Ej 1)

1. Se puede usar el patrón adapter ya que se quiere transformar el mismo archivo en distintos formatos.
2. Public interface ClientInterface{

Public

}

Ej 2

1. Capaz que adapter por las interfaces que son incompatibles??

Ej 3

1. Chatgpt me dice que use bridge, pero ni idea.

Ej 4

1. Se puede usar el patrón facade para unificar un conjunto de interfaces en un subsistema.
2. Public class Facade {

Private restaurantManagementSystem;

// repetir para cada subsistema

Public Facade(){

// instanciar todos los subsistemas

}

Public void BookTable(string tableType){

restaurantManagementSystem.bookTable(tableType)

} // repetir para todos los metodos

}

Class program {

Void main(){

Façade f = new Façade()

f.bookTable(“VIP”)

// …

}

}

Ej 5

1. Como se debe controlar el acceso al documento, es necesario implementar un proxy.
2. Public interface IDocument{

Public void Display();

}

Public class proxy: IDocument{

Private Document doc;

Public Proxy(Document d){

This.doc=d;

}

Public void checkAccess(){

// verificar acceso

}

Public void Display(){

If(this.checkAccess()){

Doc.display();

}

}

}

Public class program {

Public static void Main(){

Document doc = new Document(“ABC”);

Proxy p = new Proxy(doc)

p.display()

}

}

Ej 6

1. Se puede usar un façade para incorporar todos los subsistemas en una sola clase, para facilitar el uso de los mismos.
2. Public class Facade{

Private CartSystem cs;

Private InventorySystem is;

Private BillingSystem bs;

Public Facade(){

// instanciar todos los subsistemas

}

Public void addToCart(String producto, int quantity){

Cs.addToCart(product, quantity)

}

}

Ej 7

1. Se puede usar un facade para simplificar las interfaces que se usan???

Ej 8

1. Se puede usar un decorator para agregarle las funcionalidades al texto.
2. Public interface ElementoTexto{

Public void execute();

}

Public class ElementoTexto: IElementoTexto {

Public IElementoTexto execute(){

Return this;

}

}

Public class BaseDecorator: IElementoTexto {

Private IElementoTexto wrapee;

Public BaseDecorator(IElementoTexto et) {

This.wrapee=et;

}

Public IElementoTexto execute(){

Return Wrapee.execute()

}

}

Public class DecoratorEstiloFuente: BaseDecorator {

Private String EstiloFuente;

Publlic DecoratorEstiloFuente(IElementoTexto et, String estilo): base(et){

This.EstiloFuente=estilo

}

Public IElementoTexto execute(){

Return base.execute();

}

}