TEMA 5: PROGRAMACIÓN ORIENTADA A OBJETOS (POO)

Práctica 1 - Clases y Objeto Enero

En la teoría se han visto los conceptos fundamentales de programación orientada a objetos. Ya sabes lo que es una clase en Java. Además, se explicaron en temas anteriores los tipos primitivos del lenguaje y los operadores. En esta práctica se van a afianzar estos conceptos.

Crearemos dos ficheros: Alumno.java y PruebaAlumno.java.

- **Alumno.java** : Representa la clase Alumno que iremos mejorando sucesivamente en los diferentes ejercicios añadiendo atributos, métodos y constructores.
- **PruebaAlumno.java** : Es una clase de prueba que únicamente contiene el método main. Sirve para crear objetos de la clase Alumno y ver como se comportan los diferentes tipos de métodos y modificadores de atributos.
- Para probar una clase creamos un objeto de dicha clase, accedemos a sus atributos y métodos e imprime el resultado en pantalla.

EJERCICIO 1 : Clases y Creación de Objetos

Escribe un programa en Java para probar la clase Alumno.

a. El fichero Alumno.java contendrá la descripción de la clase Alumno.

Definición de la clase:

```
class Alumno {
  String nombre;
  String apellidos;
  int añoDeNacimiento;
  int nUmeroPersonal;
  String grupo;
  char horario; //'M': mañana, 'T': tarde
}
```

Una vez escrita la clase Alumno comprueba que compila.

b. El fichero PruebaAlumno.java que contendrá el método main() con las pruebas que queremos realizar.

El método main debe hacer lo siguiente:

- Crear dos objetos de la clase Alumno.
- Dar valor a sus atributos con los datos de los dos alumnos.
- Imprimir los datos correctamente tabulados como se presentan a continuación.

Datos del alumno

==========

Nombre:	Juan
Apellidos:	García Fernández
Año de Nacimiento:	1985
Número Personal:	1020034569
Grupo:	7031-91
Horario:	M

Recuerda que para acceder a los atributos y a los métodos de una clase se emplea el operador punto ".". Por ejemplo: alum1.nombre.

EJERCICIO 2: Clases y Objetos

Añade a la clase Alumno los métodos necesarios para dar valor a estos atributos y visualizar su contenido desde la clase PruebaAlumno.

Recuerda que los métodos se definen dentro del cuerpo de la clase. Como ejemplo, el método para rellenar el numeroPersonal sería algo así:

```
public void ponNumeroPersonal (int n){
     númeroPersonal= n ;
  }
```

Como este método no devuelve ningún valor, no hace falta que tenga una sentencia de **return** dentro del cuerpo del método. Los métodos para leer los campos, en cambio, sí que tienen que devolver un valor (el atributo pedido). Debéis tener cuidado en definir las funciones de forma que el tipo devuelto por el método (lo que en el ejemplo anterior era void) coincida con el tipo del atributo que se devuelve. Por ejemplo, el método para obtener el añoDeNacimiento del alumno sería:

```
public void ponNumeroPersonal (int númeroPersonal){
     this.númeroPersonal= númeroPersonal ;
  }
```

```
public int dameAñoDeNacimiento () {
  return añoDeNacimiento;
}
```

Modifica el main anterior para que utilice los métodos definidos para rellenar estos campos, y comprueba que imprime el resultado correcto.

EJERCICIO 3: Operadores

Añade a la clase Alumno un nuevo campo de tipo float que represente la nota de selectividad (p.e. lo podemos llamar notaSelectividad)

Modificad el programa anterior para que, además de lo anterior, el programa calcule la media de selectividad de la pareja de alumnos e imprima un mensaje en pantalla:

Media de selectividad de la pareja: valor

En la expresión usada en la solución para el cálculo de la media se han sumado las dos notas de selectividad y se han dividido por 2.0f

Los datos de los alumnos se imprimen para cada alumno por separado, crear un metodo llamado imprimirAlumno en el fichero Alumno3.java para no repetirlo

EJERCICIO 4 : Referencias, Alias y Comparación de Objetos

Este ejercicio muestra la **diferencia entre dos referencias iguales y dos objetos iguales**.

Se trata de modificar el programa donde se encuentra el main y crear los objetos de la clase Alumno que se observan en la siguiente figura:

```
Alumno alumno1;  //valen null por defecto
Alumno alumno4;
Alumno alumno5;
Alumno delegado;

alumno1 = new Alumno();
delegado = alumno1;
alumno1.ponNombre("Juan");

alumno4 = new Alumno();
alumno4.ponNombre("Clara");

alumno5 = new Alumno();
alumno5.ponNombre("Juan");
```

Nota: el resto de los atributos que no aparecen en la figura no hace falta rellenarlos.

A continuación, escribir en pantalla el valor de las siguientes expresiones lógicas (que usan operadores de relación):

1. (alumno1 == delegado)
2. (alumno1 == alumno5)
3. (alumno5 == alumno4)
4. (alumno5 == alumno1)

¿Sabéis explicar el resultado que imprime el programa?

EJERCICIO 5 : Constructores

Hasta ahora no hemos escrito ningún constructor para las clases Alumno de los ejercicios anteriores.

Así, después de crear un objeto de la clase Alumno, hemos llamado a los métodos ponNombre, ponApellidos, etc. para asignar valor a los atributos del objeto creado.

Esto tiene el inconveniente que uno puede olvidarse en el programa de rellenar algún atributo lo que dejaría un objeto Alumno con valores inconsistentes.

Como habéis visto en teoría, los constructores son el mecanismo que existe en Java para inicializar un objeto en el momento de su creación.

En este ejercicio debéis escribir un constructor para una clase Alumno5 que rellene todos los atributos del objeto.

Para ello, debéis pasarle al constructor una lista de parámetros para rellenar todos los atributos del objeto.

Esto nos permitirá llamar al constructor de la siguiente forma:

```
new Alumno5("Juan","García Fernández",1985,1020034569,"7031-91",'M',5.6f)
```

para crear un objeto de la clase Alumno5.