UNIVERSIDAD RURAL DE GUATEMALA

FACULTAD DE INGENIERÍA

INGENIERÍA EN SISTEMAS CON ESPECIALIZACIÓN EN INTELIGENCIA ARTIFICIAL

Manual de Laboratorio

ALGORITMOS

Código FE729

Cuatro sesiones presenciales de cuatro horas

ELABORADO POR

Ing. Mynor A. Cifuentes

UNIVERSIDAD RURAL DE GUATEMALA

Segundo Semestre 2026 • Laboratorios presenciales

UNIVERSIDAD RURAL DE GUATEMALA

Manual de Laboratorio de Algoritmos

Ingeniería en Sistemas con especialización en Inteligencia Artificial

UNIVERSIDAD	Universidad Rural de Guatemala
FACULTAD	Facultad de Ingeniería
CARRERA	Ingeniería en Sistemas con especialización en Inteligencia Artificial
ASIGNATURA	Algoritmos (código FE729)
CICLO	Segundo Semestre 2026
ELABORADO POR	Ing. Mynor A. Cifuentes
VERSIÓN	1.0 — 2026

Sobre este documento

Material de apoyo para las sesiones presenciales de laboratorio de la asignatura de Algoritmos, alineado con el programa oficial y con el calendario de laboratorios presenciales de la Universidad Rural de Guatemala.

El estudiante debe presentarse a cada sesión habiendo leído con anterioridad la práctica correspondiente y con los materiales solicitados. La hoja de trabajo forma parte de la evaluación del curso.

© 2026 Universidad Rural de Guatemala. Documento de uso académico interno. La reproducción total o parcial de este material debe citar la fuente.

Tecnología, Ciencia y Conciencia

Índice general

Introducción 1
- Propósitos de aplicación . . . 1
- Objetivos . . . 2
- Competencias que desarrolla la asignatura . . . 2
- Campo de aplicación . . . 3

Planificación del laboratorio 4
- Distribución de sesiones . . . 4
- Correspondencia con el programa de la asignatura . . . 5
- Estructura de una jornada de laboratorio . . . 6
- Normas de trabajo en el laboratorio . . . 6
- Evaluación de las prácticas . . . 7

Materiales y herramientas 8
- Equipo básico . . . 8
- Software requerido . . . 8
- Materiales por práctica . . . 9

01 Del problema al algoritmo: análisis, pseudocódigo y expresiones 10
- Introducción . . . 10
- Fundamento teórico . . . 11
- Procedimiento . . . 12

02 Decisiones, repeticiones y una lista de notas 17
- Introducción . . . 17
- Fundamento teórico . . . 18
- Procedimiento . . . 18

03 Dividir el trabajo: procedimientos y funciones 25
- Introducción . . . 25
- Fundamento teórico . . . 26
- Procedimiento . . . 26

04 Proyecto guiado: un menú de tres notas 31
- Introducción . . . 31
- Fundamento teórico . . . 32
- Procedimiento . . . 32

Anexo A. Convenciones de pseudocódigo 37
Estructura general de un algoritmo 37
Palabras reservadas 38
Del pseudocódigo general a PSeInt 38
Convenciones de escritura 39
Simbología de diagramas de flujo 39

Anexo B. Evaluación en línea (Socrative) 40
Cómo se evalúa cada práctica 40
Indicaciones para el estudiante 40

Referencias bibliográficas 41

Introducción

El aprendizaje de la programación no comienza frente a un compilador: comienza cuando el estudiante es capaz de entender un problema, descomponerlo y describir con precisión la secuencia de pasos que conduce a su solución. Esa capacidad es, justamente, lo que se denomina **pensamiento algorítmico**, y constituye el núcleo de la presente asignatura.

Un algoritmo es una secuencia finita, ordenada y no ambigua de instrucciones que resuelve un problema determinado. Antes de escribir una sola línea de código en un lenguaje de programación, el ingeniero debe ser capaz de expresar la solución en un lenguaje intermedio, independiente de la tecnología, que permita razonar sobre su corrección y su eficiencia. El **pseudocódigo** y los **diagramas de flujo** cumplen esa función: son las herramientas con las que se diseña antes de construir.

Este manual reúne una serie de prácticas diseñadas para guiar al estudiante desde el análisis de un problema y la redacción de su primer algoritmo en pseudocódigo, hasta la construcción de soluciones modulares que manipulan arreglos y se organizan en procedimientos y funciones. El alcance coincide con los contenidos del programa oficial **hasta la semana 12** — segundo examen parcial —: unidades 1 a 6. Cada práctica integra el fundamento conceptual, un procedimiento guiado paso a paso y, al cierre de cada sesión, una evaluación breve en línea (Socrative) que combina la teoría de la clase magistral con lo realizado en el laboratorio.

A diferencia de un laboratorio tradicional de programación, el énfasis de estas prácticas no está en memorizar la sintaxis de un lenguaje, sino en desarrollar el criterio para elegir la estructura de datos y la estructura de control adecuadas, en verificar que la solución efectivamente funciona para todos los casos, y en documentar el razonamiento de manera que otra persona pueda comprenderlo y mantenerlo.

Propósitos de aplicación

- Facilitar el aprendizaje significativo de los principios fundamentales del diseño de algoritmos mediante actividades prácticas y progresivas.
- Desarrollar la habilidad de analizar un problema para identificar con precisión sus entradas, procesos, salidas y restricciones antes de proponer una solución.
- Promover el uso de una notación de pseudocódigo clara, uniforme y estandarizada como herramienta de diseño previa a la codificación.
- Integrar herramientas computacionales que permitan ejecutar, probar y depurar algoritmos, de modo que el estudiante verifique sus soluciones de forma empírica y no únicamente teórica.
- Fomentar el razonamiento lógico, la precisión conceptual y la capacidad de resolver problemas descomponiéndolos en subproblemas más simples.
- Preparar al estudiante para cursos posteriores de programación, estructuras de datos, bases de datos e

inteligencia artificial, en los cuales el diseño algorítmico es un requisito previo.

Objetivos

- Aplicar las etapas de la resolución de problemas: planteamiento, análisis, diseño de la solución, prueba, depuración y documentación.
- Construir algoritmos en pseudocódigo utilizando correctamente el encabezado, la sección de declaraciones, las constantes, las variables y los tipos de datos.
- Evaluar expresiones aritméticas y lógicas respetando la precedencia de operadores, e interpretar el comportamiento de los operadores `DIV` y `MOD`.
- Seleccionar y aplicar la estructura de control adecuada (selectiva o repetitiva) según la naturaleza del problema planteado.
- Utilizar arreglos unidimensionales y multidimensionales para almacenar y procesar conjuntos de datos relacionados.
- Diseñar soluciones modulares mediante procedimientos y funciones, distinguiendo el ámbito de validez de las variables y el paso de parámetros por valor y por referencia.
- Integrar los contenidos de las unidades 1 a 6 en un programa corto con menú, arreglo y módulos ya escritos.
- Documentar los algoritmos desarrollados de forma interna y externa, de manera que resulten comprensibles y mantenibles por terceros.

Competencias que desarrolla la asignatura

Las prácticas de este manual contribuyen directamente al desarrollo de las competencias establecidas en el programa oficial de la asignatura.

No.	Competencia
1	Analiza problemas computacionales para identificar sus requisitos, entradas, salidas y restricciones antes de proponer una solución.
2	Diseña algoritmos eficientes utilizando estructuras de control, técnicas de diseño y buenas prácticas de programación.
3	Representa algoritmos mediante pseudocódigo y diagramas de flujo, aplicando una notación clara y estandarizada.
4	Evalúa la eficiencia de los algoritmos considerando su complejidad temporal y espacial para seleccionar la solución más adecuada.
5	Implementa algoritmos en un lenguaje de programación, verificando su correcto funcionamiento mediante pruebas y depuración.
6	Aplica el pensamiento lógico y computacional para resolver problemas de manera sistemática, creativa y eficiente en diferentes contextos.

Campo de aplicación

El presente manual está dirigido a estudiantes que tengan asignado el curso de **Algoritmos (FE729)** de la carrera de Ingeniería en Sistemas con especialización en Inteligencia Artificial de la Universidad Rural de Guatemala.

Las prácticas aquí descritas han sido diseñadas específicamente para el formato de **laboratorio presencial intensivo** establecido en el calendario institucional: cuatro días consecutivos, de lunes a jueves, con una jornada de cuatro horas diarias. Por esta razón, los contenidos del programa de la asignatura se han agrupado en cuatro prácticas unificadas, cada una de las cuales corresponde exactamente a una sesión de laboratorio.

NOTA

Este laboratorio cubre las unidades 1 a 6 del programa oficial, que corresponden a las semanas 1 a 11. La semana 12 es el segundo examen parcial. Las unidades 7, 8 y 9 — cadenas, registros y archivos — se imparten en las semanas 13 a 16 y quedan **fuera del alcance** de estas prácticas. La distribución detallada se presenta en la sección *Planificación del laboratorio*.

Planificación del laboratorio

El calendario de laboratorios presenciales de Ingeniería en Sistemas asigna a la asignatura de Algoritmos **cuatro sesiones de cuatro horas**, programadas de lunes a jueves en el horario de **13:00 a 17:00 horas**. El laboratorio se alinea con el programa oficial **hasta la semana 12** (segundo examen parcial). Por esa razón se cubren únicamente las unidades 1 a 6, unificadas en cuatro prácticas integradoras.

NOTA

Cada práctica agrupa una o más unidades de las semanas 1 a 11. Esta unificación no reduce esos contenidos: los reorganiza de modo que las unidades relacionadas se trabajen de forma continua dentro de una misma jornada. Las unidades 7 a 9 — cadenas, registros y archivos — corresponden a las semanas 13 a 16 y no forman parte de este laboratorio.

Distribución de sesiones

Ses.	Tema central	Objetivo	Contenidos
1	Del problema al algoritmo: análisis, pseudocódigo y expresiones	Analizar un problema y expresar su solución en pseudocódigo correcto, evaluando expresiones aritméticas y lógicas	■ Etapas de la resolución de problemas ■ Planteamiento y análisis básico ■ Validación, prueba y depuración ■ Documentación interna y externa ■ Encabezado y declaraciones ■ Constantes, variables y tipos de datos ■ Expresiones aritméticas, `DIV` y `MOD` ■ Expresiones lógicas y precedencia ■ **Práctica 1**
2	Estructuras de control y arreglos	Decidir con `Si`, repetir con ciclos y guardar cinco notas en un arreglo	■ Estructura selectiva simple y doble ■ Selectiva anidada y múltiple ■ Ciclo Mientras y ciclo Repetir ■ Ciclo Para ■ Un arreglo de cinco notas ■ **Práctica 2**

Ses.	Tema central	Objetivo	Contenidos
3	Diseño descendente y modularización	Separar un problema en procedimientos y funciones, y observar el paso de parámetros	■ Esquema de módulos ■ Procedimientos y sus llamadas ■ Funciones que devuelven un valor ■ Parámetros por valor y por referencia ■ **Práctica 3**
4	Proyecto guiado de las unidades 1 a 6	Juntar menú, arreglo y módulos en un programa corto ya escrito	■ Análisis breve del menú ■ Tres notas en un arreglo ■ Procedimientos de carga y listado ■ Promedio y conteo de aprobados ■ **Práctica 4**

Correspondencia con el programa de la asignatura

La siguiente tabla permite verificar que las prácticas cubren el programa oficial **hasta la semana 12**.

Unidad	Semanas	Contenido del programa oficial	Se trabaja en
1	1	Etapas de la resolución de problemas	Práctica 1
2	2	Elementos básicos del pseudocódigo	Práctica 1
3	3	Expresiones	Práctica 1
4	4 y 5	Estructuras básicas (selectivas y repetitivas)	Práctica 2
—	6	Primer examen parcial	—
5	7 y 8	Arreglos	Práctica 2
6	9 a 11	Diseño descendente y modularización	Prácticas 3 y 4
—	12	Segundo examen parcial	—
7 a 9	13 a 16	Cadenas, registros y archivos	Fuera de alcance

ADVERTENCIA

Las unidades 7, 8 y 9 se imparten después del segundo parcial y no se desarrollan en este laboratorio. El proyecto integrador de la práctica 4 reúne únicamente los contenidos de las unidades 1 a 6.

Estructura de una jornada de laboratorio

Cada sesión de cuatro horas se organiza con la siguiente distribución de tiempo. Los períodos indicados son de referencia y el instructor puede ajustarlos según el avance del grupo.

Tiempo	Actividad	Descripción
15 min	Apertura	Presentación de la práctica, objetivos y verificación del entorno de trabajo.
185 min	Desarrollo	Ejecución guiada de las partes **A** a **D** del procedimiento, con asesoría permanente del instructor.
25 min	Evaluación	Evaluación en línea en **Socrative**: teoría y lo realizado durante la práctica.
15 min	Cierre	Resolución de dudas, respaldo de archivos y cierre de la sesión.
240 min	**Total**	**Cuatro horas efectivas**

RECOMENDACIÓN

Las partes del procedimiento son acumulativas: el trabajo de la parte **A** se reutiliza en las siguientes. Guarde cada archivo en una carpeta por práctica. No se entrega informe: la calificación se obtiene en el laboratorio y en la evaluación de Socrative.

Normas de trabajo en el laboratorio

NORMAS DE SEGURIDAD

1. Presentarse puntualmente. El laboratorio inicia a las 13:00 horas y no se repone el tiempo perdido por impuntualidad.
2. Leer la práctica correspondiente *antes* de la sesión. El tiempo de laboratorio se destina a construir y probar, no a leer por primera vez el enunciado.
3. Traer el equipo y el software solicitados, previamente instalados y verificados.
4. No instalar software ajeno a la práctica en los equipos del laboratorio ni modificar su configuración.
5. Guardar el trabajo de forma periódica y respaldarlo en medio propio antes de finalizar la sesión. Los equipos del laboratorio pueden ser restablecidos entre jornadas.
6. Mantener el orden y la limpieza del área de trabajo. Al finalizar, cerrar las aplicaciones y dejar el equipo apagado.
7. El trabajo puede realizarse individualmente o en parejas, según indique el instructor. La evaluación en Socrative es individual.

8. Las soluciones copiadas total o parcialmente de un compañero o de una fuente externa sin la cita correspondiente se califican con cero.

Evaluación de las prácticas

Cada práctica se califica sobre 100 puntos con los criterios siguientes. La suma de las cuatro prácticas constituye la nota de laboratorio, que se reporta dentro del rubro *Otras evaluaciones* del programa de la asignatura.

Criterio	Puntos	Descripción
Trabajo en el laboratorio	60	El estudiante sigue el procedimiento, completa las tablas y logra que los algoritmos se ejecuten.
Evaluación en línea	40	Evaluación individual en Socrative al finalizar la sesión. Combina preguntas de teoría con lo realizado en la práctica.
Total	**100**	

NOTA

No se entrega informe ni hoja de trabajo escrita. Al cierre de cada práctica queda un espacio reservado para el código de sala de Socrative. El detalle aparece en el anexo B.

Materiales y herramientas

A diferencia de un laboratorio de electrónica, el laboratorio de Algoritmos no requiere componentes ni instrumentos de medición. Sus herramientas son el razonamiento, el papel y un entorno de ejecución que permita comprobar que los algoritmos diseñados funcionan correctamente.

Equipo básico

MATERIALES Y EQUIPO

- **1 computadora** (portátil o de escritorio) por estudiante o por pareja de trabajo, según indique el instructor.
- **1 cuaderno cuadriculado**, lápiz y borrador. El diseño de diagramas de flujo y el trazado de tablas de seguimiento se realizan a mano antes de pasar al equipo.
- **1 memoria USB** o acceso a un servicio de almacenamiento en la nube, para respaldar y entregar el trabajo.

Software requerido

Todo el software solicitado es de distribución libre o gratuita. Debe estar **instalado y verificado antes de la primera sesión**; el tiempo de laboratorio no se destina a instalaciones.

Herramienta	Uso	Observaciones
PSeInt	Intérprete de pseudocódigo	Herramienta principal del laboratorio. Permite escribir, ejecutar y depurar pseudocódigo en español, y generar automáticamente el diagrama de flujo correspondiente. Distribución libre.
Navegador web	Evaluación	Se usa al final de cada sesión para ingresar a **Socrative**. Verifique que la computadora tenga conexión a internet.
draw.io o similar	Diagramas	Opcional. Sirve para dibujar un diagrama de flujo si el instructor lo pide.

NOTA

PSeInt es la herramienta de referencia de este manual porque su sintaxis coincide con la notación de pseudocódigo descrita en el anexo A y porque muestra los errores en español, lo que facilita la etapa de depuración. Si el instructor autoriza otra herramienta equivalente, el estudiante debe documentar cuál utilizó.

Materiales por práctica

Práctica	Materiales específicos
1	Computadora con PSeInt, cuaderno cuadriculado y acceso a Socrative.
2	Computadora con PSeInt, cuaderno para tablas de seguimiento y acceso a Socrative.
3	Computadora con PSeInt, cuaderno para el esquema de módulos y acceso a Socrative.
4	Computadora con PSeInt, memoria USB para respaldar el proyecto y acceso a Socrative.

RECOMENDACIÓN

Cree desde la primera sesión una carpeta con la estructura `Algoritmos_FE729/Practica_01`, `.../Practica_02`, y así sucesivamente. Guarde cada parte del procedimiento en un archivo independiente y nombre los archivos de forma descriptiva, por ejemplo `P01_ParteC_expresiones.psc`. Esta organización es parte del criterio de *trabajo en el laboratorio* de la evaluación.

01 PRÁCTICA
NO. 01

Del problema al algoritmo: análisis, pseudocódigo y expresiones

SESIÓN	DURACIÓN	UNIDADES	HERRAMIENTA
1 de 4	4 horas	1, 2 y 3	PSeInt

Introducción

Hoy se escribe el primer algoritmo. No hace falta inventarlo desde cero: el manual muestra cada paso y el estudiante lo copia, lo ejecuta y observa qué ocurre.

Un algoritmo es una lista de instrucciones, en orden, que resuelve un problema. Antes de escribirlo se responde a tres preguntas sencillas: **qué datos entran**, **qué se calcula** y **qué se muestra**. Eso se llama análisis **EPS** (Entrada-Proceso-Salida).

Después se escribe el algoritmo en **pseudocódigo**, con PSeInt. El pseudocódigo se parece al español y permite probar la idea sin un lenguaje de programación. Dentro del algoritmo se usan **expresiones**: sumas, restas, comparaciones, el cociente entero (`DIV`) y el residuo (`MOD`).

La clase magistral se lleva en línea. En el laboratorio el ritmo es pausado: se escribe, se ejecuta y se anota el resultado. Al final hay una evaluación breve en Socrative.

OBJETIVOS

- Completar una tabla de entradas, proceso y salidas para un problema corto.
- Distinguir una constante de una variable.
- Copiar, guardar y ejecutar un algoritmo en PSeInt.
- Calcular a mano y en PSeInt expresiones con `+`, `*`, `DIV` y `MOD`.
- Evaluar expresiones con `Y`, `O` y `NO`.

MATERIALES Y EQUIPO

- 1 computadora con **PSeInt** instalado y acceso a internet (Socrative).
- 1 cuaderno, lápiz y borrador.

- Memoria USB o carpeta en la nube para guardar el trabajo.

PLANIFICACIÓN DE LA SESIÓN

Parte	Actividad	Tiempo
—	Apertura y apertura de PSeInt	15 min
A	Análisis del problema (tabla EPS)	40 min
B	Escribir y ejecutar el primer algoritmo	50 min
C	Expresiones aritméticas, `DIV` y `MOD`	45 min
D	Expresiones lógicas y condición de aprobado	40 min
—	Evaluación en Socrative y cierre	25 min
	Total	**215 min + 25 min**

Fundamento teórico

Las seis etapas, en palabras simples

No.	Etapa	En qué consiste
1	Planteamiento	Decir con claridad qué se pide.
2	Análisis	Separar entradas, cálculos y salidas.
3	Diseño	Escribir los pasos en papel o en pseudocódigo.
4	Codificación	Pasar esos pasos a la computadora.
5	Prueba	Ejecutar con varios números y revisar el resultado.
6	Documentación	Poner el nombre, la fecha y un comentario.

Tipos de datos que se usan hoy

Tipo	Almacena	Ejemplo
`Entero`	Número sin decimales	`0`, `61`, `100`
`Real`	Número con decimales	`15.5`, `3.14`
`Cadena`	Texto	`"Ana"`, `"Universidad"`
`Logico`	Verdadero o falso	`Verdadero`, `Falso`

Orden en que se calculan las expresiones

Se calcula primero lo que está entre paréntesis. Después, de arriba hacia abajo:

Orden	Operadores	Ejemplo
1	`( )`	`(7 + 3) * 2` vale `20`
2	`*  /  MOD`	`7 + 3 * 2` vale `13`
3	`+  -`	`10 - 4` vale `6`
4	`<  <=  >  >=  =  <>`	`61 >= 61` es `Verdadero`
5	`Y`	Las dos condiciones deben cumplirse
6	`O`	Basta con que se cumpla una

El **cociente entero** (en muchos libros, `DIV`) se escribe en PSeInt como `trunc(17 / 5)` y vale `3`: `trunc` quita los decimales. El **residuo** se escribe `17 MOD 5` y vale `2`. Sirven, por ejemplo, para pasar minutos a horas.

Procedimiento

Parte A. Análisis del problema: entradas, proceso y salidas — 40 min

Se usará el mismo problema en toda la sesión. Es más corto que el reglamento completo de notas: así se puede terminar con calma.

Problema. Un estudiante tiene dos parciales. Cada parcial vale como máximo 30 puntos. El examen final vale como máximo 40 puntos. El algoritmo debe sumar las tres notas y decir si aprobó. Se aprueba con 61 puntos o más.

1. Lea el problema dos veces. En la primera lectura marque *qué se pide*. En la segunda, *con qué números se calcula*.
2. Complete la tabla. Si no está seguro del tipo, pregunte al instructor antes de seguir.

Tabla 01.1. Tabla de análisis Entrada-Proceso-Salida

Dato	Clasificación	Tipo	Rango
Primer parcial	Entrada		0 a 30
Segundo parcial	Entrada		0 a 30
Examen final	Entrada		0 a 40
Suma de las tres notas	Proceso	—	—
Nota final	Salida		
Aprobó (sí o no)	Salida		

NOTA

Esta tabla se llena **en el cuaderno o en este manual**, todavía sin abrir PSeInt. El teclado viene después.

Parte B. Escribir y ejecutar el primer algoritmo

50 min

1. Abra PSeInt. Cree un algoritmo nuevo y guárdelo como `P01_nota.psc` en la carpeta de la práctica.
2. Copie el algoritmo de abajo **tal como está**. No cambie los nombres todavía.
3. Pulse el botón de ejecutar. Cuando PSeInt pida los datos, use `20`, `25` y `30`.
4. Calcule a mano: $20 + 25 + 30 = 75$. ¿Coincide con la pantalla? Anote el resultado.

PRIMER ALGORITMO: SUMA DE TRES NOTAS

```
Algoritmo CalculoNota
  // Universidad Rural de Guatemala - Algoritmos FE729
  // Proposito: sumar dos parciales y el examen final
  // Autor: (escriba su nombre)   Fecha: (escriba la fecha)

  // Declaracion de variables
  Definir NOTA_MINIMA Como Entero
  Definir parcial1, parcial2, examenFinal, notaFinal Como Real
  Definir aprobado Como Logico

  // Inicializacion
  NOTA_MINIMA <- 61

  Escribir "Primer parcial (0 a 30):"
  Leer parcial1

  Escribir "Segundo parcial (0 a 30):"
  Leer parcial2

```

```
20    Escribir "Examen final (0 a 40):"
21    Leer examenFinal
22
23    // Calculos
24    notaFinal <- parcial1 + parcial2 + examenFinal
25    aprobado <- notaFinal >= NOTA_MINIMA
26
27    // Resultados
28    Escribir "Nota final: ", notaFinal
29    Escribir "Aprobado: ", aprobado
30  FinAlgoritmo
```

NOTA

PSeInt no tiene la palabra `Const`. Una constante se declara como variable con `Definir` y se le asigna su valor **una sola vez**, como `NOTA_MINIMA <- 61`. El nombre en MAYÚSCULAS recuerda que no debe cambiar.

5. Ejecute otra vez con `20`, `20` y `20`. La nota es 60. ¿Aparece `Falso` en aprobado? Debe ser así, porque 60 es menor que 61.
6. En su cuaderno, complete la frase: *NOTA_MINIMA es una constante porque* _____.

RECOMENDACIÓN

Si PSeInt marca un error, casi siempre falta el `FinAlgoritmo`, una coma o una comilla. Lea el mensaje en español: indica la línea.

Parte C. Expresiones aritméticas, `DIV` y `MOD` — 45 min

1. Cree el archivo `P01_expresiones.psc`.
2. Resuelva **primero a mano** las seis expresiones. Luego escríbalas en PSeInt con `Escribir` y compare.

Tabla 01.2. Evaluación de expresiones

No.	Expresión	A mano	En PSeInt
1	`7 + 3 * 2`		
2	`(7 + 3) * 2`		
3	`20 / 4`		
4	`trunc(20 / 3)`		
5	`20 MOD 3`		
6	`17 MOD 5 + trunc(17 / 5) * 5`		

3. Copie y ejecute el algoritmo siguiente. Pruébelo con `130` minutos. Debe mostrar 2 horas y 10 minutos.

</> MINUTOS A HORAS CON TRUNC Y MOD

```
Algoritmo DescomponerTiempo
  // Convierte minutos en horas y minutos
  Definir totalMinutos, horas, minutos Como Entero

  Escribir "Cantidad de minutos:"
  Leer totalMinutos

  // Calculos
  horas <- trunc(totalMinutos / 60)
  minutos <- totalMinutos MOD 60

  // Resultados
  Escribir horas, " hora(s) y ", minutos, " minuto(s)"
FinAlgoritmo
```

4. Pruebe también con `60` y con `59`. Anote los dos resultados en el cuaderno.

Parte D. Expresiones lógicas y condición de aprobado

40 min

1. Use $a = 5$, $b = 10$ y $c = 5$. Complete la tabla a mano y luego verifíquela en PSeInt.

Tabla 01.3. Expresiones lógicas con a=5, b=10, c=5

No.	Expresión	A mano	En PSeInt
1	`a = c`		
2	`a <> b`		
3	`a < b Y b > c`		
4	`a > b O b > c`		
5	`NO (a = c)`		
6	`a + 5 = b Y a = c`		

2. Abra de nuevo `P01_nota.psc`. Agregue la línea `Definir` junto a las demás declaraciones y las otras cuatro líneas después de `Leer examenFinal`:

</> VALIDAR LOS DATOS DE ENTRADA

```
  // Junto a los demas Definir:
  Definir datosOk Como Logico

  // Despues de Leer examenFinal:
```

```
5   datosOk <- (parcial1 >= 0) Y (parcial1 <= 30)
6   datosOk <- datosOk Y (parcial2 >= 0) Y (parcial2 <= 30)
7   datosOk <- datosOk Y (examenFinal >= 0) Y (examenFinal <= 40)
8   Escribir "Datos validos: ", datosOk
```

3. Pruebe un caso válido y otro con un parcial igual a `35`. Cada línea agrega una revisión con `Y`: `datosOk` solo sigue en `Verdadero` si todas se cumplen.

ADVERTENCIA

`Y` se evalúa antes que `O`. Si una expresión mezcla ambos, use paréntesis. Así no hay duda de qué se calcula primero.

EVALUACIÓN EN LÍNEA (SOCRATIVE)

Al terminar la práctica se aplicará una evaluación breve en **Socrative**. Las preguntas combinan la teoría de la clase magistral en línea con lo que se acaba de hacer en el laboratorio.
El instructor indicará el código de la sala. Anótelo aquí, junto con cualquier indicación del día.

Código de sala / enlace

Espacio para la evaluación

Reservado para Socrative: el instructor colocará aquí el acceso o las indicaciones

02 PRÁCTICA
NO. 02

Decisiones, repeticiones y una lista de notas

SESIÓN	DURACIÓN	UNIDADES	HERRAMIENTA
2 de 4	4 horas	4 y 5	PSeInt

Introducción

Hasta ahora el algoritmo hacía las mismas operaciones siempre. Hoy se aprende a **decidir** y a **repetir**.

Una **estructura selectiva** elige un camino. El caso más usado es `Si …Entonces …SiNo …FinSi`: si la nota llega a 61 se muestra APROBADO; si no, REPROBADO.

Una **estructura repetitiva** vuelve a hacer lo mismo varias veces. El ciclo `Mientras` pregunta primero y puede no entrar nunca. El ciclo `Para` se usa cuando ya se sabe cuántas veces se va a repetir, por ejemplo *cinco notas*.

Cuando hay varias notas, no se declaran cinco variables sueltas: se usa un **arreglo** (una lista). En esta sesión el arreglo es pequeño: solo cinco posiciones. No se trabajan matrices.

OBJETIVOS

- Usar `Si` / `SiNo` para mostrar APROBADO o REPROBADO.
- Asignar una categoría de nota con varias decisiones encadenadas.
- Repetir una lectura con `Mientras` hasta que el usuario termine.
- Llenar un arreglo de cinco notas con `Para` y calcular el promedio.

MATERIALES Y EQUIPO

- 1 computadora con PSeInt y acceso a Socrative.
- 1 cuaderno para la tabla de seguimiento.
- El archivo `P01_nota.psc` de la práctica anterior, si todavía lo tiene.

PLANIFICACIÓN DE LA SESIÓN

Parte	Actividad	Tiempo
—	Apertura y repaso breve	15 min
A	Decidir: aprobado o reprobado	40 min
B	Varias decisiones: la categoría de la nota	40 min
C	Repetir con `Mientras`	45 min
D	Cinco notas en un arreglo	50 min
—	Evaluación en Socrative y cierre	25 min
	Total	**215 min + 25 min**

Fundamento teórico

Cuándo usar cada ciclo

Ciclo	Pregunta la condición	Se usa cuando
`Mientras`	Antes de entrar	No se sabe si habrá que repetir.
`Repetir-Hasta`	Al final	Hay que hacerlo por lo menos una vez, por ejemplo al pedir de nuevo una nota.
`Para`	Lleva un contador	Ya se sabe el número de vueltas: 5 notas, 10 filas.

Un arreglo en una frase

Si se escribe `Definir notas Como Real` y después `Dimension notas[5]`, hay cinco cajas: `notas[1]` hasta `notas[5]`. En PSeInt (perfil Flexible) el primer índice es `1`, no `0`.

ADVERTENCIA

No pida `notas[0]` ni `notas[6]`. Esas posiciones no existen y el programa se detiene o muestra un valor incorrecto.

Procedimiento

Parte A. Decidir: aprobado o reprobado

40 min

1. Cree `P02_aprobado.psc` y copie el algoritmo. Ejecútelo con las notas `45`, `61` y `95`.

</> SELECTIVA DOBLE Y SELECTIVA SIMPLE

```
Algoritmo EvaluarEstudiante
  Definir NOTA_MINIMA Como Entero
  Definir notaFinal Como Real
  NOTA_MINIMA <- 61

  Escribir "Ingrese la nota final:"
  Leer notaFinal

  Si notaFinal >= NOTA_MINIMA Entonces
    Escribir "APROBADO"
  SiNo
    Escribir "REPROBADO"
  FinSi

  Si notaFinal >= 90 Entonces
    Escribir "Felicitaciones"
  FinSi
FinAlgoritmo
```

2. Complete: con `61` debe decir APROBADO y no debe felicitar. Con `95` debe decir las dos cosas.

Tabla 02.1. Salida observada

Nota	Mensajes que aparecieron
45	
61	
95	

Parte B. Varias decisiones: la categoría de la nota

40 min

1. Cree `P02_categoria.psc`. Copie el algoritmo y pruébelo con `50`, `70`, `80` y `92`.

Tabla 02.2. Escala que se usa en esta práctica

Rango	Categoría
90 a 100	Sobresaliente
76 a 89	Notable
61 a 75	Satisfactorio
0 a 60	Insuficiente

</> DECISIONES ENCADENADAS

```
Algoritmo CategoriaDeNota
  Definir nota Como Real
  Definir categoria Como Cadena

  Escribir "Nota (0 a 100):"
  Leer nota

  Si nota >= 90 Entonces
    categoria <- "Sobresaliente"
  SiNo
    Si nota >= 76 Entonces
      categoria <- "Notable"
    SiNo
      Si nota >= 61 Entonces
        categoria <- "Satisfactorio"
      SiNo
        categoria <- "Insuficiente"
      FinSi
    FinSi
  FinSi

  Escribir "Categoria: ", categoria
FinAlgoritmo
```

NOTA

Cuando hay *rangos* (de 61 a 75) se usa `Si` encadenado. El `Segun` sirve para valores sueltos, como el número de un día, no para intervalos.

2. Si le sobra tiempo, copie este `Segun` en otro archivo y pruébelo con `1`, `6` y `9`.

</> SELECTIVA MULTIPLE, OPCIONAL

```
Algoritmo NombreDelDia
  Definir dia Como Entero

  Escribir "Numero del dia (1 a 7):"
  Leer dia

  Segun dia Hacer
    1:
      Escribir "Lunes"
    2:
      Escribir "Martes"
    3:
      Escribir "Miercoles"
    4:
      Escribir "Jueves"
    5:
      Escribir "Viernes"
    6, 7:
      Escribir "Fin de semana"
    De Otro Modo:
      Escribir "Valor invalido"
  FinSegun
FinAlgoritmo
```

Parte C. Repetir con `Mientras` — 45 min

Se van a leer notas hasta que el usuario escriba `-1`. Ese valor se llama **centinela**: no es una nota, es la señal de *ya terminé*.

1. Cree `P02_promedio.psc` y copie el algoritmo.
2. Ejecútelo con la secuencia `80`, `90`, `70`, `-1`.
3. Complete la tabla de seguimiento. En cada fila anote los valores *después* de sumar esa nota.

</> CICLO MIENTRAS CON CENTINELA

```
Algoritmo PromedioConCentinela
  Definir CENTINELA, cantidad Como Entero
  Definir nota, suma, promedio Como Real
  CENTINELA <- -1

  suma <- 0
  cantidad <- 0

  Escribir "Ingrese notas. Escriba -1 para terminar."
  Leer nota

  Mientras nota <> CENTINELA Hacer
```

```
    suma <- suma + nota
    cantidad <- cantidad + 1
    Leer nota
  FinMientras

  Si cantidad > 0 Entonces
    promedio <- suma / cantidad
    Escribir "Cantidad: ", cantidad
    Escribir "Promedio: ", promedio
  SiNo
    Escribir "No se ingresaron notas"
  FinSi
FinAlgoritmo
```

Tabla 02.3. Seguimiento con 80, 90, 70, -1

Después de	nota	suma	cantidad
Empezar		0	0
Primera nota	80		
Segunda nota	90		
Tercera nota	70		

ADVERTENCIA

Dentro del `Mientras` **siempre** debe volver a leerse `nota`. Si se olvida esa línea, el ciclo no termina.

Parte D. Cinco notas en un arreglo

50 min

1. Cree `P02_arreglo.psc` y copie el algoritmo.
2. Ingrese cinco notas de su elección. Verifique el promedio a mano.

ARREGLO DE CINCO NOTAS

```
Algoritmo CincoNotas
  Definir CANTIDAD, NOTA_MINIMA, i, aprobados Como Entero
  Definir suma, promedio Como Real
  Definir notas Como Real
  Dimension notas[5]

  CANTIDAD <- 5
  NOTA_MINIMA <- 61
  suma <- 0
  aprobados <- 0

```

```
  Para i <- 1 Hasta CANTIDAD Con Paso 1 Hacer
    Escribir "Nota ", i, ":"
    Leer notas[i]
    suma <- suma + notas[i]
    Si notas[i] >= NOTA_MINIMA Entonces
      aprobados <- aprobados + 1
    FinSi
  FinPara

  promedio <- suma / CANTIDAD

  Escribir "Promedio: ", promedio
  Escribir "Aprobados: ", aprobados, " de ", CANTIDAD
FinAlgoritmo
```

Tabla 02.4. Mis cinco notas

1	2	3	4	5	Promedio

RECOMENDACIÓN

El `Para` recorre el arreglo del 1 al 5. En cada vuelta, `i` es el número de la caja que se está usando.

EVALUACIÓN EN LÍNEA (SOCRATIVE)

Al terminar la práctica se aplicará una evaluación breve en **Socrative**. Las preguntas combinan la teoría de la clase magistral en línea con lo que se acaba de hacer en el laboratorio.
El instructor indicará el código de la sala. Anótelo aquí, junto con cualquier indicación del día.

Código de sala / enlace

Espacio para la evaluación

Reservado para Socrative: el instructor colocará aquí el acceso o las indicaciones

03 PRÁCTICA
NO. 03

Dividir el trabajo: procedimientos y funciones

SESIÓN	DURACIÓN	UNIDADES	HERRAMIENTA
3 de 4	4 horas	6	PSeInt

Introducción

Cuando un algoritmo crece, conviene partirlo en pedazos. Cada pedazo hace **una sola cosa** y tiene un nombre claro.

Hay dos tipos de pedazo:

- Un **procedimiento** *hace* algo: mostrar un título, pedir datos, imprimir una lista.
- Una **función** *devuelve* un valor: el promedio, el mayor, la cantidad de aprobados.

El algoritmo principal queda corto: solo llama a esos pedazos. Eso se llama **diseño descendente**: primero se piensa el problema grande y luego se baja a las tareas chicas.

Hoy no se traduce a C++ ni se usa recursividad. El objetivo es ver, con ejemplos ya escritos, cómo se llama un procedimiento y cómo se usa el resultado de una función.

OBJETIVOS

- Dibujar un esquema simple: algoritmo principal y tres módulos.
- Copiar y llamar un procedimiento desde el algoritmo principal.
- Copiar y usar una función que devuelve el promedio.
- Observar la diferencia entre pasar un dato por valor y por referencia.

MATERIALES Y EQUIPO

- 1 computadora con PSeInt y acceso a Socrative.
- 1 cuaderno para el esquema de módulos.

PLANIFICACIÓN DE LA SESIÓN

Parte	Actividad	Tiempo
—	Apertura y repaso	15 min
A	Esquema: partir el problema en tres	35 min
B	Un procedimiento que muestra un título	40 min
C	Una función que calcula el promedio	50 min
D	Por valor y por referencia	50 min
—	Evaluación en Socrative y cierre	25 min
	Total	**215 min + 25 min**

Fundamento teórico

Procedimiento y función

	Procedimiento	Función
¿Devuelve un valor?	No	Sí, uno
¿Cómo se llama?	Como una instrucción	Dentro de una expresión o de un `Escribir`
Ejemplo de hoy	`MostrarTitulo`	`CalcularPromedio`

Por valor y por referencia

> **Por valor**: el módulo recibe una copia. Si la cambia, el original no se altera. **Por referencia**: el módulo trabaja sobre la variable original. En PSeInt se escribe `Por Referencia` después del nombre del parámetro, por ejemplo `SubProceso Cambiar(dato Por Referencia)`.

Procedimiento

Parte A. Esquema: partir el problema en tres — 35 min

Problema. Se van a leer tres notas, calcular su promedio y mostrar un mensaje con el resultado.

1. En el cuaderno dibuje tres cajas bajo el título *Algoritmo principal*:
   - `PedirNotas` — procedimiento
   - `CalcularPromedio` — función
   - `MostrarResultado` — procedimiento
2. Una las cajas con flechas que bajen desde el principal.
3. Complete la tabla.

Tabla 03.1. Los tres módulos

Nombre	Tipo	¿Qué hace?
PedirNotas	Procedimiento	
CalcularPromedio	Función	
MostrarResultado	Procedimiento	

RECOMENDACIÓN

Si al nombrar un módulo necesita la palabra "y" — por ejemplo *calcular y mostrar* — casi siempre son dos módulos.

Parte B. Un procedimiento que muestra un título

40 min

1. Cree `P03_titulo.psc` y copie el algoritmo. Ejecútelo.
2. Cambie el texto del título por el nombre de su curso y vuelva a ejecutar.

PROCEDIMIENTO SIN PARAMETROS

```
SubProceso MostrarTitulo
  Escribir "=============================="
  Escribir "  Laboratorio de Algoritmos  "
  Escribir "=============================="
FinSubProceso

Algoritmo DemoProcedimiento
  MostrarTitulo
  Escribir "El procedimiento ya termino."
FinAlgoritmo
```

NOTA

El procedimiento se *escribe* fuera del algoritmo principal (arriba de `Algoritmo`) y se *llama* desde adentro con su nombre. Si no se llama, no se ejecuta.

ADVERTENCIA

Si el procedimiento no recibe parámetros, se escribe *sin paréntesis*, tanto al declararlo (`SubProceso MostrarTitulo`) como al llamarlo (`MostrarTitulo`). Con paréntesis vacíos, PSeInt espera un parámetro adentro y marca el error "falta el nombre de argumento".

Parte C. Una función que calcula el promedio

50 min

1. Cree `P03_promedio.psc` y copie el algoritmo.
2. Ejecútelo con `80`, `70` y `90`. El promedio debe ser `80`.

FUNCION QUE DEVUELVE UN REAL

```
Funcion prom <- CalcularPromedio(n1, n2, n3)
  Definir prom Como Real
  prom <- (n1 + n2 + n3) / 3
FinFuncion

SubProceso MostrarResultado(promedio)
  Escribir "El promedio es: ", promedio
FinSubProceso

Algoritmo DemoFuncion
  Definir a, b, c, resultado Como Real

  Escribir "Tres notas:"
  Leer a, b, c
  resultado <- CalcularPromedio(a, b, c)
  MostrarResultado(resultado)
FinAlgoritmo
```

3. En el cuaderno escriba la línea que *guarda* el valor que devuelve la función. Es la que tiene `<- CalcularPromedio`.

Parte D. Por valor y por referencia

50 min

1. Cree `P03_parametros.psc` y copie el algoritmo.

2. Ejecútelo una vez. Observe los cuatro mensajes.

</> MISMA VARIABLE, DOS FORMAS DE PASARLA

```
SubProceso PorValor(dato Por Valor)
  dato <- 99
  Escribir "Dentro de PorValor, dato vale ", dato
FinSubProceso

SubProceso PorReferencia(dato Por Referencia)
  dato <- 99
  Escribir "Dentro de PorReferencia, dato vale ", dato
FinSubProceso

Algoritmo DemoParametros
  Definir numero Como Entero

  numero <- 10
  Escribir "Antes: ", numero
  PorValor(numero)
  Escribir "Despues de PorValor: ", numero

  numero <- 10
  PorReferencia(numero)
  Escribir "Despues de PorReferencia: ", numero
FinAlgoritmo
```

Tabla 03.2. Lo que se observó

Momento	Valor de numero
Antes de llamar	10
Después de `PorValor`	
Después de `PorReferencia`	

> Después de `PorValor` el número debe seguir en 10. Después de `PorReferencia` debe quedar en 99. Si no fue así, compare su archivo con el del manual línea por línea.

EVALUACIÓN EN LÍNEA (SOCRATIVE)

Al terminar la práctica se aplicará una evaluación breve en **Socrative**. Las preguntas combinan la teoría de la clase magistral en línea con lo que se acaba de hacer en el laboratorio.
El instructor indicará el código de la sala. Anótelo aquí, junto con cualquier indicación del día.

Código de sala / enlace

Espacio para la evaluación

Reservado para Socrative: el instructor colocará aquí el acceso o las indicaciones

04

PRÁCTICA
NO. 04

Proyecto guiado: un menú de tres notas

SESIÓN	DURACIÓN	UNIDADES	HERRAMIENTA
4 de 4	4 horas	1 a 6	PSeInt

Introducción

Las tres sesiones anteriores se juntan hoy en un solo programa, pero el programa ya viene casi escrito. El estudiante lo copia, lo completa en dos sitios señalados y lo prueba.

El sistema guarda **tres notas** en un arreglo, las muestra y calcula el promedio. Un menú permite elegir qué hacer. No se usa C++, ni búsqueda por nombre, ni archivos.

OBJETIVOS

- Reconocer las entradas, el proceso y las salidas del menú.
- Ejecutar un algoritmo que combina `Segun`, `Repetir` y un arreglo.
- Completar el cálculo del promedio y el conteo de aprobados.
- Probar el menú con un caso normal y un caso vacío.

MATERIALES Y EQUIPO

- 1 computadora con PSeInt y acceso a Socrative.
- Memoria USB para respaldar el archivo final.

PLANIFICACIÓN DE LA SESIÓN

Parte	Actividad	Tiempo
—	Apertura	15 min
A	Qué hace el sistema (tabla corta)	30 min
B	Copiar el menú y ejecutarlo	55 min
C	Completar promedio y aprobados	50 min
D	Probar dos casos y respaldar	40 min
—	Evaluación en Socrative y cierre	25 min
	Total	**215 min + 25 min**

Fundamento teórico

Hoy se reúnen herramientas ya usadas:

Unidad	Dónde aparece
1 y 2	Constantes, variables y el encabezado del algoritmo
3	La suma y la división del promedio
4	El `Segun` del menú y el `Si` de aprobado
5	El arreglo `notas[3]`
6	Los procedimientos `CargarNotas` y `MostrarNotas`

Procedimiento

Parte A. Qué hace el sistema — 30 min

El programa tiene cuatro opciones: (1) ingresar tres notas, (2) mostrarlas, (3) ver el promedio y cuántos aprobaron, (4) salir.

1. Complete la tabla antes de tocar el teclado.

Tabla 04.1. Análisis rápido

Pregunta	Respuesta
¿Qué entra?	
¿Qué se calcula?	
¿Qué se muestra?	
¿Con cuántas notas se trabaja?	

Parte B. Copiar el menú y ejecutarlo

55 min

1. Cree `P04_menu.psc` y copie todo el algoritmo.
2. Ejecútelo. Elija la opción 2 *antes* de cargar notas. Debe aparecer el mensaje de que todavía no hay datos.
3. Elija 1, ingrese `80`, `55` y `90`, y luego la opción 2.

</> MENU DE TRES NOTAS

```
SubProceso CargarNotas(notas, hayDatos Por Referencia)
  Definir i Como Entero
  Escribir "Ingrese las 3 notas:"
  Para i <- 1 Hasta 3 Con Paso 1 Hacer
    Escribir "Nota ", i, ":"
    Leer notas[i]
  FinPara
  hayDatos <- Verdadero
FinSubProceso

SubProceso MostrarNotas(notas, hayDatos)
  Definir i Como Entero
  Si hayDatos = Falso Entonces
    Escribir "Todavia no hay notas."
  SiNo
    Para i <- 1 Hasta 3 Con Paso 1 Hacer
      Escribir "Nota ", i, ": ", notas[i]
    FinPara
  FinSi
FinSubProceso

Algoritmo MenuTresNotas
  Definir CANTIDAD, NOTA_MINIMA, opcion, i, aprobados Como Entero
  Definir suma, promedio Como Real
  Definir hayDatos Como Logico
  Definir notas Como Real
  Dimension notas[3]

  CANTIDAD <- 3
  NOTA_MINIMA <- 61
```

```
  hayDatos <- Falso

  Repetir
    Escribir "1. Ingresar las tres notas"
    Escribir "2. Mostrar las notas"
    Escribir "3. Promedio y aprobados"
    Escribir "4. Salir"
    Escribir "Opcion:"
    Leer opcion

    Segun opcion Hacer
      1:
        CargarNotas(notas, hayDatos)
      2:
        MostrarNotas(notas, hayDatos)
      3:
        Escribir "(complete esta opcion en la parte C)"
      4:
        Escribir "Fin del programa"
      De Otro Modo:
        Escribir "Opcion invalida"
    FinSegun
  Hasta Que opcion = 4
FinAlgoritmo
```

Parte C. Completar promedio y aprobados

50 min

1. En la opción 3, **borre** el `Escribir` temporal que está debajo de `3:` y copie en su lugar el bloque siguiente (la línea `3:` se deja como está).

</> BLOQUE PARA LA OPCION 3

```
      3:
        Si hayDatos = Falso Entonces
          Escribir "Todavia no hay notas."
        SiNo
          suma <- 0
          aprobados <- 0
          Para i <- 1 Hasta CANTIDAD Con Paso 1 Hacer
            suma <- suma + notas[i]
            Si notas[i] >= NOTA_MINIMA Entonces
              aprobados <- aprobados + 1
            FinSi
          FinPara
          promedio <- suma / CANTIDAD
          Escribir "Promedio: ", promedio
          Escribir "Aprobados: ", aprobados, " de ", CANTIDAD
        FinSi
```

2. Guarde y ejecute. Con `80`, `55` y `90` el promedio es 75 y hay 2 aprobados.
3. Anote esos dos números en la tabla.

Tabla 04.2. Resultado de la opción 3

Dato	Valor
Promedio con 80, 55 y 90	
Aprobados de 3	

Parte D. Probar dos casos y respaldar

40 min

1. Marque en la tabla lo que ocurrió.
2. Copie `P04_menu.psc` a su USB o a la nube.

Tabla 04.3. Pruebas mínimas

Caso	¿Funcionó?	Qué se vio
Opción 2 sin haber cargado notas		
Cargar 80, 55, 90 y usar las opciones 2 y 3		
Escribir 9 en el menú		
Salir con la opción 4		

RECOMENDACIÓN

Cada `SubProceso` define su propia variable `i`: las variables del algoritmo principal no se ven dentro de los subprocesos. El arreglo `notas` sí llega, porque se pasa como parámetro.

EVALUACIÓN EN LÍNEA (SOCRATIVE)

Al terminar la práctica se aplicará una evaluación breve en **Socrative**. Las preguntas combinan la teoría de la clase magistral en línea con lo que se acaba de hacer en el laboratorio.
El instructor indicará el código de la sala. Anótelo aquí, junto con cualquier indicación del día.

Código de sala / enlace

Espacio para la evaluación

Reservado para Socrative: el instructor colocará aquí el acceso o las indicaciones

Anexo A. Convenciones de pseudocódigo

Todo el pseudocódigo de este manual está escrito con la **sintaxis de PSeInt** (perfil Flexible): se puede copiar y ejecutar tal como aparece. Conviene respetar estas convenciones en el laboratorio: así el instructor puede leer el algoritmo con rapidez y la evaluación en Socrative puede preguntar por esas mismas convenciones.

Estructura general de un algoritmo

</> PLANTILLA GENERAL DE UN ALGORITMO

```
Algoritmo NombreDescriptivo
  // Proposito: (que resuelve el algoritmo)
  // Autor: (nombre del estudiante)      Fecha: (dd/mm/aaaa)
  // Entradas: (datos que recibe)
  // Salidas: (datos que produce)

  // Declaracion de variables (tambien las que hacen de constante)
  Definir NOMBRE_CONSTANTE Como Entero
  Definir nombreVariable Como Real

  // Inicializacion: las constantes reciben su valor una sola vez
  NOMBRE_CONSTANTE <- 0

  // Entrada, proceso y salida
FinAlgoritmo
```

Los subprocesos y las funciones se escriben **fuera** del bloque `Algoritmo` … `FinAlgoritmo`, antes o después de él, y se llaman por su nombre desde el algoritmo principal.

Palabras reservadas

Categoría	Palabras reservadas
Delimitadores	`Algoritmo`, `FinAlgoritmo`
Declaraciones	`Definir`, `Como`, `Dimension`
Tipos de datos	`Entero`, `Real`, `Caracter`, `Cadena`, `Logico`
Entrada y salida	`Leer`, `Escribir`
Selectivas	`Si`, `Entonces`, `SiNo`, `FinSi`, `Segun`, `Hacer`, `De Otro Modo`, `FinSegun`
Repetitivas	`Mientras`, `FinMientras`, `Repetir`, `Hasta Que`, `Para`, `Hasta`, `Con Paso`, `FinPara`
Módulos	`SubProceso`, `FinSubProceso`, `Funcion`, `FinFuncion`, `Por Valor`, `Por Referencia`
Operadores lógicos	`Y`, `O`, `NO`, `Verdadero`, `Falso`
Operadores y funciones numéricas	`MOD`, `trunc( )`, `redon( )`, `rc( )`, `abs( )`

Del pseudocódigo general a PSeInt

En libros y en la clase en línea puede aparecer otra notación de pseudocódigo. Para ejecutarla en PSeInt se traduce así:

Notación general	PSeInt
`Const NOTA_MINIMA <- 61`	`Definir NOTA_MINIMA Como Entero` y después `NOTA_MINIMA <- 61`
`Var Real : nota`	`Definir nota Como Real`
`Inicio` … `Fin`	No se escriben: todo va entre `Algoritmo` y `FinAlgoritmo`
`Arreglo notas[5] de Real`	`Definir notas Como Real` y `Dimension notas[5]`
← ≥ ≤ ≠	`<-` `>=` `<=` `<>` (se teclean con dos caracteres)
`a DIV b`	`trunc(a / b)`
`SubProceso P(Por Referencia x)`	`SubProceso P(x Por Referencia)`
`SubProceso P()` y la llamada `P()`	`SubProceso P` y la llamada `P`: sin parámetros no se ponen paréntesis

NOTA

Si el perfil de PSeInt está en modo estricto, cada instrucción debe terminar con punto y coma (;) y los arreglos pueden empezar en el índice `0`. Este manual supone el perfil Flexible: `Configurar` → `Opciones del lenguaje (perfiles)`.

Convenciones de escritura

Elemento	Convención	Ejemplo
Nombre del algoritmo	`PascalCase`, sin espacios ni acentos	`CalculoNotaFinal`
Constantes	`MAYUSCULAS_CON_GUION`	`NOTA_MINIMA`
Variables	`camelCase`, descriptivas	`notaFinal`, `cantidad`
Módulos	`PascalCase`, con verbo en infinitivo	`CalcularPromedio`
Asignación	Menor que y guion, sin espacio	`suma <- suma + nota`
Comparación de igualdad	Un solo signo igual	`Si nota = 100`
Desigualdad	Signos menor y mayor	`Si nota <> 0`
Mayor o igual, menor o igual	Dos caracteres, sin símbolos especiales	`nota >= 61`, `nota <= 100`
Indentación	Dos espacios por nivel de anidamiento	—
Comentarios	Doble barra inclinada	`// explica el por qué`

Simbología de diagramas de flujo



NOTA

Todo diagrama de flujo tiene exactamente un símbolo de inicio y al menos uno de fin. Las líneas de flujo se dibujan preferentemente de arriba hacia abajo y de izquierda a derecha, y nunca se cruzan: cuando sería necesario cruzarlas, se emplea un conector.

Anexo B. Evaluación en línea (Socrative)

Cómo se evalúa cada práctica

No se entrega informe ni hoja de trabajo escrita. Al finalizar cada sesión se aplica una evaluación breve en **Socrative**. Esa evaluación es individual y vale 40 puntos de la práctica.

Las preguntas combinan dos cosas:

- **Teoría** de la clase magistral en línea: definiciones, operadores, estructuras de control y vocabulario básico.
- **Práctica** de esa misma sesión: lo que se escribió, se ejecutó y se anotó en las tablas.

Indicaciones para el estudiante

- La evaluación se realiza en los últimos minutos de la jornada.
- El instructor indicará el **código de la sala**. Anótelo en el espacio reservado al final de cada práctica.
- Ingrese con su nombre completo, tal como aparece en la lista del curso.
- Si el instructor no lo autoriza, no consulte el manual ni a un compañero durante la evaluación.

NOTA

El espacio al final de cada práctica está en blanco a propósito: allí se colocará después el código de sala o las indicaciones de acceso.

ADVERTENCIA

Copiar o hacerse pasar por otra persona en Socrative se sanciona conforme al Reglamento de Evaluación y Promoción Estudiantil de la Universidad Rural de Guatemala.

Referencias bibliográficas

1. Joyanes Aguilar, L. (2008). *Fundamentos de programación: algoritmos, estructura de datos y objetos* (4.ª ed.). McGraw-Hill Interamericana.
2. Joyanes Aguilar, L., & Zahonero Martínez, I. (2014). *Programación en C, C++, Java y UML* (2.ª ed.). McGraw-Hill Interamericana.
3. Cairó Battistutti, O. (2005). *Metodología de la programación: algoritmos, diagramas de flujo y programas* (3.ª ed.). Alfaomega.
4. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). *Introduction to algorithms* (4.th ed.). MIT Press.
5. Deitel, P., & Deitel, H. (2017). *C++ cómo programar* (10.ª ed.). Pearson Educación.
6. Wirth, N. (1987). *Algoritmos y estructuras de datos*. Prentice-Hall Hispanoamericana.
7. Novara, P. (2026). *PSeInt: intérprete de pseudocódigo* [Software]. https://pseint.sourceforge.io/
8. Universidad Rural de Guatemala. (2026). *Programa de la asignatura de Algoritmos, código FE729*. Facultad de Ingeniería.
9. Universidad Rural de Guatemala. (2026). *Calendario de laboratorios presenciales de Ingeniería en Sistemas, segundo semestre 2026*. Facultad de Ingeniería.



Tecnología, Ciencia y Conciencia

Ingeniería en Sistemas con especialización en Inteligencia Artificial

Manual de Laboratorio de Algoritmos • FE729

Versión 1.0 • Segundo Semestre 2026