|  |  |  |
| --- | --- | --- |
| Imagen que contiene dibujo, taza  Descripción generada automáticamente | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | José Antonio Ayala Barbosa |
| *Asignatura:* | programación orientada a objetos |
| *Grupo:* | Gpo 1 22-2 |
| *No de Práctica(s):* | 05 |
| *Integrante(s):* | Rosillo Montijo Emmanuel Alonso |
| *No. de Equipo de cómputo empleado:* |  |
| *No. de Lista o Brigada:* |  |
| *Semestre:* | 03 |
| *Fecha de entrega:* | 17/02/2022 |
| *Observaciones:* |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

**Objetivo**

Implementar los conceptos de herencia en un lenguaje de programación orientado a objetos.

**Introducción**

Una definición de herencia: “La herencia es un mecanismo que permite la definición de una clase a partir de la definición de otra ya existente. La herencia permite compartir automáticamente métodos y datos entre clases, subclases y objetos”, para contextualizar, la herencia son los atributos y métodos de otras clases que pueden ser usados en las clases hijas.

**Desarrollo**

Al empezar a abordar la práctica, podemos partir las secciones de su elaboración en las siguientes secciones.

Primero se creó una clase empleado

Con los atributos:

Texto

Descripción generada automáticamente

Contine los clásicos getters y setters de las clases.

Y un método que se implementó: Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Y también contiene el clásico toString.

Luego la clase gerente que hereda de empleados. Esto se puede observar en el método constructor lleno: Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

contiene un atributo heredado indicado por “super”, se encarga de recibir los parámetros de un objeto empleado y utilizarlo como herencia dentro de la clase gerente, gerente contiene de la misma forma los clásicos setters, getters y toString

la clase gerente puede utilizar los atributos de empleado, en el menú principal se puede observar el ejemplo

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

Con el método que utiliza los atributos de herencia anterior, logra modificar los atributos de la clase empleado.

Y devuelve como resultado compuesto de la clase gerente y empleado

Texto

Descripción generada automáticamente

Las instancias se pueden observar con método *instanceOf*

Texto

Descripción generada automáticamente

Para poder observar a que instancia pertenece cada uno.

Luego como actividades extra se generaron 11 clases, cada una generarda con diferentes métodos y atributos, pero en común la herencia que posee cada una, para poder observar estás clases, al ser bastantes, se creó un **UML** para poder observarlas gráficamente y optimizar así el tiempo y el espacio de la práctica, de esta forma.



Podemos observar sus métodos y atributos de cada clase, cuales son sus respectivos padres, el tipo de encapsulamiento y sus respectivas instancias

Cada uno tiene un método en el constructor de herencia, como ejemplo: Texto, Chat o mensaje de texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Y su instancia:

Texto

Descripción generada automáticamente

**Conclusiones:**

La herencia en la programación orientada objetos, facilita la manera de realizar un código más complejo, debido a que no existe la necesidad de copiar clases para ser utilizadas en otras y de esta forma, tener un programa más entendible y corto.

**Referencias:**

Alex Walton. (agosto 12, 2020).

Herencia en Java: Tipos y Ejemplos.

Hoy, de Java desde cero Sitio web: https://javadesdecero.es/poo/herencia-java-tipos-ejemplos/