Enlace reunión #1:

https://unabedu-my.sharepoint.com/:v:/g/personal/karen_fandino_o365_unab_edu_co/EQzjMqmEBqNFtyXXgE6TWv8BRtirbXS-HSx54YUuobJO8A

Patrón singleton:

El propósito de este patrón es evitar que sea creado más de un objeto por clase, se crea el objeto deseado de una clase y se recupera con una instancia estática.

Un ejemplo de este patrón es el gobierno, un país solo puede tener un gobierno oficial.

Otro ejemplo un servidor necesita ser representado por un objeto, esta debería ser único, es decir solo debería existir una sola instancia y el resto de clases deberían comunicarse con el mismo servidor.

```
public class ClaseSingleton {
   private static ClaseSingleton instanciaUnica = new ClaseSingleton();
   private ClaseSingleton() {}
   public static ClaseSingleton getInstance() {
      return instanciaUnica;
   }
}
```

Patron prototype:

Nos permite copiar objetos existentes sin que el código dependa de sus clases.

Se crea un grupo de objetos configurados de maneras diferentes. Cuando necesites un objeto como el que has configurado, clonas un prototipo en lugar de construir un nuevo objeto desde cero.

En el mundo real los prototipos se utilizan para crear pruebas de todo tipo antes de comenzar con la producción en masa de un producto.

También podemos crear la clonación completa de un listado de productos para crear otros derivados.

```
// Los productos deben implementar esta interface
public interface Producto extends Cloneable {
   Object clone();
   // Aquí van todas las operaciones comunes a los productos que genera la factoría
}

// Un ejemplo básico de producto
public class UnProducto implements Producto {
   private int atributo;

   public UnProducto(int atributo) {
      this.atributo = atributo;
   }

   public Object clone() {
      return new UnProducto(this.atributo);
   }

   public String toString() {
      return ((Integer)atributo).toString();
   }
}

// La clase encargada de generar objetos a partir de los prototipos
public class FactoriaPrototipo {
   private HashMap mapaObjetos;
   private String nombrePorDefecto;

   public FactoriaPrototipo() {
      mapaObjetos = new HashMap();
      // Se incluyen en el mapa todos los productos prototipo
```

```
        mapaObjetos.put("producto 1", new UnProducto(1));
    }

    public Object create() {
        return create(nombrePorDefecto);
    }

    public Object create(String nombre) {
        nombrePorDefecto = nombre;
        Producto objeto = (Producto)mapaObjetos.get(nombre);
        return objeto != null ? objeto.clone() : null;
    }
}

public class PruebaFactoria {
    static public void main(String[] args) {
        FactoriaPrototipo factoria = new FactoriaPrototipo();
        Producto producto = (Producto) factoria.create("producto 1");
        System.out.println ("Este es el objeto creado: " + producto);
    }
}
```