Documentación b1.php
Autor: Héctor Arciniega Valencia

---

1 <?php
Apertura del intérprete PHP.

2 /**
Inicio de DocBlock. No ejecuta.

3 * Nivel B – Ejercicio 1 (Condicional)
Título del ejercicio.

4 * Tarifa eléctrica simple por tramos
Descripción del problema.

5 *
Separador en DocBlock.

6 * Entrada: kwh (entero ≥ 0)
Define el dato esperado.

7 * Tarifas:
Encabezado de tabla de tramos.

8 * - 0..150 → $0.80/kWh
Tramo 1.

9 * - 151..300→ $1.00/kWh
Tramo 2.

10 * - >300 → $1.20/kWh
Tramo 3.

11 * Salida: kwh, tarifa_aplicada y costo_total (2 decimales).
Define salidas.

12 */
Fin DocBlock.

13 *(línea en blanco)*
Separación.

14 declare(strict_types=1);
Directiva. Activa tipado estricto en el archivo. Sin coerción implícita en firmas.

15 *(línea en blanco)*
Separación.

16 function validar_kwh(?string $raw): array {
Declara función. Parámetro $raw: string|null. Retorno: array con claves ok, val?, msg?.

17 // 1) Verificar que haya dato
Comentario.

18 if ($raw === null || $raw === '') {
Condicional. Detecta ausencia o vacío literal.

19 return ['ok'=>false, 'msg'=>'Dato vacío.'];
Retorno temprano. ok=false, msg describe el error.

20 }
Cierra if.

21 // 2) Debe ser entero no negativo (sin decimales)
Comentario.

22 if (!preg_match('/^\d+$/', trim($raw))) {
Valida formato con preg_match(patrón,sujeto).
Patrón ^\d+$: ancla inicio ^, clase \d dígito [0–9], cuantificador + uno o más, ancla fin $. Acepta “0”, “15”, “300”. Rechaza “-1”, “1.2”, “ 2”, “2 ”, “+3”.

23 return ['ok'=>false, 'msg'=>'Debe ser entero no negativo.'];
Retorno de error de formato.

24 }
Cierra if.

25 $val = (int)$raw;
Conversión explícita a entero. Seguro tras la regex.

26 // 3) Rango válido
Comentario.

27 if ($val < 0) {
Chequeo defensivo. Redundante por la regex, protege si cambia validación.

28 return ['ok'=>false, 'msg'=>'No puede ser negativo.'];
Retorno de error por rango.

29 }
Cierra if.

30 return ['ok'=>true, 'val'=>$val];
Éxito. Devuelve ok=true y val entero validado.

31 }
Fin de validar_kwh.

32 *(línea en blanco)*
Separación.

33 $kwh = null;
Inicializa consumo validado. null indica “sin calcular”.

34 $tarifa = null;
Inicializa tarifa aplicada por kWh.

35 $costo = null;
Inicializa costo total.

36 $error = null;
Inicializa mensaje de error.

37 *(línea en blanco)*
Separación.

38 if ($_SERVER['REQUEST_METHOD'] === 'POST') {
Procesa solo en peticiones POST. $_SERVER['REQUEST_METHOD'] es string “GET”/“POST”.

39 $res = validar_kwh($_POST['kwh'] ?? null);
Lee $_POST['kwh'] si existe, si no null (??). Llama validador. $_POST contiene pares nombre→valor enviados por el formulario.

40 if (!$res['ok']) {
Rama de error de validación.

41 $error = $res['msg'];
Guarda mensaje para mostrarlo.

42 } else {
Rama de éxito en validación.

43 $kwh = (int)$res['val'];
Asigna el entero validado.

44 // Determinar tarifa según tramo (bordes 150 y 300 incluidos donde corresponde).
Comentario.

45 if ($kwh <= 150) {
Condición tramo 1. Incluye 0..150.

46 $tarifa = 0.80;
Tarifa 0.80 por kWh.

47 } elseif ($kwh <= 300) {
Tramo 2. Aplica a 151..300.

48 $tarifa = 1.00;
Tarifa 1.00 por kWh.

49 } else {
Tramo 3. Aplica a >300.

50 $tarifa = 1.20;
Tarifa 1.20 por kWh.

51 }
Cierra condicional de tramos.

52 // Costo total = kwh * tarifa
Comentario.

53 $costo = $kwh * $tarifa;
Multiplica entero kwh por float tarifa. Resultado float.

54 }
Cierra else de validación OK.

55 }
Cierra if de método.

56 *(línea en blanco)*
Separación.

57 function fmt($n){ return number_format((float)$n, 2, '.', ','); }

Función de formateo.
Parámetro $n: numérico.
number_format(n,2,'.',','): 2 decimales, punto decimal . y coma de miles ,. Retorna string.

58 ?>
Cierra bloque PHP. Lo siguiente es HTML literal salvo bloques <?php ... ?>.

59 <!doctype html>
Declaración HTML5. Activa modo estándar.

60 <html lang="es">
Raíz del documento. Atributo lang="es": idioma español.

61 <head>
Inicio de cabecera.

62 <meta charset="utf-8" />
Metadato. charset="utf-8" evita problemas con acentos.

63 <title>NB-B1 Tarifa eléctrica</title>
Título de la pestaña.

64 <style>
Inicio de CSS embebido.

65 body{font-family:system-ui, -apple-system, Segoe UI, Roboto, Arial, sans-serif; margin:24px}
Regla CSS.
font-family: pila de fuentes del sistema.
margin:24px: margen exterior.

66 fieldset{border:1px solid #ccc;padding:12px;border-radius:8px;max-width:520px}
border, padding, border-radius, max-width del grupo.

67 label{display:block;margin:8px 0 4px}
display:block para ocupar línea. Márgenes.

68 input[type=number]{width:100%;padding:8px;border:1px solid #bbb;border-radius:6px}
Ancho, padding, borde y radios del input.

69 button{padding:8px 14px;border:0;border-radius:8px;background:#07a;color:#fff;cursor:pointer;margin-top:10px}

Estilo del botón.

70 .error{color:#b00020;font-weight:700}
Rojo y negrita para errores.

71 .result{margin-top:12px;padding:8px;border-left:4px solid
#07a;background:#eef7ff}
Bloque de resultados con borde lateral y fondo.

72 code{background:#f7f7f7;padding:2px 4px;border-radius:4px}
Estilo de <code>.

73 </style>
Fin CSS.

74 </head>
Fin cabecera.

75 <body>
Inicio cuerpo.

76 <h1>NB-B1 – Tarifa eléctrica simple</h1>
Encabezado visible.

77 <form method="post" novalidate>
Formulario.
method="post": envía en cuerpo HTTP.
novalidate: desactiva validación HTML5; validación queda en el servidor.

78 <fieldset>
Agrupa controles.

79 <legend>Entrada</legend>
Título del grupo.

80 <label for="kwh">kWh consumidos (entero ≥ 0)</label>
Etiqueta asociada al control id="kwh".

81 <input type="number" id="kwh" name="kwh" min="0" step="1" value="<?php echo
htmlspecialchars($_POST['kwh'] ?? ''); ?>" />
Campo numérico.
type="number": entrada numérica.
id="kwh" + name="kwh": identificador y clave en POST.
min="0": navegador impide negativos.

step="1": enteros.
value="...": repone último valor enviado. htmlspecialchars(texto): escapa & < > " '.
Previene XSS reflejado.

82 <button type="submit">Calcular</button>
Botón enviar. type="submit" dispara envío.

83 <?php if ($error): ?> <?php echo htmlspecialchars($error);
?> <?php endif; ?>
Bloque condicional embebido.
Muestra si $error es truthy.
Escapa el mensaje con htmlspecialchars para salida segura.

84 </fieldset>
Cierra grupo.

85 </form>
Cierra formulario.

86 *(línea en blanco)*
Separación.

87 <?php if ($kwh !== null && $tarifa !== null): ?>
Condición de presentación. Requiere que se haya calculado kwh y tarifa.

88
Contenedor de resultados.

89 Salida:
Etiqueta de sección.

90 kWh: <code><?php echo $kwh; ?></code>
Muestra consumo como número. No requiere escape adicional al ser int.

91 Tarifa aplicada: $<code><?php echo fmt($tarifa); ?></code> por kWh
Muestra tarifa formateada con fmt. Resultado es cadena segura.

92 Costo total: $<code><?php echo fmt($costo); ?></code>
Muestra costo total formateado.

93
Cierra resultados.

94 <?php endif; ?>

Fin de la condición de presentación.

95 </body>
Cierra cuerpo.

96 </html>
Cierra documento HTML.

- Puntos críticos aclarados:
- Regex ^\d+$ y sus casos aceptados/rechazados.
- Tramos con límites inclusivos: <=150, <=300, >300.
- number_format con parámetros exactos.
- Seguridad de salida: htmlspecialchars en entrada reinyectada y en errores.
- Flujo: procesa solo en POST; en GET muestra formulario en limpio.