ADRIAN MAXIMILIANO MUNOZ MARTINEZ

Laboratorio POO  Sabado 7 am – 9am

PRactica 9i: Java COllection

**Introducción**

Se desarrollará una aplicación en Java que en la que se implemente colecciones

**Objetivo**

El objetivo de la practica es familiarizarnos con el concepto de collection, así como practicar su implementación y conocer sus ventajas y desventajas

**Practica**

Practica9i.java

package practica9i;

import java.util.ArrayList;

import java.util.Scanner;

public class Practica9i {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int i, j;

String[] propiedades = { "Nombre", "Gamma", "Color" };

String[] elements = new String[3];

ArrayList telefonos = new ArrayList();

for (i = 0; i < 5; i++) {

Movil nuevo\_movil = new Movil();

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

System.out.print("\tIngresa el nombre: ");

elements[0] = sc.nextLine();

System.out.print("\tIngresa el precio: ");

nuevo\_movil.setPrecio(sc.nextInt());

sc.nextLine();

System.out.print("\tIngresa el modelo: ");

nuevo\_movil.setModelo(sc.nextLine());

System.out.print("\tIngresa la gamma: ");

elements[1] = sc.nextLine();

System.out.print("\tIngresa el color: ");

elements[2] = sc.nextLine();

System.out.print("\tEstas son las propiedades del telefono: ");

j = 0;

for (String element : nuevo\_movil.propiedades(elements)) {

System.out.print(String.format("\n\t%s: %s", propiedades[j], element));

j++;

}

System.out.println();

telefonos.add(nuevo\_movil);

}

// Imprimir los atributos y llamar las funciones de cada objeto Movil en el

// array list

j = 0;

for (Object objectMovil : telefonos) {

// Convertir el elemento del array a tipo movil, y guardar la referencia en una

// nueva variable

// De esta manera Java va a saber de que tipo de objeto estamos hablando

Movil movil = (Movil) objectMovil;

System.out.println(String.format("\nTelefono %d", ++j));

System.out.println(String.format("Precio del telefono %d", movil.getPrecio()));

System.out.println(String.format("Modelo del telefono %s", movil.getModelo()));

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

System.out.print("Ingresa el numero al que deseas marcar: ");

movil.hacerLlamada(sc.nextInt());

sc.nextLine();

movil.finalizarLlamada();

System.out.print("Ingresa el nombre del contacto al que deseas marcar: ");

movil.hacerLlamada(sc.nextLine());

movil.finalizarLlamada();

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

}

}

}

DispositivoElectronico.java

package practica9i;

public interface DispositivoElectronico {

public abstract String encender();

public abstract String apagar();

}

Telefono.java

package practica9i;

public abstract class Telefono {

private String modelo;

public abstract String[] propiedades(String[] element);

public String getModelo() {

return modelo;

}

public void setModelo(String modelo) {

this.modelo = modelo;

}

}

Movil.java

package practica9i;

public class Movil extends Telefono implements DispositivoElectronico {

private int precio;

@Override

public String encender() {

return "El telefono se ha encendido";

}

@Override

public String apagar() {

return "El telefono se ha apagado";

}

public void hacerLlamada(int tel) {

System.out.println(String.format("Llamando a %d", tel));

}

public void hacerLlamada(String contacto) {

System.out.println(String.format("Llamando a %s", contacto));

}

public void finalizarLlamada() {

System.out.println("Finalizando llamada");

}

public void setPrecio(int precio) {

this.precio = precio;

}

public int getPrecio() {

return this.precio;

}

@Override

public String[] propiedades(String[] elemnts) {

return elemnts;

}

}

**Screenshots**

**Imagen que contiene texto

Descripción generada automáticamente**

**Explicación**

Se creo una aplicación que implemento un objeto Movil. Dentro de la clase principal (Practica9i) se creo un ArrayList cuya función era la de almacenar los objetos de tipo Movil que fueron creados y configurados con la entrada del usuario.

Para acceder a esos objetos fue necesario castear cada uno de los elementos del ArrayList y guardar la referencia del objeto casteado en una variable de tipo Movil. Se hizo esto ya que Java no sabia que tipo de elementos se encontraban en el arreglo y por lo tanto no nos permitía acceder a las funciones de la clase Movil, porque hasta donde Java sabia dentro del ArrayList solo había objetos, pero no sabia de que tipo.

Por eso fue necesario castear y almacenar el resultado en una variable tipo Movil, para que Java supiera que tipo de objeto era determinado elemento del ArrayList y pudiéramos acceder a las funciones de Movil.

**Conclusión**

Las colecciones en Java son muy útiles ya que nos permiten almacenar información y objetos de una manera indeterminada, ya sea en tipo o numero. También ofrecen funciones que nos permiten manipular y obtener la información dentro del array muy fácilmente, siempre y cuando le especifiquemos el tipo de dato que se esta utilizando.