**Práctica # 10 Polimorfismo II**

Nombre: GIAN MOLINA RAIGOZA Matrícula:1636155 Calificación:\_\_\_\_\_\_  
 Ponderación: 10% Día: \_\_\_\_\_\_\_\_\_\_\_\_ Hora: \_\_\_\_\_\_:\_\_\_\_\_\_

**Objetivos Generales:**

* Reforzar el conocimiento del concepto de herencia y polimorfismo.
* Lograr inicializar objetos con constructores.

**Actividad #1 – Polimorfismo**

**Ponderación:** 100%

Se pedirá construir un sistema que funcione como administrador de una escuela de idiomas. Tendremos una clase padre que es Entidad, y las clases hijas Alumno y Profesor. A demás se creará una cuarta clase que **no** heredará de Entidad y será llamada Clase

**Procedimiento**

1. Cree un programa llamado AdministradorEscuela.java aquí va a radicar nuestro método main.
2. Cree una clase llamada Entidad.java:

|  |
| --- |
| **Entidad** |
| * String nombre; * double pagoSalarioBase; * Clase clases[]; * int numClases; * final int numClasesMax = 5; |
| public Entidad(  String nombre,  double pagoSalarioBase  )  + métodos get y set de atributos  + aniadirClase(Clase c)  + calcularPagoSalario()  + printEntidad() |

1. El atributo clases será un arreglo de tipo Clase para almacenar hasta un máximo de 5 clases.
2. numClases representara la cantidad de clases guardadas, se inicia en 0
3. numClasesMax representa el máximo de clases que se pueden almacenar, este valor es constante.
4. La función aniadirClase añadirá una Clase al arreglo clases, validando que no se exceda el limite de clases establecido.
5. La función calcularPagoSalario, devolverá el pago o Salario especificado con la siguiente formula .
6. La función printEntidad() mostrara en consola todos los atributos de la clase Entidad.
7. Cree una clase llamada Alumno.java que herede de la clase Entidad.java con las siguientes especificaciones:

|  |
| --- |
| **Alumno** |
| * int anio; |
| + public Alumno(  String nombre,  double pagoSalarioBase,  int anio  )  + métodos get y set de atributos  + calcularPagoSalario()  + printEntidad() |

1. Utilizar la función super() para llamar al constructor de la clase padre y pasarle los parámetros necesarios.
2. La función calcularPagoSalario sobre escribirá la función de la clase padre y devolverá el Pago especificado con la siguiente formula

Si el Alumno lleva mas de 2 años en la escuela, en caso contrario se deberá de llamar al método de la clase padre y ejecutar ese método.

1. La función printEntidad() mostrara en consola todos los atributos de la clase Entidad y los de la clase Alumno.
2. Cree una clase llamada Profesor.java que herede de la clase Entidad.java con las siguientes especificaciones:

|  |
| --- |
| **Profesor** |
| * String tipo; * String idiomas[]; * final int maxIdiomas = 5; |
| + public Profesor(  String nombre,  double pagoSalarioBase,  String tipo  )  + métodos get y set de atributos  + calcularPagoSalario()  + printEntidad() |

1. Utilizar la función super() para llamar al constructor de la clase padre y pasarle los parámetros necesarios.
2. a función calcularPagoSalario sobre escribirá la función de la clase padre y devolverá el Salario especificado. Si el tipo de Profesor es ‘Planta’ se devolverá el pagoSalarioBase solamente y en caso de ser ‘Clase’ se llamará al método de la clase padre y se devolverá ese valor.
3. Cree una clase llamada Clase.java con las siguientes especificaciones:

|  |
| --- |
| **Clase** |
| * String idioma; * String dias; |
| + public Profesor(  String idioma,  String dias  )  + métodos get y set de atributos |

1. En la clase en donde se tiene el método main se declararán 2 objetos uno Profesor y uno Alumno.
2. Se introducirán los datos base de los constructores de los dos objetos para instanciarlos.
3. En la clase en donde se tiene el método main se seguirá el siguiente flujo de procedimiento. El objetivo es estar desplegando un menú de opciones:
4. Añadir clase a Alumno
   1. Se le añadirá una Clase al Alumno
5. Añadir clase a Profesor
   1. Se le añadirá una Clase al Profesor, verificar que el profesor conozca ese idioma.
6. Imprimir Alumno
   1. Se imprimirán los datos del Alumno con el método correspondiente.
7. Imprimir Profesor
   1. Se imprimirán los datos del Profesor con el método correspondiente.
8. Añadir idioma a profesor.
   1. Se le añadirá un idioma al arreglo de idiomas del profesor.
9. Salir