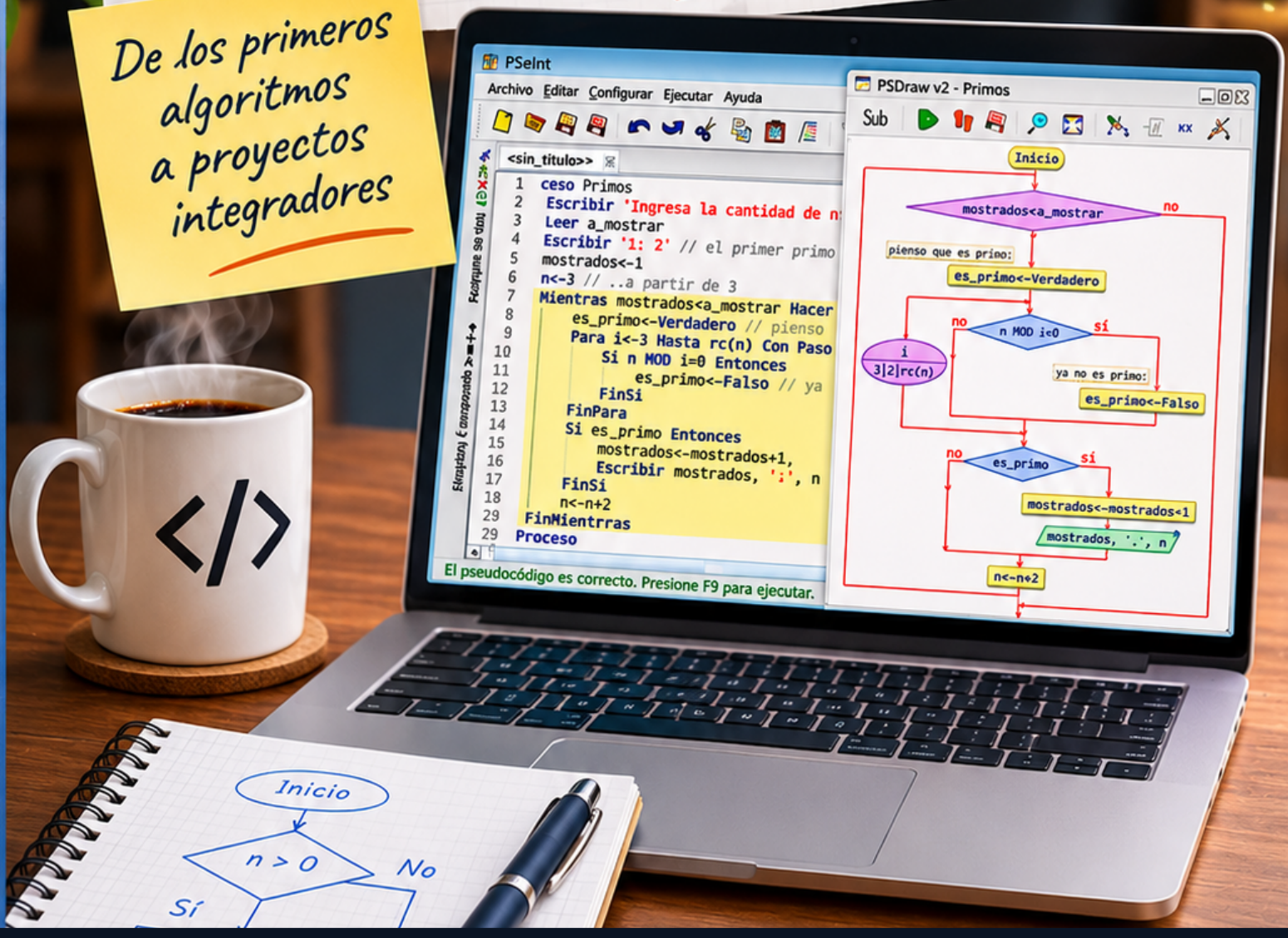
Club Programador

100 desafíos de lógica y programación con PSeInt

De los primeros algoritmos a proyectos integradores



MUESTRA GRATUITA · 10 EJERCICIOS

Edición 01

Créditos editoriales

100 desafíos de lógica y programación con PSeInt

De los primeros algoritmos a proyectos integradores

Autor y editor: Club Programador

Sitio web: https://clubprogramador.com

Contacto: info@clubprogramador.com

Edición: Primera edición · 2026

Formato: Libro digital PDF + archivos fuente .psc

Idioma: Español

Derechos y condiciones de uso

© 2026 Club Programador. Todos los derechos reservados. Esta obra y sus materiales originales se ofrecen para el estudio personal del comprador. Queda prohibida su redistribución, reventa o publicación no autorizada, sin perjuicio de las excepciones legales aplicables.

Independencia editorial

Esta publicación es independiente y no está patrocinada, avalada ni afiliada a los desarrolladores de PSeInt. PSeInt y los nombres de productos de terceros pertenecen a sus respectivos titulares. Los programas de ejemplo tienen finalidad educativa; su comportamiento puede variar según la versión y configuración utilizada.

Erratas y correcciones

Para informar una errata, escribí a info@clubprogramador.com indicando el número de ejercicio y una descripción del problema. Las correcciones editoriales que se publiquen podrán consultarse en clubprogramador.com. Esto no implica un compromiso de actualizaciones gratuitas ilimitadas.

Antes de empezar

Este libro está pensado para quienes comienzan a programar. No necesitás conocimientos previos: vas a conocer los conceptos esenciales, practicar con problemas progresivos y llegar a proyectos que combinan múltiples técnicas.

Qué incluye esta edición

- Fundamentos: qué es programar, algoritmos, pseudocódigo y PSeInt.
- Guía de instrucciones, estructuras de control y operadores.
- 100 ejercicios numerados, con soluciones y material explicativo.
- Archivos .psc para abrir los algoritmos en PSeInt.
- Anexo final con simbología y cinco ejemplos de diagramas de flujo.

Cómo aprovechar cada ejercicio

01. Leé el enunciado y definí entradas, salidas y restricciones.
02. Intentá escribir una solución propia antes de consultar la propuesta.
03. Compará tu código con el ejemplo, línea por línea.
04. Reproducí los casos de prueba, incluidos datos inválidos y límites.
05. Realizá el desafío adicional para reforzar lo aprendido.

La estructura de aprendizaje

Primero comprendé el problema; luego ejecutá y verificá la solución.
Los diagramas están reunidos en el anexo final para consulta independiente.

PARTE I — FUNDAMENTOS

1. Primeros pasos en programación

¿Qué es programar?

Programar consiste en diseñar y expresar una secuencia de instrucciones que una computadora pueda ejecutar para resolver una necesidad. Incluye comprender el problema, decidir cómo representaremos los datos, crear una solución, probarla y corregirla cuando sea necesario.

¿Qué es un algoritmo?

Un algoritmo es un conjunto ordenado y finito de pasos precisos para cumplir una tarea o resolver un problema. No tiene por qué estar escrito para una computadora: una receta con cantidades, operaciones y condiciones también puede describirse como un algoritmo.

Por ejemplo, para calcular el vuelto de una compra: (1) conocer el importe de la compra, (2) conocer el dinero entregado, (3) verificar que alcance y (4) restar el total al dinero recibido.

> La idea central: un programa implementa un algoritmo mediante instrucciones que la computadora interpreta o ejecuta. Diseñar bien los pasos es tan importante como escribir bien su sintaxis.

¿Qué es el pseudocódigo?

Es una manera de describir algoritmos con expresiones próximas al lenguaje humano y una estructura semejante a la de un programa. No busca seguir exactamente la sintaxis de Python, PHP o JavaScript; su objetivo es comunicar la lógica con claridad.

¿Por qué usamos pseudocódigo?

Porque permite concentrarse en la secuencia de acciones, las entradas, las condiciones y los ciclos antes de afrontar los detalles de un lenguaje específico. Además, facilita revisar una solución con otras personas y detectar errores de razonamiento. No reemplaza el aprendizaje posterior de un lenguaje real: lo prepara.

¿Qué es PSeInt?

PSeInt es una herramienta educativa que permite escribir y ejecutar algoritmos mediante pseudocódigo en español. Su sintaxis y algunas opciones pueden depender del perfil de configuración elegido. En este libro se utiliza la notación con Algoritmo, Definir, Leer, Escribir y asignación <-.

Un primer programa

```
Algoritmo PrimerSaludo
    Definir nombre Como Cadena
    Escribir "¿Cómo te llamás?"
    Leer nombre
    Escribir "Hola, ", nombre
FinAlgoritmo
```

El programa declara una variable de texto, muestra una pregunta, espera una respuesta y construye un saludo. Este patrón de entrada → proceso → salida aparece en muchos de los ejercicios siguientes.

2. Datos, variables y asignación

Una variable es un espacio identificado por un nombre cuyo valor puede cambiar durante la ejecución. Conviene elegir nombres descriptivos, como precio, cantidad o promedio.

Tipo	Qué representa	Ejemplo
Entero	Números sin parte decimal	cantidad = 3
Real	Números con parte decimal	precio = 12.5
Cadena	Texto	nombre = "Ana"
Caracter	Un carácter	inicial = "A"
Logico	Verdadero o Falso	aprobado = Verdadero

Declaración e inicialización

```
Definir edad Como Entero
Definir precio Como Real
Definir nombre Como Cadena
Definir valido Como Logico
edad <- 18
precio <- 250.5
```

El operador <- asigna el valor de la expresión de la derecha a la variable de la izquierda. No significa igualdad matemática: contador <- contador + 1 incrementa el contador.

3. Entrada, salida y estructuras de control

Escribir

Presenta textos, variables y resultados en pantalla. Podés combinar varias expresiones separadas por comas.

```
Escribir "Edad: ", edad
```

Leer

Solicita un dato al usuario y lo almacena en la variable indicada. Debe ser compatible con el tipo declarado.

```
Definir edad Como Entero
Leer edad
```

> Tené en cuenta: Un algoritmo robusto también verifica si el valor ingresado respeta las restricciones del problema.

Si, Entonces, SiNo y FinSi

Permiten tomar una decisión. Si la condición es verdadera se ejecuta una rama; de lo contrario, la rama SiNo, cuando existe.

```
Si edad >= 18 Entonces
    Escribir "Mayor de edad"
SiNo
    Escribir "Menor de edad"
FinSi
```

Segun, Hacer y FinSegun

Seleccionan una alternativa en función del valor de una expresión, útil para menús con opciones discretas.

```
Segun opcion Hacer
    1: Escribir "Consultar"
    2: Escribir "Salir"
    De Otro Modo: Escribir "Opción inválida"
FinSegun
```

Mientras, Hacer y FinMientras

Repite instrucciones mientras la condición se mantenga verdadera. Puede no ejecutarse ninguna vez.

```
Mientras cantidad < 3 Hacer
    Escribir cantidad
    cantidad <- cantidad + 1
FinMientras
```

> Tené en cuenta: Si la condición nunca cambia a falso, el ciclo podría no finalizar.

Repetir y Hasta Que

Ejecuta las instrucciones al menos una vez y repite hasta que la condición sea verdadera.

```
Repetir
    Escribir "Ingresá un número positivo:"
    Leer numero
Hasta Que numero > 0
```

Para y FinPara

Repite con una variable de control que toma sucesivos valores. Es útil cuando conocemos el número de repeticiones.

```
Para i <- 1 Hasta 5 Con Paso 1 Hacer
    Escribir i
FinPara
```

Dimension, vectores y matrices

Reserva espacio para colecciones de datos del mismo tipo. Un vector utiliza un índice; una matriz utiliza dos. Comprobá el perfil de indexación del intérprete.

```
Dimension notas[5]
Dimension ventas[3,4]
notas[1] <- 8
```

> Tené en cuenta: El acceso fuera de los límites produce errores. En los ejemplos del libro se debe verificar la configuración de indexación.

SubProceso y Funcion

Permiten separar responsabilidades, reutilizar lógica y dar nombres a operaciones. Una función devuelve un valor; un procedimiento puede realizar acciones y modificar parámetros según su declaración.

```
Funcion r <- Doble(n)
    Definir r Como Entero
    r <- n * 2
FinFuncion
```

Comentarios

Sirven para documentar decisiones y facilitar el mantenimiento. PSeInt permite comentarios de línea con barras dobles.

```
// Validar antes de calcular
Si divisor <> 0 Entonces
    resultado <- numero / divisor
FinSi
```

4. Operadores y expresiones

Operadores aritméticos

Símbolo	Significado	Ejemplo
+	Suma	a + b
-	Resta	a - b
*	Multiplicación	a * b
/	División	a / b
^	Potencia	a ^ 2
MOD	Resto entero	10 MOD 3 = 1

Operadores relacionales

Operador	Interpretación
=	Igual a
<>	Distinto de
> / <	Mayor / menor que
>= / <=	Mayor o igual / menor o igual

Operadores lógicos

Se utilizan para combinar o negar condiciones que producen Verdadero o Falso.

Operador	Significado	Ejemplo
Y	Ambas condiciones	edad >= 18 Y registrado
O	Al menos una condición	efectivo O tarjeta
NO	Invierte una condición	NO bloqueado

Tabla de verdad de Y y O

A	B	A Y B	A O B
V	V	V	V
V	F	F	V
F	V	F	V
F	F	F	F

Tabla de verdad de NO

A	NO A
V	F
F	V

Precedencia y paréntesis

Los paréntesis ayudan a dejar explícito el orden de evaluación. Cuando mezcles comparaciones y operadores lógicos, agrupá las condiciones relevantes para evitar interpretaciones equivocadas.

```
Si (edad >= 18 Y tieneDocumento) O tieneAutorizacion Entonces
    Escribir "Acceso permitido"
FinSi
```

5. Cómo abordar un ejercicio

Antes de escribir código, identificá las entradas, el resultado esperado, las restricciones y los casos límite. Después elegí las estructuras de control apropiadas, comprobá que cada repetición termine y prepará pruebas con datos normales, extremos e inválidos.

> Ruta de práctica: comprender → diseñar → programar → simular → probar → corregir → extender.

MÓDULO 01 / 08

Fundamentos y validaciones

Ejercicios 01–15. En esta unidad se desarrollan problemas de complejidad progresiva, con soluciones en pseudocódigo, seguimiento manual y casos para probar.

Objetivo de aprendizaje

Validar una edad entre 0 y 120 y clasificar mayoría de edad.

MÓDULO 01 · EJERCICIO 001

01. Control de acceso por edad

El problema

Validar una edad entre 0 y 120 e informar si la persona es mayor de edad. La entrada inválida debe solicitarse otra vez.

Solución en PSeInt

```
Algoritmo CP01_ControlEdad
    Definir edad Como Entero
    Escribir "Ingrese la edad (0 a 120):"
    Leer edad
    Mientras edad < 0 O edad > 120 Hacer
        Escribir "Edad invalida. Reingrese:"
        Leer edad
    FinMientras
    Si edad >= 18 Entonces
        Escribir "Mayor de edad"
    SiNo
        Escribir "Menor de edad"
    FinSi
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Ingresar -3, 121 y 18: se rechazan las dos primeras entradas y se imprime «Mayor de edad».

Casos de prueba

-1, 0, 17, 18, 120, 121

Error frecuente

Confundir validación (dato posible) con clasificación (mayor/menor).

Desafío adicional

Modificá una validación o condición y diseñá dos casos de prueba adicionales.

MÓDULO 01 · EJERCICIO 002

02. Venta con descuento limitado

El problema

Ingresar un precio positivo y un porcentaje de descuento entre 0 y 50. Calcular precio final sin permitir importes inválidos.

Solución en PSeInt

```
Algoritmo CP02_DescuentoControlado
    Definir precio, descuento, total Como Real
    Repetir
        Escribir "Precio positivo:"
        Leer precio
    Hasta Que precio > 0
    Repetir
        Escribir "Descuento entre 0 y 50:"
        Leer descuento
    Hasta Que descuento >= 0 Y descuento <= 50
    total <- precio * (1 - descuento / 100)
    Escribir "Total: ", total
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Para precio 200 y descuento 25, total = 150.

Casos de prueba

0; 200 y 0%; 200 y 50%; porcentaje -1; porcentaje 51

Error frecuente

Calcular el descuento sobre 100 como si fuera un importe fijo.

Desafío adicional

Modificá una validación o condición y diseñá dos casos de prueba adicionales.

MÓDULO 01 · EJERCICIO 003

03. Promedio con cantidad variable

El problema

Solicitar una cantidad de notas entre 1 y 20. Validar cada nota entre 0 y 10 y calcular el promedio.

Solución en PSeInt

```
Algoritmo CP03_PromedioValidado
    Definir n, i Como Entero
    Definir nota, suma, promedio Como Real
    Repetir
        Escribir "Cantidad de notas (1 a 20):"
        Leer n
    Hasta Que n >= 1 Y n <= 20
    suma <- 0
    Para i <- 1 Hasta n Hacer
        Repetir
            Escribir "Nota ", i, " (0 a 10):"
            Leer nota
        Hasta Que nota >= 0 Y nota <= 10
        suma <- suma + nota
    FinPara
    promedio <- suma / n
    Escribir "Promedio: ", promedio
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Para tres notas 7, 8 y 9, el promedio es 8.

Casos de prueba

n=0, n=1, n=20, nota -1, nota 10, nota 11

Error frecuente

Dividir por una cantidad no validada o acumular una nota inválida.

Desafío adicional

Modificá una validación o condición y diseñá dos casos de prueba adicionales.

MÓDULO 01 · EJERCICIO 004

04. Caja registradora con centinela

El problema

Leer importes positivos hasta que se ingrese 0. Mostrar cantidad de ítems y total; rechazar importes negativos.

Solución en PSeInt

```
Algoritmo CP04_CajaCentinela
    Definir cantidad Como Entero
    Definir importe, total Como Real
    cantidad <- 0
    total <- 0
    Escribir "Importe (0 para terminar):"
    Leer importe
    Mientras importe <> 0 Hacer
        Si importe > 0 Entonces
            total <- total + importe
            cantidad <- cantidad + 1
        SiNo
            Escribir "Importe negativo ignorado"
        FinSi
        Escribir "Importe (0 para terminar):"
        Leer importe
    FinMientras
    Escribir "Cantidad: ", cantidad
    Escribir "Total: ", total
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Secuencia 20, -5, 30, 0: cantidad = 2; total = 50.

Casos de prueba

0 como primera entrada, solo negativos, valores positivos repetidos

Error frecuente

Sumar el valor centinela o contar entradas inválidas.

Desafío adicional

Modificá una validación o condición y diseñá dos casos de prueba adicionales.

MÓDULO 01 · EJERCICIO 005

05. Separar dígitos con control de rango

El problema

Solicitar un entero positivo de tres cifras y mostrar centenas, decenas y unidades por separado.

Solución en PSeInt

```
Algoritmo CP05_DescomponerTresCifras
    Definir n, centenas, decenas, unidades Como Entero
    Repetir
        Escribir "Entero entre 100 y 999:"
        Leer n
    Hasta Que n >= 100 Y n <= 999
    centenas <- Trunc(n / 100)
    decenas <- Trunc((n MOD 100) / 10)
    unidades <- n MOD 10
    Escribir "Centenas: ", centenas
    Escribir "Decenas: ", decenas
    Escribir "Unidades: ", unidades
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Para n = 507 se obtiene 5 centenas, 0 decenas y 7 unidades.

Casos de prueba

99, 100, 507, 999, 1000

Error frecuente

No aplicar MOD antes de extraer las decenas.

Desafío adicional

Modificá una validación o condición y diseñá dos casos de prueba adicionales.

MÓDULO 01 · EJERCICIO 006

06. Clasificar una serie sin guardar sus valores

El problema

Ingresar una cantidad de números entre 1 y 30 y contar positivos, negativos y ceros. Verificar que los tres contadores sumen la cantidad ingresada.

Solución en PSeInt

```
Algoritmo CP06_Contadores
    Definir n, i, positivos, negativos, ceros Como Entero
    Definir valor Como Real
    Repetir
        Escribir "Cantidad (1 a 30):"
        Leer n
    Hasta Que n >= 1 Y n <= 30
    positivos <- 0
    negativos <- 0
    ceros <- 0
    Para i <- 1 Hasta n Hacer
        Escribir "Valor ", i, ":"
        Leer valor
        Si valor > 0 Entonces
            positivos <- positivos + 1
        SiNo
            Si valor < 0 Entonces
                negativos <- negativos + 1
            SiNo
                ceros <- ceros + 1
            FinSi
        FinSi
    FinPara
    Escribir "Positivos: ", positivos
    Escribir "Negativos: ", negativos
    Escribir "Ceros: ", ceros
    Escribir "Control total: ", positivos + negativos + ceros
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Con 2, -3, 0, 0 y 8: positivos = 2, negativos = 1 y ceros = 2.

Casos de prueba

todos positivos; todos ceros; mezcla de signos

> **Error frecuente**
>
> Olvidar el caso cero o usar condiciones excluyentes incorrectamente.

> **Desafío adicional**
>
> Modificá una validación o condición y diseñá dos casos de prueba adicionales.

MÓDULO 01 · EJERCICIO 007

07. Conversión de temperatura con límites

El problema

Convertir grados Celsius a Fahrenheit y Kelvin; rechazar temperaturas inferiores al cero absoluto.

Solución en PSeInt

```
Algoritmo CP07_Temperatura
    Definir c, f, k Como Real
    Repetir
        Escribir "Temperatura en Celsius:"
        Leer c
    Hasta Que c >= -273.15
    f <- c * 9 / 5 + 32
    k <- c + 273.15
    Escribir "Fahrenheit: ", f
    Escribir "Kelvin: ", k
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Con 0 grados Celsius: Fahrenheit = 32 y Kelvin = 273.15.

Casos de prueba

-274 (rechazar); -273.15; 0; 100

Error frecuente

No validar el cero absoluto antes de convertir.

Desafío adicional

Mostrar además si la temperatura es superior al punto de ebullición del agua.

MÓDULO 01 · EJERCICIO 008

08. Tarifa de estacionamiento

El problema

Calcular un precio por horas iniciadas, con límite diario y entrada válida.

Solución en PSeInt

```
Algoritmo CP08_Estacionamiento
    Definir minutos, horas, importe Como Entero
    Repetir
        Escribir "Minutos estacionados (positivo):"
        Leer minutos
    Hasta Que minutos > 0
    horas <- Trunc((minutos + 59) / 60)
    importe <- horas * 500
    Si importe > 4000 Entonces
        importe <- 4000
    FinSi
    Escribir "Horas facturadas: ", horas
    Escribir "Total: ", importe
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Con 61 minutos se facturan 2 horas y 1000 unidades. Con 600 minutos se aplica el tope de 4000.

Casos de prueba

1, 60, 61, 480, 600 y 0 (rechazado)

Error frecuente

Truncar la división sin redondear las horas iniciadas hacia arriba.

Desafío adicional

Agregar una tolerancia gratuita de cinco minutos.

MÓDULO 01 · EJERCICIO 009

09. Liquidación de compra

El problema

Obtener subtotal, impuesto y total a partir de cantidades y precios válidos.

Solución en PSeInt

```
Algoritmo CP09_Liquidacion
    Definir cantidad Como Entero
    Definir precio, subtotal, impuesto, total Como Real
    Repetir
        Escribir "Cantidad de unidades:"
        Leer cantidad
    Hasta Que cantidad > 0
    Repetir
        Escribir "Precio unitario:"
        Leer precio
    Hasta Que precio > 0
    subtotal <- cantidad * precio
    impuesto <- subtotal * 0.21
    total <- subtotal + impuesto
    Escribir "Subtotal: ", subtotal
    Escribir "Impuesto (21%): ", impuesto
    Escribir "Total: ", total
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Con 2 unidades a 100: subtotal 200, impuesto 42, total 242.

Casos de prueba

cantidad 0; precio -1; 1 a 100; 2 a 100

Error frecuente

Calcular el impuesto sobre el total en lugar del subtotal.

Desafío adicional

Permitir cargar tres productos y acumular el subtotal.

MÓDULO 01 · EJERCICIO 010

10. Cálculo de consumo eléctrico

El problema

Calcular costo por kilovatios con cargo fijo y consumo no negativo.

Solución en PSeInt

```
Algoritmo CP10_ConsumoElectrico
    Definir consumo, total Como Real
    Repetir
        Escribir "Consumo de kWh (no negativo):"
        Leer consumo
    Hasta Que consumo >= 0
    total <- 1200 + consumo * 85
    Escribir "Cargo fijo: 1200"
    Escribir "Importe total: ", total
FinAlgoritmo
```

Explicación y seguimiento manual

Paso / variable	Valor y explicación
Seguimiento	Con 10 kWh: cargo variable 850, fijo 1200, total 2050.

Casos de prueba

-1 (rechazar); 0; 10; 100

Error frecuente

Omitir el cargo fijo cuando el consumo es cero.

Desafío adicional

Incorporar dos tarifas por consumo según un umbral.

CLUB PROGRAMADOR

¿Querés seguir practicando?

Esta muestra incluye los primeros 10 ejercicios del libro.

La edición completa incluye 147 páginas, 100 ejercicios resueltos, 100 algoritmos en formato .psc y un anexo de diagramas de flujo.

Avanzá desde las primeras validaciones hasta problemas integradores y practicá a tu propio ritmo.

Conocé la edición completa