

Ejercicios UP2 Lenguaje PHP

DWES - 2DAW

1. Escribe una función que calcule todas las potencias de un número hasta llegar al exponente indicado, las almacene en un vector y muestre el resultado de cada potencia indicando además la suma de todas las potencias incluyendo la del exponente indicado.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.f, RA3.g

2. Crea una función llamada permutaciones que reciba un vector \$V y que cambie la posición de los elementos dicho vector haciendo permutaciones. Las permutaciones se harán entre los elementos \$V[\$N-1] y \$V[0], \$V[\$N-2] y \$V[1], \$V[\$N-3] y \$V[2] etc. siendo \$N el tamaño del vector.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.f, RA3.g

3. Solicita al usuario nombre y salario para crear un vector asociativo de trabajadores. Usa funciones para calcular el salario máximo, mínimo y salario medio. Calcula además un posible incremento del sueldo a introducir por el usuario. Deberás mostrar los nombres y salarios iniciales, el salario máximo, mínimo y promedio en ese momento, aplicar después del incremento y mostrar los nuevos salarios con el salario máximo, mínimo y promedio después del incremento.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.d, RA3.f, RA3.g

4. Define tres arrays de 20 números enteros cada uno, con nombres "numero", "cuadrado" y "cubo". Carga el array "numero" con valores aleatorios entre 0 y 100. En el array "cuadrado" se deben almacenar los cuadrados de los valores que hay en el array "numero". En el array "cubo" se deben almacenar los cubos de los valores que hay en "numero". A continuación, muestra el contenido de los tres arrays dispuesto en tres columnas. Usa funciones en la medida de lo posible

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.d, RA3.f, RA3.g

5. Crear una matriz de tamaño 5x5 y rellenarla de la siguiente forma: la posición M[n,m] debe contener n+m. Después se debe mostrar su contenido en formato matriz. Además, se deberá añadir al resultado la suma de cada una de las filas y la suma de cada una de las columnas. Usa funciones en la medida de lo posible.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.d, RA3.f, RA3.g

6. Diseñar la función operaMatriz, a la que se le pasa dos matrices del mismo tamaño (a indicar por el usuario) de enteros positivos mayores de 0 (pueden ser random o introducidos por el usuario) y la operación que se desea realizar: sumar, restar, multiplicar o dividir (mediante un carácter: 's', 'r', 'm', 'd'). La función debe imprimir las matrices originales, indicar la operación a realizar y la matriz con los resultados. Usa funciones en la medida de lo posible.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.d, RA3.f, RA3.g

7. Desarrolla un programa **menu_basico** para imprimir un menú de opciones. Para ello se preguntará al usuario el número de opciones del menú, el carácter a usar para identificar cada opción (número o letra) y el texto de cada opción. Además se le preguntará por el carácter de terminación o fin del programa. La salida debe ser el menú impreso en pantalla y totalmente funcional, para probar las opciones, devuelve un texto indicando la opción elegida. Realiza el programa en un único bloque de instrucciones.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.g

8. Desarrolla un programa **menu2** con las mismas características que el del ejercicio anterior pero deberás aportar una solución modular usando funciones

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.d, RA3.g

9. Desarrolla un programa **menu** con las mismas características que el del ejercicio anterior pero deberás usar array asociativo.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.d, RA3.f, RA3.g

10. Desarrolla un programa **menu_submenu** donde además de lo indicado en el ejercicio anterior, las opciones de menú pueden contener a su vez un menú en su interior. La salida debe ser el menú impreso en pantalla y totalmente funcional con los submenús en las opciones que correspondan. Para probar las opciones, devuelve un texto indicando la opción elegida y en el caso del submenú, también la opción del menú principal.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.d, RA3.g

11. Crea un programa para gestionar los pedidos de un restaurante. Cada pedido tendrá: **numeroPedido, cliente, platos** (array de platos con **nombre** y **precio**). Se guardarán en un array asociativo con el número de pedido como clave. Usa funciones ya creadas en otros ejercicios cuando sea posible. El menú del programa debe permitir:

- Crear un pedido.
- Añadir plato a un pedido.
- Ver detalle de un pedido (platos y total).
- Listar todos los pedidos.
- Salir.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.d, RA3.g

12. Crea un programa que simule una biblioteca. Cada libro tendrá: **título, autor, año, prestado** (true/false). Los libros se guardan en un array asociativo con el título como clave. Usa funciones ya creadas en otros ejercicios cuando sea posible. El menú debe permitir:

- Añadir libro.
- Listar libros.
- Prestar libro (cambia prestado = true).
- Devolver libro (prestado = false).
- Buscar libros por autor.
- Salir.

RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA3.a, RA3.b, RA3.c, RA3.d, RA3.g