mouredev pro

ENUNCIADOS EJERCICIOS CURSO DE JAVA Y POO DESDE CERO

Primeros pasos

1. Imprime un mensaje que diga tu nombre en lugar de "¡Hola Mundo!".

2. Imprime dos líneas: "Hola" y luego "Mundo" con un solo println.

3. Añade un comentario sobre lo que hace cada línea del programa.

4. Crea un comentario en varias líneas.

5. Imprime tu edad, tu color favorito y tu ciudad.

6. Explora los diferentes System.XXX.println(); más allá de "out".

7. Utiliza varios println para imprimir una frase.

8. Imprime un diseño ASCII (por ejemplo, una cara feliz usando símbolos).

9. Intenta ejecutar el programa sin el método main y observa el error.

10. Intenta cambiar el nombre del archivo a uno diferente del de la clase y compílalo. ¿Qué pasa?

Variables y constantes

1. Declara una variable de tipo String y asígnale tu nombre.
2. Crea una variable de tipo int y asígnale tu edad.
3. Crea una variable double con tu altura en metros.
4. Declara una variable de tipo boolean que indique si te gusta programar.
5. Declara una constante con tu email.
6. Crea una variable de tipo char y guárdale tu inicial.
7. Declara una variable de tipo String con tu localidad, y a continuación cambia su valor y vuelve a imprimirla.
8. Crea una variable int llamada a, otra b, e imprime la suma de ambas.
9. Imprime el tipo de dos variables creadas anteriormente.
10. Intenta declarar una variable sin inicializarla y luego asígnale un valor antes de imprimirla.

Operadores

1. Crea una variable con el resultado de cada operación aritmética.

2. Crea una variable para cada tipo de operación de asignación.

3. Imprime 3 comparaciones verdaderas con diferentes operadores de comparación.

4. Imprime 3 comparaciones falsas con diferentes operadores de comparación.

5. Utiliza el operador lógico and.

6. Utiliza el operador lógico or.

7. Combina ambos operadores lógicos.

8. Añade alguna negación.

9. Imprime 3 ejemplos de uso de operadores unarios.

10. Combina operadores aritméticos, de comparación y lógicos.

Strings

1. Concatena dos cadenas de texto.
2. Muestra la longitud de una cadena de texto.
3. Muestra el primer y último carácter de un string.
4. Convierte a mayúsculas y minúsculas un string.
5. Comprueba si una cadena de texto contiene una palabra concreta.
6. Formatea un string con un entero.
7. Elimina los espacios en blanco al principio y final de un string.
8. Sustituye todos los espacios en blanco de un string por un guión (-).
9. Comprueba si dos strings son iguales.
10. Comprueba si dos strings tienen la misma longitud.

Condicionales

1. Establece la edad de un usuario y muestra si puede votar (mayor o igual a 18).

2. Declara dos números y muestra cuál es mayor, o si son iguales.

3. Dado un número, verifica si es positivo, negativo o cero.

4. Crea un programa que diga si un número es par o impar.

5. Verifica si un número está en el rango de 1 a 100.

6. Declara una variable con el día de la semana (1-7) y muestra su nombre con switch.

7. Simula un sistema de notas: muestra "Sobresaliente", "Aprobado" o "Suspenso" según la nota (0-100).

8. Escribe un programa que determine si puedes entrar al cine: debes tener al menos 15 años o ir acompañado.

9. Crea un programa que diga si una letra es vocal o consonante.

10. Usa tres variables a, b, c y muestra cuál es el mayor de las tres.

Estructuras

1. Crea un Array con 5 valores e imprime su longitud.
2. Modifica uno de los valores del Array e imprime el valor del índice antes y después de modificarlo.
3. Crea un ArrayList vacío.
4. Añade 4 valores al ArrayList y elimina uno a continuación.
5. Crea un HashSet con 2 valores diferentes.
6. Añade un nuevo valor repetido y otro sin repetir al HashSet.
7. Elimina uno de los elementos del HashSet.
8. Crea un HashMap donde la clave sea un nombre y el valor el número de teléfono. Añade tres contactos.
9. Modifica uno de los contactos y elimina otro.
10. Dado un Array, transfórmalo en un ArrayList, a continuación en un HashSet y finalmente en un HashMap con clave y valor iguales.

Bucles

1. Imprime los números del 1 al 10 usando while.
2. Usa do-while para mostrar todos los valores de un ArrayList.
3. Imprime los múltiplos de 5 del 1 al 50 usando for.
4. Recorre un Array de 5 números e imprime la suma total.
5. Usa un for para recorrer un Array y mostrar sus valores.
6. Usa for-each para recorrer un HashSet y un HashMap.
7. Imprime los números del 10 al 1 (descendiente) con un bucle for.
8. Usa continue para saltar los múltiplos de 3 del 1 al 20.
9. Usa break para detener un bucle cuando encuentres un número negativo en un array.
10. Crea un programa que calcule el factorial de un número dado.

Funciones

1. Crea una función que imprima "¡Te doy la bienvenida al curso de Java desde cero!".

2. Escribe una función que reciba un nombre como parámetro y salude a esa persona.

3. Haz un método que reciba dos números enteros y devuelva su resta.

4. Crea un método que calcule el cuadrado de un número (n * n).

5. Escribe una función que reciba un número y diga si es par o impar.

6. Crea un método que reciba una edad y retorne true si es mayor de edad (y false en caso contrario).

7. Implementa una función que reciba una cadena y retorne su longitud.

8. Crea un método que reciba un array de enteros, calcula su media y lo retorna.

9. Escribe un método que reciba un número y retorna su factorial.

10. Crea una función que reciba un ArrayList<String> y lo recorra mostrando cada elemento.

POO: Clases y objetos

1. Crea una clase Book con atributos title y author. Crea un objeto y muestra sus datos.

2. Crea una clase Dog con un método bark() que imprima su sonido.

3. Añade un constructor a la clase Book que reciba title y author.

4. Crea una clase Car con atributos brand y model y un método showData().

5. Crea una clase Student con atributo score y un método que diga si aprobó (mayor o igual a 60).

6. Crea una clase BankAccount con atributo balance y un método deposit() que sume el saldo.

7. Crea una clase Rectangle con métodos para calcular el área y el perímetro.

8. Crea una clase Worker que reciba nombre y salario, y un método para mostrar su salario.

9. Crea varios objetos Person y guárdalos en un ArrayList.

10. Crea una clase Product y un método que aplique un descuento sobre su precio.

POO: Modificadores de acceso

1. Crea una clase Person con atributos privados name y age. Usa los métodos getName(), setName(), getAge() y setAge() para asignar y mostrar valores desde otra clase.

2. Crea una clase Product con el atributo privado price. Añade el método setPrice(double price) que solo permita precios mayores a 0.

3. Crea una clase BankAccount con el atributo privado balance. Implementa los métodos deposit(double amount) y withdraw(double amount) que validen las cantidades correctamente.

4. Crea una clase Book con el atributo privado title. Permite leerlo con el método getTitle() pero no modificarlo (sin setTitle()). El título debe asignarse solo por el constructor.

5. Crea una clase Temperature con el atributo privado celsius. El método setCelsius(double celsius) solo debe aceptar valores entre -100 y 100.

6. Crea una clase User con los atributos privados username y password. Implementa los métodos setUsername(String username), setPassword(String password) y checkPassword(String inputPassword) que compare contraseñas.

7. Crea una clase Employee con el atributo privado salary. Agrega el método raiseSalary(double percent) que solo permita aumentos positivos.

8. Crea una clase Rectangle con los atributos privados width y height. Agrega setters y el método calculateArea() que devuelva el resultado de width * height.

9. Crea una clase Student con el atributo privado grade. Agrega los métodos setGrade(int grade) y isPassed() que retorne true si la nota es mayor o igual a 60.

10. Crea una clase Car con el atributo privado speed. Agrega los métodos accelerate(int amount) que aumente la velocidad (máximo 120) y brake(int amount) que reduzca la velocidad (mínimo 0).

POO: Herencia

1. Crea una clase Vehicle con un método move(). Luego crea una subclase Car que herede de Vehicle y agrega el método honk().

2. Define una clase Person con los atributos name y age. Luego crea una clase Student que agregue el atributo grade y un método study().

3. Crea una clase Animal con el método makeSound(). Haz que Dog diga “Woof” y Cat diga “Meow” sobrescribiendo ese método.

4. La clase Employee tiene los atributos name y salary. Manager hereda de Employee y agrega el atributo department.

5. Crea una clase abstracta Shape con un método calculateArea(). Luego implementa ese método en Circle y Rectangle.

6. Crea una clase Bird con el método fly(). Luego crea Eagle que sobrescriba fly() pero también llame al método original con super.fly().

7. Haz una clase Device con un constructor que imprima “Device created”. Luego crea Phone que herede de Device y en su constructor imprima “Phone ready”.

8. Account tiene un saldo y métodos para deposit() y withdraw(). SavingsAccount hereda y agrega un método addInterest().

9. Crea una clase Vehicle y tres subclases: Car, Bike y Truck, cada una con un método describe() sobrescrito.

10. Crea un ArrayList<Animal> que contenga instancias de Dog, Cat y Bird. Recorre la lista y llama a makeSound().

POO: Polimorfismo

1. Crea una clase Animal con el método makeSound(). Luego crea subclases Dog, Cat y Cow que sobrescriban ese método con sonidos diferentes. Llama al método desde una lista de Animal.

2. Crea una clase Shape con el método calculateArea(). Luego implementa subclases Circle y Rectangle con sus propias fórmulas. Usa una lista de Shape para recorrer e imprimir el área de varias figuras.

3. Crea una clase Printer con varios métodos print() sobrecargados que acepten diferentes tipos de parámetros (String, int, double). Llama a cada uno desde main.

4. Crea una clase Greeter con dos métodos greet(): uno que salude con “Hello”, y otro que reciba un nombre y salude con “Hello, [nombre]”.

5. Crea una clase Vehicle con un método start(). Luego crea Car, Bike y Truck que sobrescriban ese método. Recorre una lista ArrayList<Vehicle> para llamar a start() en cada uno.

6. Crea una clase Notification con método send(), y subclases EmailNotification, SMSNotification. Luego crea una función sendNotification(Notification n) que reciba cualquier tipo y lo ejecute.

7. Crea una función showAnimalType(Animal animal) que imprima el tipo de animal. Pasa diferentes subclases (Dog, Cat, Horse) para que cada una imprima su tipo con su propio getType() sobrescrito.

8. Crea una clase Converter con métodos convert(int), convert(double), y convert(String) que devuelvan diferentes formatos de texto.

9. Crea una clase Product con el método getPrice(). Luego, Book y Electronic deben sobrescribirlo con su propia lógica de descuento. Recorre una lista de Product e imprime el precio final de cada uno.

10. Crea una clase Character con método attack(). Luego crea subclases Warrior, Archer, Mage con ataques diferentes. En main, crea un array de Character y llama a attack() para cada uno.

POO: Abstracción

1. Crea una clase abstracta Shape con el método calculateArea(). Luego implementa dos subclases: Circle y Rectangle, y haz que cada una calcule su propia área.

2. Crea una interfaz Playable con el método play(). Luego implementa esa interfaz en dos clases: Guitar y Piano. Cada una debe mostrar un mensaje diferente al ejecutarse.

3. Define una clase abstracta Animal con el método makeSound(). Implementa Dog y Cat para que hagan sonidos distintos. Crea un array de Animal para mostrar polimorfismo.

4. Crea una interfaz Drawable. Implementa las clases Circle, Square, y Triangle que muestren cómo se dibuja cada figura usando draw().

5. Crea una clase abstracta Employee con un método calculateSalary(). Implementa FullTimeEmployee y PartTimeEmployee con lógica diferente para calcular el salario.

6. Crea una interfaz Movable con el método move(). Haz que las clases Car y Robot implementen ese método con comportamientos diferentes.

7. Crea una clase abstracta Appliance con método turnOn() y turnOff. Implementa TV y WashingMachine con mensajes diferentes al encender y apagar.

8. Crea dos interfaces Flyable y Swimmable. Crea una clase Duck que implemente ambas interfaces y muestre cómo puede volar y nadar.

9. Crea una clase abstracta Document con el método print(). Luego crea PDFDocument y WordDocument, cada una con su forma de imprimir.

10. Crea una interfaz Payable con el método pay(). Luego implementa las clases Invoice y EmployeePayment, cada una mostrando un mensaje de pago diferente.

Excepciones

1. Divide dos números almacenados en dos variables. Maneja la división por cero con try-catch.

2. Crea un array de 3 elementos e intenta acceder al índice 5. Captura el ArrayIndexOutOfBoundsException.

3. Crea una variable String nula e intenta imprimir su longitud. Maneja el NullPointerException.

4. Escribe una función que transforma texto a número. Usa try-catch para manejar entradas no válidas (NumberFormatException).

5. Escribe un programa con un bloque finally que se ejecute siempre, haya o no error.

6. Usa throw para lanzar un IllegalArgumentException si un número introducido negativo.

7. Crea una clase TemperatureChecker que lanza una excepción personalizada si la temperatura es menor a -50 o mayor a 50.

8. Crea un programa con varios bloques catch: uno para ArithmeticException, otro para ArrayIndexOutOfBoundsException.

9. Crea una función checkPassword(String pass) que lance una excepción si la contraseña es demasiado corta.

10. Implementa una clase LoginSystem que use una excepción personalizada LoginFailedException si el usuario o contraseña son incorrectos.

Extras

1. Crea una variable de tipo String inicializada como null y verifica que no esté vacía antes de usarla.

2. Escribe un programa que lea el nombre y edad del usuario usando Scanner.

3. Declara una constante final llamada MAX_SCORE con valor 100 y muéstrala.

4. Crea una variable global message y otra local message dentro del método main(). Muestra ambas.

5. Usa import java.util.Scanner; para leer un número y mostrar si es positivo negativo.

6. Declara una variable static en una clase y accede a ella desde main() sin crear un objeto.

7. Importa java.util.Random y genera un número aleatorio del 1 al 10.

8. Crea una clase con comentarios adecuados explicando cada sección del código.

9. Define una clase User con una constante APP_NAME, una variable global username y una función que imprima ambas.

10. Haz debug del código implementado haciendo uso de sus diferentes herramientas.


{m}pro