ADRIAN MAXIMILIANO MUNOZ MARTINEZ

Laboratorio POO  Sabado 7 am – 9am

PRactica 8: Herencia y Polimorfisomo

**Introducción**

Se desarrollará una aplicación en Java que en la que se implemente herencia de clases abstractas e interfaces, así como algunas características del polimorfismo.

**Objetivo**

El objetivo de la practica es el de reforzar los conceptos de herencia y polimorfismo.

**Practica**

Practica8.java

package practica8;

import java.util.Scanner;

import practica8.DispositivoElectronico;

public class Practica8 {

public static void main(String[] args) {

int i;

Scanner sc = new Scanner(System.in);

// Crear objeto telefono y configurar con los atributos senalados

Telefono tel = new Telefono();

System.out.println("\nCreando telefono: ");

System.out.print("Ingresa la marca del telefono: ");

tel.setMarca(sc.nextLine());

System.out.print("Ingresa el modelo del telefono: ");

tel.setModelo(sc.nextLine());

System.out.print("Ingresa el color del telefono: ");

tel.setColor(sc.nextLine());

System.out.print("Ingresa el precio del telefono: ");

tel.setPrecio(sc.nextFloat());

// Limpiar buffer

sc.nextLine();

// Crear arreglo de contactos con la informacion ingresada por el usuario

System.out.println("\nCrear contactos:");

Contacto[] contactos = new Contacto[5];

for (i = 0; i < 0; i++) {

contactos[i] = new Contacto();

System.out.println(String.format("\nContacto %d:", i + 1));

System.out.print("Telefono: ");

contactos[i].setTelefono(sc.nextLine());

System.out.print("Nombre: ");

contactos[i].setNombre(sc.nextLine());

System.out.print("Correo electronico: ");

contactos[i].setMail(sc.nextLine());

}

System.out.println("\n");

System.out.println(tel.encender());

// Pedir por la contrasena de la red inalambrica

System.out.println("Ingrese la contrasena de la red wifi: ");

System.out.println("(hint)a=b abc");

System.out.println(tel.wifiCon(sc.nextLine()));

// Realizar una llamada a un contacto si el usario lo desea, a un numero

// especifico de lo contrario

System.out.println("\nDesea llamar a alguien de su lista de contactos?(s/n): ");

if (sc.nextLine().startsWith("s")) {

System.out.println("Eliga un contacto:");

i = 0;

for (Contacto contact : contactos) {

System.out.println(String.format("%d %s", ++i, contact.getNombre()));

}

do {

i = sc.nextInt() - 1;

} while (i < 0 || i > 4);

System.out.println(tel.iniciar\_llamada(contactos[i].getNombre()));

} else {

System.out.println("Escriba el numero que desea llamar: ");

System.out.println(tel.iniciar\_llamada(sc.nextInt()));

}

System.out.println(tel.finalizar\_llamada());

tel.apagar();

// Limpiar buffer

sc.nextLine();

// Crear nuevo telefono usando las caracteristicas del polimorfismo

System.out.println("\nNuevo telefono: ");

DispositivoElectronico ntel = new Telefono();

System.out.println("Ingresa la marca del telefono: ");

ntel.setMarca(sc.nextLine());

System.out.println("Ingresa el modelo del telefono: ");

ntel.setModelo(sc.nextLine());

System.out.println("Informacion del telefono:");

System.out.println(String.format("Marca: %s", ntel.getMarca()));

System.out.println(String.format("Modelo: %s", ntel.getModelo()));

System.out.println(ntel.encender());

System.out.println(ntel.apagar());

}

}

WifiConection.java

package practica8;

public interface WifiConexion {

public String wifiCon(String pswd);

}

DispositivoElectronico.java

package practica8;

public abstract class DispositivoElectronico {

private String marca;

private String modelo;

public abstract String encender();

public abstract String apagar();

public void setMarca(String marca) {

this.marca = marca;

}

public String getMarca() {

return this.marca;

}

public void setModelo(String modelo) {

this.modelo = modelo;

}

public String getModelo() {

return this.modelo;

}

}

Telefono.java

package practica8;

import practica8.DispositivoElectronico;

import practica8.WifiConexion;

public class Telefono extends DispositivoElectronico implements WifiConexion {

private float precio;

private String color;

@Override

public String encender() {

return "Telefono encendido";

}

@Override

public String apagar() {

return "Telefono apagado";

}

public String wifiCon(String pswd) {

// Establezer como contrasena correcta la cadena bcd

if (pswd.equals("bcd")) {

return "Contrasena correcta, conetado a la red WiFi";

}

return "Contrasena incorrecta";

}

public String iniciar\_llamada(int num) {

return String.format("Llamando a %d", num);

}

public String iniciar\_llamada(String name) {

return String.format("Llamando a %s", name);

}

public String finalizar\_llamada() {

return "Llamada finalizada";

}

public void setPrecio(float precio) {

this.precio = precio;

}

public float getPrecio() {

return this.precio;

}

public void setColor(String color) {

this.color = color;

}

public String getColor() {

return this.color;

}

}

Contacto.java

package practica8;

public class Contacto {

private String nombre;

private String telefono;

private String passwd;

private String mail;

public void setNombre(String nombre) {

this.nombre = nombre;

}

public String getNombre() {

return this.nombre;

}

public void setTelefono(String telefono) {

this.telefono = telefono;

}

public String getTelefono() {

return this.telefono;

}

public void setPasswd(String passwd) {

this.passwd = passwd;

}

public String getPasswd() {

return this.passwd;

}

public void setMail(String mail) {

this.mail = mail;

}

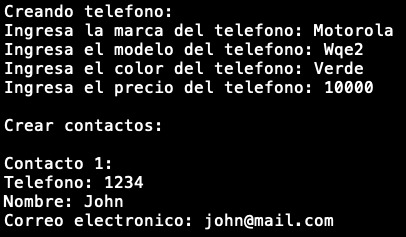
public String getMail() {

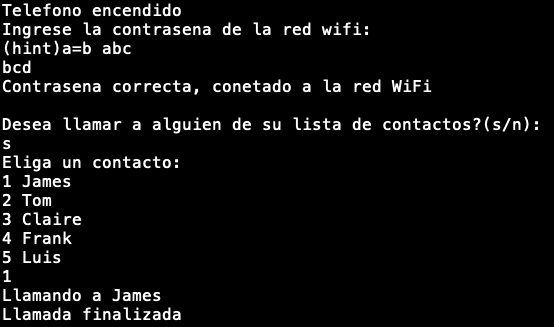
return this.mail;

}

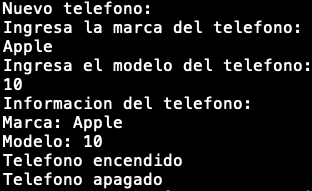
}

**Screenshots**

****

****

**Imagen que contiene negro, televisión, pantalla, ciudad

Descripción generada automáticamente**

**Explicación**

Se creo un aplicación que implemento la clase Telefono, la cual heredo parte se funcionalidad de la interfaz WifiConection y de la clase abstracta DispositivosElectronicos

Se le asignaron ciertos atributos al objeto teléfono basándose en la información que ingreso el usuario, se encendió y se conecto a la red WiFi, después se creo un arreglo de 5 contactos para finalmente simular una llamada y apagar el teléfono.

Con esta actividad pudimos hacer uso herencia.

Al final se creo un objeto de tipo DispositivoElectronico llamando al constructor de Telefono y se ejecutaron todos los métodos y se asignaron todos los atributos disponibles.

Con esta ultima actividad pudimos hacer uso del polimorfismo.

**Conclusión**

La herencia y el polimorfismo nos permite crear código mas versátil y mas fácil de mantener.

Mientras el polimorfismo nos permite manipular los datos y los objetos de una manera mas dinámica, la herencia nos permite heredar las características de clases creadas previamente y nos libra de la necesidad de tener que implementar todo nuevamente para añadir una nueva funcionalidad.