Ejercicio 1: Verificación de números pares e impares

Escribe un programa que verifique si un número es par o impar utilizando `if`.

Enunciado:

Solicita al usuario que ingrese un número y verifica si es par o impar.

Ejercicio 2: Calificación de una nota

Escribe un programa que determine si una nota numérica es "Aprobado" o "Reprobado" usando `if`.

Enunciado:

Solicita al usuario una calificación y determina si la nota es aprobatoria (>= 60) o reprobatoria (< 60).

Ejercicio 3: Calculadora básica

Utiliza `match` para implementar una calculadora simple.

Enunciado:

Crea una calculadora que solicite dos números y una operación matemática (+, -, *, /). Usa `match` para realizar la operación correspondiente.

Ejercicio 4: Determinación del tipo de triángulo

Escribe un programa que determine el tipo de triángulo en función de sus lados usando `if`.

Enunciado:

Solicita al usuario que ingrese las longitudes de los tres lados de un triángulo. Determina si el triángulo es equilátero, isósceles o escaleno.

Ejercicio 5: Días de la semana

Escribe un programa que, dado un número del 1 al 7, imprima el día correspondiente de la semana usando `match`.

Enunciado:

Solicita al usuario un número del 1 al 7 y muestra el día de la semana correspondiente (1 = Lunes, 7 = Domingo).

Ejercicio 6: Juego de adivinanza de números

Escribe un programa que implemente un juego de adivinanza de números.

Enunciado:

El programa genera un número aleatorio entre 1 y 10. El usuario debe adivinar el número, y el programa indica si el número ingresado es mayor, menor o igual al número generado.

Ejercicio 7: Número positivo, negativo o cero

Escribe un programa que determine si un número es positivo, negativo o cero usando `if`.

Enunciado:

Solicita al usuario que ingrese un número y determina si es positivo, negativo o cero.

Ejercicio 8: Determinación de año bisiesto

Escribe un programa que determine si un año es bisiesto o no.

Enunciado:

Solicita al usuario que ingrese un año y determina si es bisiesto (divisible entre 4, pero no entre 100, salvo que sea divisible entre 400).

Ejercicio 9: Clasificación de edades

Escribe un programa que clasifique a una persona en función de su edad.

Enunciado:

Solicita la edad de la persona e indica si es niño (0-12 años), adolescente (13-17 años), adulto (18-64 años) o anciano (65 años o más).

Ejercicio 10: Clasificación de notas

Escribe un programa que asigne una calificación basada en una nota numérica.

Enunciado:

Solicita una nota numérica y clasifícala como A (90-100), B (80-89), C (70-79), D (60-69), o F (<60).

Ejercicio 11: Conversión de temperaturas

Escribe un programa que convierta grados Celsius a Fahrenheit o Fahrenheit a Celsius usando `match`.

Enunciado:

Solicita al usuario que ingrese una temperatura y una escala (C o F). Convierte la temperatura a la escala opuesta usando `match`.

Ejercicio 12: Calculadora de IMC (Índice de Masa Corporal)

Escribe un programa que calcule el IMC y determine el estado de peso.

Enunciado:

Solicita al usuario su peso (en kg) y su altura (en metros). Calcula el IMC y clasifícalo en bajo peso (<18.5), peso normal (18.5-24.9), sobrepeso (25-29.9), o obesidad (>=30).

Ejercicio 13: Comparación de tres números

Escribe un programa que determine el mayor de tres números usando `if`.

Enunciado:

Solicita al usuario que ingrese tres números y determina cuál es el mayor.

Ejercicio 14: Adivinanza de letras

Escribe un programa que permita al usuario adivinar una letra secreta usando `match`.

Enunciado:

El programa contiene una letra secreta (por ejemplo, "A"). El usuario debe adivinar la letra, y el programa le indicará si acertó o no.

Ejercicio 15: Cálculo del salario neto

Escribe un programa que calcule el salario neto de un empleado después de aplicar impuestos.

Enunciado:

Solicita al usuario su salario bruto y su país de residencia. Calcula el salario neto aplicando impuestos: el 20% para residentes de "País A", el 15% para "País B" y el 10% para "País C". Si el país no está en la lista, aplica un 25% de impuestos.

Ejercicio 16: Cálculo del tiempo de viaje

Escribe un programa que calcule el tiempo que tarda en llegar un automóvil a su destino.

Enunciado:

Solicita al usuario la distancia a recorrer (en km) y la velocidad promedio del automóvil (en km/h). Calcula el tiempo de viaje en horas y minutos. Si la velocidad es mayor a 120 km/h, muestra un mensaje de advertencia.

Ejercicio 17: Sistema de calificaciones con bonificaciones

Escribe un programa que calcule la calificación final de un estudiante basándose en su calificación y si ha hecho tareas adicionales. Las tareas adicionales pueden darle un extra de puntos, pero el máximo de puntos no puede exceder 100.

Enunciado:

Solicita la calificación del estudiante y pregunta si hizo tareas adicionales. Si la respuesta es "sí", añade un 5% extra a la calificación, pero si la calificación supera 100, ajústala a 100. Si la respuesta es "no", simplemente muestra la calificación original.

Ejercicio 18: Sistema de evaluación de créditos universitarios

Escribe un programa que calcule el número de créditos totales de un estudiante en base a las materias cursadas y el puntaje obtenido en cada una. El puntaje debe ser evaluado como aprobado o no aprobado.

Enunciado:

Solicita al usuario ingresar el número de materias que ha cursado. Para cada materia, solicita el puntaje y determina si ha aprobado o no (>= 60). Luego, calcula el número total de créditos del estudiante (cada materia aprobada otorga 3 créditos).

Ejercicio 20: Conversión de calificaciones numéricas a letras

Escribe un programa que convierta una calificación numérica en una letra de acuerdo a un sistema de calificación específico, usando `match`.

Enunciado:

Solicita una calificación numérica (0-100) y convierte esa calificación a una letra usando el siguiente esquema:

- A: 90-100
- B: 80-89
- C: 70-79
- D: 60-69
- F: 0-59

Ejercicio 21: Sistema de estacionamiento con tarifas progresivas

Escribe un programa que calcule el costo de estacionamiento basado en el número de horas, con tarifas progresivas.

Enunciado:

El costo de estacionamiento se calcula de la siguiente manera:

- Primera hora: $5
- Segunda a cuarta hora: $4 por hora
- Más de cuatro horas: $3 por cada hora adicional

Solicita al usuario el número de horas de estacionamiento y calcula el costo total.

Ejercicio 22: Clasificación de triángulos por sus ángulos

Escribe un programa que clasifique un triángulo en agudo, obtuso o rectángulo según sus ángulos internos usando `if`.

Enunciado:

Solicita al usuario los tres ángulos de un triángulo y clasifícalo en:

- Agudo: Todos los ángulos son menores a 90°.
- Rectángulo: Un ángulo es exactamente 90°.
- Obtuso: Un ángulo es mayor a 90°.

Estructuras Iterativas

Ejercicio 1: Suma de los primeros N números enteros

Enunciado:
Escribe un programa que solicite al usuario un número entero positivo `n` y calcule la suma de los primeros `n` números enteros. Utiliza un ciclo `for` para realizar la suma.

Ejercicio 2: Contador de vocales en una cadena

Enunciado:
Escribe un programa que solicite al usuario una cadena de texto y cuente cuántas vocales (a, e, i, o, u) contiene. Usa un ciclo `for` para recorrer la cadena y realizar la cuenta.

Ejercicio 3: Factorial de un número

Enunciado:
Escribe un programa que solicite al usuario un número entero positivo `n` y calcule el factorial de dicho número (`n! = 1 * 2 * 3 * ... * n`). Usa un ciclo `for` para realizar el cálculo.

Ejercicio 4: Números pares en un rango

Enunciado:
Escribe un programa que solicite al usuario dos números enteros, un valor de inicio y un valor de fin. El programa debe imprimir todos los números pares en ese rango, incluyendo los límites. Usa un ciclo `for` para recorrer el rango.

Ejercicio 6: Adivina el número (con `while`)

Enunciado: (random.randint(1, 100))
Escribe un programa que genere un número aleatorio entre 1 y 100 y permita al usuario adivinarlo. El programa debe dar pistas si el número ingresado es mayor o menor que el número secreto. Usa un ciclo `while` para permitir al usuario seguir intentando hasta que adivine el número

Ejercicio 7: Suma de números pares hasta que se introduce un impar

Enunciado:
Escribe un programa que solicite al usuario ingresar números enteros indefinidamente. El programa debe sumar los números siempre que sean pares, pero debe detenerse si el usuario ingresa un número impar. Usa un ciclo `while` para lograr esto.