**Informe**

**Curso:** Programación orientada a objetos  
**Profesor:** Luis Enrique Jaramillo Montaño  
**NRC:** 1323

**I. Portada**

1. **Título del Proyecto:**
   * Informe sobre el Control de Lectura 2
2. **Miembros:**
   * Erick Sebastián Ramírez Tasiguano
3. **Fecha de Entrega:**
   * Miércoles, 27 de noviembre de 2024, 00:00

**II. Índice**

1. Objetivos del Proyecto
2. Recursos usados en el Proyecto
3. Desarrollo del Proyecto
4. Diagrama UML
5. Objetos en código
6. Bibliografía

**III. Objetivos del Proyecto**

* **Objetivo General:**
  + Diseñar 5 objetos diferentes con su correspondiente diagrama UML y código, asegurándose de mostrar las relaciones entre ellos.
* **Objetivos Específicos:**
  + Diseñar 5 objetos diferentes que se relacionen entre ellos.
  + Crear un diagrama UML a partir de los 5 objetos diseñados.
  + Representar los 5 objetos en un código de Java
  + Explicar cómo se relacionan los objetos entre ellos.
  + Determinar qué tipo de relación tienen los objetos entre ellos.

**IV. Recursos usados en el Proyecto**

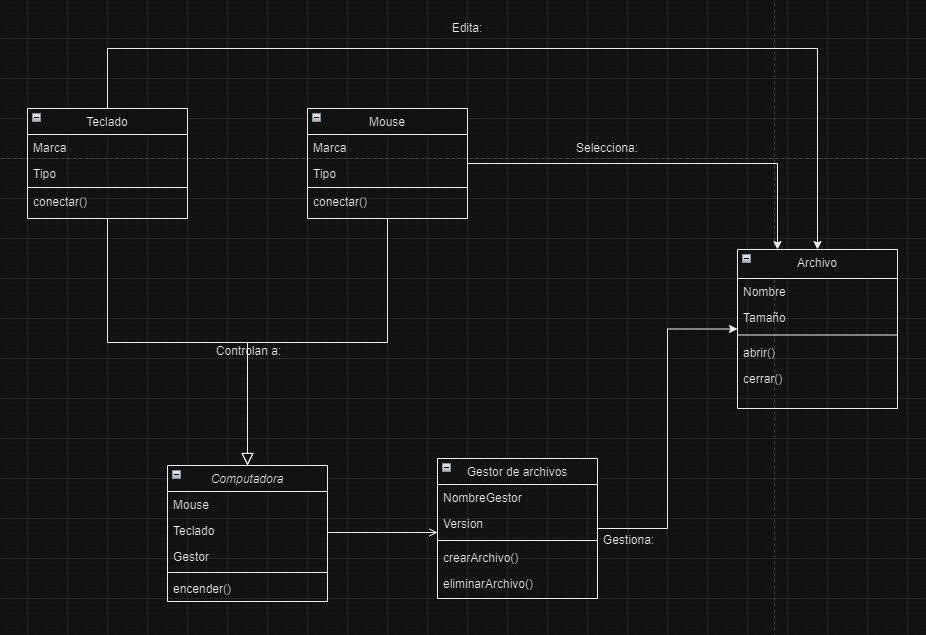
* **Recursos:**
  + Microsoft Word
  + Navegador de internet
  + Creador de diagramas UML ‘‘draw.io’’
  + Página web ‘‘www.onlinegdb.com’’

**V. Desarrollo del Proyecto**

* **Resumen del proyecto:**
  + El primer paso es determinar objetos, siendo muy importante que tengan una relación entre dichos objetos.
  + El primer objeto determinado fue el de ‘‘Computadora’’.
  + A partir del objeto ‘‘Computadora’’ se pueden determinar los demás objetos.
  + Se determinan 4 objetos que tienen una relación con computadora: Teclado, Mouse, Gestor de Archivos y Archivo.
  + Esto nos da un total de 5 objetos: Computadora, Teclado, Mouse, Gestor de Archivos, y Archivo.
  + Una vez determinados, ahora debemos asignar atributos y métodos a los objetos para formar el diagrama UML, dichos atributos y métodos deben guardar relación con los otros objetos.
  + Ya determinados los atributos y métodos de cada objeto, debemos relacionarlos entre ellos en el espacio del diagrama.
  + Completados los anteriores pasos, el diagrama UML está ahora completo.
  + A partir del diagrama UML, ahora se procede a representar estos objetos en forma de código Java.
  + Cada uno de los objetos tiene su propio código, al igual que en el diagrama UML, el código que se diseñara debe demostrar la relación que tienen los objetos entre ellos.
  + Una vez terminado el código de cada uno de los objetos en Java, se procede a la explicación de cómo se relacionan los objetos entre ellos, y su tipo de relación.

**VI. Diagrama UML**

* **Diagrama UML de objetos:**
  + En esta sección, se mostrarán a continuación los cinco objetos que se diseñaron, representados en un diagrama UML:



**VII. Objetos en código**

* En esta sección, se mostrarán a continuación los cinco objetos, representados en forma de un código Java:

**import java.util.ArrayList;**

**import java.util.List;**

**class Computadora {**

**public Mouse mouse;**

**public Teclado teclado;**

**public GestorDeArchivos gestor;**

**public Computadora(Mouse mouse, Teclado teclado, GestorDeArchivos gestor) {**

**this.mouse = mouse;**

**this.teclado = teclado;**

**this.gestor = gestor;**

**}**

**public void encender() {**

**System.out.println("Computadora encendida.");**

**}**

**}**

**class Mouse {**

**public String Marca;**

**public String Tipo;**

**public Mouse(int id, String tipo) {**

**this.Marca = Marca;**

**this.Tipo = Tipo;**

**}**

**public void conectar() {**

**System.out.println("Mouse detectado");**

**}**

**}**

**class Teclado {**

**public String Marca;**

**public String Tipo;**

**public Teclado(int id, String tipo) {**

**this.Marca = Marca;**

**this.Tipo = Tipo;**

**}**

**public void conectar() {**

**System.out.println("Teclado detectado");**

**}**

**}**

**class GestorDeArchivos {**

**public List<Archivo> archivos;**

**public GestorDeArchivos() {**

**this.archivos = new ArrayList<>();**

**}**

**public void agregarArchivo(Archivo archivo) {**

**archivos.add(archivo);**

**System.out.println("Archivo agregado: " + archivo.getNombre());**

**}**

**public void eliminarArchivo(String nombre) {**

**archivos.removeIf(archivo -> archivo.getNombre().equals(nombre));**

**System.out.println("Archivo eliminado: " + nombre);**

**}**

**}**

**class Archivo {**

**public String nombre;**

**public int tamaño;**

**public Archivo(String nombre, int tamaño) {**

**this.nombre = nombre;**

**this.tamaño = tamaño;**

**}**

**public String getNombre() {**

**return nombre;**

**}**

**public void abrir() {**

**System.out.println("Archivo abierto: " + nombre);**

**}**

**public void cerrar() {**

**System.out.println("Archivo cerrado: " + nombre);**

**}**

**}**

**public class Main {**

**public static void main(String[] args) {**

**Mouse mouse = new Mouse(1, "Conectado");**

**Teclado teclado = new Teclado(1, "Conectado");**

**GestorDeArchivos gestor = new GestorDeArchivos();**

**Computadora computadora = new Computadora(mouse, teclado, gestor);**

**computadora.encender();**

**mouse.conectar();**

**teclado.conectar();**

**Archivo archivo1 = new Archivo("Archivo.txt", 1024);**

**gestor.agregarArchivo(archivo1);**

**archivo1.abrir();**

**archivo1.cerrar();**

**gestor.eliminarArchivo("Archivo.txt");**

**}**

**}**

**VII. Referencias**

* **Bibliografía:**
  + Admin. (2022, November 24). ¿Cuáles son los seis tipos de relaciones en los diagramas de clases UML? - Visual Paradigm Blog español. Visual Paradigm Blog Español. <https://blog.visual-paradigm.com/es/what-are-the-six-types-of-relationships-in-uml-class-diagrams/>