Ejercicio 1: Conversión

La gravedad de la Luna es aproximadamente el 17% de la gravedad de la Tierra. Se pide:

1. Crear un programa que, dado un peso de la Tierra, calcule su equivalente en la Luna.
2. El programa debe mostrar por pantalla un mensaje similar a este: “Un peso de 30 kilos en la Tierra equivalen a X kilos en la Luna”.

Ejercicio 2: Operaciones matemáticas

Crear un programa que realice algunas operaciones matemáticas.

1. Pedir por consola al usuario que introduzca 3 números enteros y almacenar su valor en variables.
2. Operación 1. a*b/c Mostrar por consola el resultado
3. Operación 2. (a*c)%bMostrar por consola el resultado
4. Operación 3. 2*(a+c-b)/(b*c) Mostrar por consola el resultado
5. Operación 4. ((a*c)+(b%a))/a-c Mostrar por consola el resultado
6. Opcional: Repetir las operaciones con números decimales para ver las diferencias de resultado.

Ejercicio 3: Notas

Crear un programa que indique al usuario la nota media global de su curso. Requisitos:

1. El programa pedirá al usuario que introduzca 5 notas, para las asignaturas: Matemáticas, Física, Química, Lenguaje e Historia.
2. El programa realizará la media de esas cinco notas.
3. El programa indicará al usuario el rango al que equivale la nota final que ha sacado:
    a. Entre 0 – 3 Muy deficiente
    b. Entre 3 – 5 Insuficiente
    c. Entre 5 – 6 Suficiente
    d. Entre 6 – 7 Bien
    e. Entre 7 – 9 Notable
    f. Entre 9 – 10 Sobresaliente

Ejercicio 4: Contar espacios

Crear un programa que lea caracteres desde teclado indefinidamente. Requisitos:

1. El programa pedirá al usuario que inserte caracteres o frases completas por teclado de manera indefinida
2. El programa contará el número de espacios que se van introduciendo por consola
3. El programa finalizará cuando el usuario introduzca un punto “.”
4. El programa imprimirá por consola el número total de espacios que se han introducido

Ejercicio 5: Amstron

Desarrollar un programa que determine si un número es un número de Amstrong. Un número de Amstrong es aquel que es igual a la suma de sus dígitos elevados a la potencia de su número de cifras.

Recursos para el ejercicio:

- Math.floor: Devuelve el máximo entero menor o igual a un número pasado como parámetro.
- Math.log10: Devuelve el logaritmo en base 10 de un número pasado como parámetro.
- Math.pow: Devuelve el valor del primer argumento elevado a la potencia del segundo argumento.

Ejercicio 6: Conversión 2

Al igual que se hizo con la Ejercicio 1 de conversión de pesos, en este caso se pide crear un programa que imprima una tabla de conversión de pulgadas a metros. Requisitos:

1. Un metros son 39,37 pulgadas
2. Elaborar el programa desde la pulgada 1 hasta la 144
3. La tabla debe de dejar un espacio libre cada 12 pulgadas para ser más legible.

Ejercicio 7: CandyCalculator

En las fiestas de un barrio hay competiciones deportivas que premian el desempeño en ellas con cupones que luego puedes cambiar por golosinas. Una barra de caramelo se puede cambiar por 10 cupones, y un chicle por 3 cupones.

Escribe una clase CandyCalculator que tenga un método candyCalculator que permita:

1. Calcular, dado un número dado de cupones, cuantas barras de caramelo y chicles puedes obtener si gastas todos tus cupones en barras de caramelo primero, y utilizas los cupones restantes en chicles.
2. También te tiene que devolver el número de cupones restantes que no puedes gastar.
3. Devuelve los valores en un array en donde la posición [0] representa las barras de caramelo, la posición [1] los chicles y la posición [2] los cupones restantes.

Ejercicio 8: Elementos Duplicados

Crear un programa que, dado un array de números enteros, determine cuales son sus elementos que se encuentran duplicados. Por ejemplo:

int [] arrayDePrueba = {1, 2, 3, 3, 9, 8, 7, 4, 6, 7, 0, 4, 5};

Ejercicio 9: Contar vocales

Implementa la función int contarVocales(String) que dada una cadena, cuenta el número de vocales que existe en la misma. No importa que las vocales estén en mayúsculas o minúsculas y el resto de caracteres que no sean vocales son ignorados. También se ignoran las vocales acentuadas.

Ejercicio 10: Persona

Se requiere un programa que modele el concepto de una persona. Una persona posee nombre, apellido, número de documento de identidad y año de nacimiento. La clase debe tener un constructor que inicialice los valores de sus respectivos atributos.

La clase debe incluir los siguientes métodos:

1. Definir un método que imprima por pantalla los valores de los atributos del objeto.
2. En el método main se deben crear dos personas y mostrar los valores de sus atributos por pantalla.

Ejercicio 11: Calculadora Básica

Crea una clase en Java llamada Calculadora que implemente una calculadora básica con las siguientes funcionalidades:

Suma: Un método que toma dos números como parámetros y devuelve la suma de ambos.

Resta: Un método que toma dos números como parámetros y devuelve la resta del primero menos el segundo.

Multiplicación: Un método que toma dos números como parámetros y devuelve su producto.

División: Un método que toma dos números como parámetros y devuelve el cociente resultante de la división del primero entre el segundo. Si el segundo número es cero, debe mostrar un mensaje indicando que la división por cero no está permitida.

En el programa principal, crea una instancia de la clase Calculadora, realiza varias operaciones y muestra los resultados.

Ejercicio 12: Rectángulo

Crea una clase Rectángulo con las siguientes características

- Incluyen un constructor al que se le pasan los datos de base y altura. Si se intenta dar valor negativo a alguna de las dimensiones, lo corrige al valor positivo usando Math.abs(int a).
- Un constructor sin parámetros que inicializa un nuevo rectángulo con base 2 y altura 1.
- Un método esCuadrado que indica si el rectángulo actual es un cuadrado.
- Un método area que calcula el área del rectángulo actual.
- Un método perimetro que calcula el perímetro del rectángulo actual.
- Un método gira que gira 90 grados el rectángulo actual (intercambiado la base por la altura).

Ejercicio 13: Sistema de Gestión de Animales

Crea un sistema de gestión de animales utilizando herencia en Java. Define una clase base llamada Animal con los siguientes atributos: nombre y edad. Implementa un constructor y métodos de acceso para estos atributos.

Luego, crea dos clases derivadas: Perro y Gato. Ambas clases deben heredar de la clase base Animal. La clase Perro debe incluir un atributo adicional para representar la raza del perro, y la clase Gato debe incluir un atributo para representar el color del pelaje.

En el programa principal, crea instancias de ambas clases, establece valores para sus atributos y muestra la información básica de cada animal, incluyendo su nombre, edad y características específicas de la especie.

Ejercicio 14: Figuras geométricas

Crea una jerarquía de clases para representar figuras geométricas. La clase base debe ser Figura y debe contener métodos para calcular el área y el perímetro. Luego, implementa subclases para representar diferentes tipos de figuras como Círculo, Rectángulo y Triángulo. Cada subclase debe heredar de la clase Figura y proporcionar implementaciones específicas para los cálculos de área y perímetro. En el programa principal, crea instancias de estas clases y muestra los resultados de los cálculos.

Ejercicio 15: Sistema de Gestión de Empleados

Crea un sistema de gestión de empleados utilizando herencia en Java. Define una clase base llamada Empleado con los siguientes atributos: nombre, edad y salario. Implementa un constructor y métodos de acceso para estos atributos.

Luego, crea dos clases derivadas: EmpleadoTiempoCompleto y EmpleadoTiempoParcial. La clase EmpleadoTiempoCompleto debe incluir un atributo adicional para representar el cargo (por ejemplo, "Desarrollador" o "Gerente"). La clase EmpleadoTiempoParcial debe incluir un atributo para representar las horas trabajadas por semana.

Ambas clases derivadas deben heredar de la clase base Empleado. Implementa métodos específicos en cada clase derivada para calcular el salario total de un empleado. Para EmpleadoTiempoCompleto, el salario total se calcula sumando el salario base más un bono (por ejemplo, 10% del salario base). Para EmpleadoTiempoParcial, el salario total se calcula multiplicando las horas trabajadas por una tarifa por hora.

En el programa principal, crea instancias de ambas clases, establece valores para sus atributos y muestra la información detallada de cada empleado, incluyendo su salario total.

Ejercicio 16: Sistema de Biblioteca

Diseña un sistema de gestión para una biblioteca utilizando principios de Programación Orientada a Objetos (POO) en Java. Debes implementar al menos las siguientes clases:

Libro: Representa un libro con atributos como título, autor, año de publicación, y disponibilidad (si está prestado o no).

Lector: Representa a una persona que puede tomar libros prestados de la biblioteca. Debe tener atributos como nombre, número de identificación, y una lista de libros prestados.

Biblioteca: Representa la biblioteca en sí. Debe contener una lista de libros disponibles, una lista de libros prestados y una lista de lectores registrados. Implementa métodos para prestar un libro a un lector, devolver un libro, mostrar información sobre los libros disponibles y los lectores registrados.

Programa Principal: En el programa principal, crea instancias de libros, lectores y la biblioteca. Realiza algunas operaciones como registrar un lector, agregar libros a la biblioteca, prestar libros a los lectores y mostrar información relevante.

Este ejercicio te permitirá aplicar conceptos de POO como encapsulamiento, herencia y polimorfismo. Además, deberías considerar la relación entre las clases (por ejemplo, un lector puede tener varios libros prestados, un libro puede estar en la biblioteca o prestado a un lector).

Ejercicio 17: Sistema de Autenticación

Crea una interfaz llamada Autenticable que tenga un método autenticar que devuelve un valor booleano. Luego, implementa esta interfaz en clases como Usuario y Administrador. Cada clase debe proporcionar su propia implementación del método autenticar. En la clase principal, crea instancias de estas clases y demuestra la autenticación.

Ejercicio 18: Sistema de Gestión de Empleados

Crea una interfaz llamada Empleado que tenga métodos para calcular el salario y mostrar la información del empleado. Luego, implementa esta interfaz en clases como EmpleadoAsalariado, EmpleadoPorHoras y EmpleadoComision. Cada clase debe tener propiedades específicas, como salario mensual, tarifa por hora, ventas realizadas, etc.

En la clase principal, crea un array o una lista de objetos Empleado que contenga instancias de las clases anteriores. Calcula y muestra el salario de cada empleado y su información.

Ejercicio 19: Cálculos numéricos

Se requiere definir una clase denominada CalculosNumericos que realice las siguientes operaciones:

1. Calcular el logaritmo neperiano recibiendo un valor double como parámetro. Este método debe ser estático. Si el valor no es positivo se genera una excepción aritmética.
2. Calcular la raíz cuadrada recibiendo un valor double como parámetro. Este método debe ser estático. Si el valor no es positivo se genera una excepción aritmética.
3. Se debe crear un método main que utilice dichos métodos ingresando un valor por teclado.
4. Se deben de capturar las excepciones.

Ejercicio 20: Equipo de programadores

Un equipo de programadores desea participar en una maratón de programación. El equipo tiene los siguientes atributos:

- Nombre del equipo.
- Universidad que está representando al equipo.
- Lenguaje de programación que va a utilizar el equipo en la competencia.
- Tamaño del equipo.

Se requiere un constructor que inicialice los atributos del equipo. El equipo está conformado por varios programadores, mínimo dos y máximo tres. Cada programador posee nombre y apellidos. Se requieren además los siguientes métodos:

- Un método para determinar si el equipo está completo.
- Un método para añadir programadores al equipo. Si el equipo está lleno se debe imprimir la excepción correspondiente.
- Un método para validar los atributos nombre y apellidos de un programador para que reciban datos que sean solo texto. Si se reciben datos numéricos se debe generar la excepción correspondiente. Además, no se permiten que los campos String tengan una longitud igual o superior a 20 caracteres.
- En un método main se debe crear un equipo solicitando sus datos por teclado y se validan los nombres y apellidos de los programadores.