenado, actualizado y con los cambios debidamente registrados.
  - Verificar que el enlace esté visible y accesible para el instructor.
6. **Fecha y hora de entrega**
- La entrega se debe realizar antes del

### **Criterios de Cancelación de la Actividad**

La actividad será **anulada** o **no evaluada** si ocurre alguna de las siguientes situaciones:

1. **Uso de herramientas no autorizadas**
  - Está **prohibido** utilizar herramientas adicionales para generar el código, incluyendo:
    - Editores externos distintos a Apache NetBeans
    - Sistemas automatizados de generación de código
    - *Plugins* o extensiones que completen código automáticamente
2. **Uso de Inteligencia Artificial (IA)**
  - Queda totalmente **prohibido** el uso de herramientas de IA para:
    - Generar código
    - Resolver ejercicios
    - Recibir soluciones o explicaciones externas
  - Cualquier evidencia de uso de IA será causa inmediata de **cancelación**.
3. **Colaboración indebida entre compañeros**
  - No se permite:
    - Intercambio de código
    - Realización conjunta de ejercicios
    - Copias parciales o totales
  - Si dos o más trabajos presentan similitudes notorias, todos serán anulados.
4. **No cumplir los criterios de organización, evidencia o funcionamiento**
  - Proyectos incompletos, sin compilar o sin subir al repositorio
  - Ausencia de estructura adecuada
  - Código no desarrollado por el aprendiz



## CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
<b>Autor (es)</b>	<b>Carlos Julio Cadena</b>	<b>Instructor</b>	<b>CIES</b>	<b>10/11/2025</b>