**Analiza los siguientes Ejemplos**

public class Lampara {

int contador1; // Contador 1

static int contador2; // Contador 2

public Lampara() {

contador1 = 0;

contador2 ++;

}

public void incrementa1() {

contador1++;

}

public void incrementa2() {

contador2++;

}

public static void incrementa() {

contador1++;

contador2++;

}

}

¿Que ocirriría al compilarla?

a.compilaria sin problemas

b. Saldría un error de compilación en el método incrementa1().

c. Saldría un error de compilación en el método incrementa2().

d. Saldría un error de compilación en el método incrementa().

public class ClaseDelPaqueteAcceso {

private int privado = 1;

int paquete = 2;

protected int protegido = 3;

public int publico = 4;

private void metodoPrivado() {

System.out.println("Soy un método privado de la clase ClaseDelPaqueteAcceso.");

}

void metodoDePaquete() {

System.out.println("Soy un método de paquete de la clase ClaseDelPaqueteAcceso.");

}

protected void metodoProtegido() {

System.out.println("Soy un método protegido de la clase ClaseDelPaqueteAcceso.");

}

public void metodoPublico() {

System.out.println("Soy un método público de la clase ClaseDelPaqueteAcceso.");

}

public static void main(String[] args) {

ClaseDelPaqueteAcceso a = new ClaseDelPaqueteAcceso();

System.out.println("Soy un atributo privado de la clase ClaseDelPaqueteAcceso: " + a.privado );

System.out.println("Soy un atributo de paquete de la clase ClaseDelPaqueteAcceso: " + a.paquete );

System.out.println("Soy un atributo protegido de la clase ClaseDelPaqueteAcceso: "+ a.protegido);

System.out.println("Soy un atributo público de la clase ClaseDelPaqueteAcceso: "+ a.publico );

System.out.println("");

a.metodoPrivado();

a.metodoDePaquete();

a.metodoProtegido();

a.metodoPublico();

}

}

**¿Qué imprime el resultado de la ejecución de este código por pantalla?**

**Dado el siguente código Java :** 

class HolaMundo {

String sHola="¡Hola mundo!";

public static void main( String args[] ) {

System.out.println( this.sHola );

}

}

a.El código no compilaría porque el atributo sHola no se puede inicializar al mismo tiempo que su declaración.

b.El código no compilaría porque no hay ningún contructor declarado de la clase.

c.  El código no compilaría porque el método main() está mal declarado.

d. El código no compilaría porque no se puede acceder desde el método estático main() al atributo sHola.

**Escribir los algoritmos correspondientes en Java**

1. Leer un número e indicar si es positivo o negativo. El proceso se repetirá hasta que se introduzca un 0.
2. Pedir números hasta que se teclee uno negativo, y mostrar cuántos números se han introducido.
3. Leer números hasta que se introduzca un 0. Para cada uno indicar si es par o impar.
4. Realizar un juego para adivinar un número. Para ello pedir un número N, y luego ir pidiendo números indicando “mayor” o “menor” según sea mayor o menor con respecto a N. El proceso termina cuando el usuario acierta.
5. Pedir un número N, y mostrar todos los números del 1 al N.
6. Escribir todos los números del 100 al 0 de 7 en 7.
7. Pedir 15 números y escribir la suma total.
8. Pedir 10 sueldos. Mostrar su suma y cuantos hay mayores de 1000€.
9. Dadas las edades y alturas de 5 alumnos, mostrar la edad y la estatura media, la cantidad de alumnos mayores de 18 años, y la cantidad de alumnos que miden más de 1.75.
10. Pide un número (que debe estar entre 0 y 10) y mostrar la tabla de multiplicar de dicho número.
11. . Una empresa que se dedica a la venta de desinfectantes necesita un programa para gestionar las facturas. En cada factura figura: el código del artículo, la cantidad vendida en litros y el precio por litro. Se pide de 5 facturas introducidas: Facturación total, cantidad en litros vendidos del artículo 1 y cuantas facturas se emitieron de más de 600 €.
12. Leer 5 números y mostrarlos en el mismo orden introducido.

13.1

Crear una clase abstracta denominada Animal, de la cual derivan las clases Perro, Gato y Vaca. Todas las subclases deben redefinir el método abstracto hablar(), de la clase Animal, según como “hable” cada uno. Para ello puede declarar el método como de tipo de retorno void y abstracto y luego utilizar el método println de System.out en cada implementación del mismo. Además realice una clase que contenta el método main() denominada Principal que invoque a un método denominado hablarAnimal(Animal a) que reciba un “Animal” como parámetro, y que le permita “hablar”, polimórficamente, según sea Gato, Perro o Vaca.

13.2 Modifique el método main utilizado en el ejercicio anterior, invocando a otro método denominado comerAnimal que reciba como parámetro a un Animal e invoque al correspondiente método para comer (comerCarne o comerHierba) según el tipo de animal que sea.

13.3 Crear un Arraylist y almacenar diferentes cantidades de objetos: perros, gatos y vacas, desarrollar un método que se llame contarAnimales, en donde permita pasarle el arraylist generado previamente y que devuelva la cantidad de perros, gatos y vacas almacenados.