README - Proyecto 01- MonosChinosMX

Equipo: *Los Canekitas*

Nombre	Numero de Cuenta
Ramirez Macias Ulises Dante	319217535
Ramirez Palacios Miguel	322216376
Reyes Grimaldo Angel Ismael	119003833

Patrones de Diseno usado para el proyecto

Patron **Abstract Factory**

El patron abstract factory se implemento mediante la clase FabricaMestra, una clase que devuelve otro tipo de fabricas dependiendo del tipo que se requiera con ayuda de la clase ContratoFabrica, pues esta ayuda a obtener los componentes o piezas, haciendo que el builder y otro codigo utilizado no tenga que depender de las clases concretas, ademas cada fabrica ya sabe como realizar istancias de su tipo de pieza, por lo que es mas facil manejarlo.
Quisimos usar Abstract Factory para este segmento puesto que son muchas piezas y por tanto muchas fabricas, al implementar este patron nos permite manejar varias familias de piezas similares, ademas de que hace mas facil el poder agregar nuevas piezas.

Tipos de piezas y sus precios (en pesos) con los que se puede construir la PC

CPU
- INTEL:
  - Core i3-13100: 1500
  - Core i5-13600K: 3200
  - Core i7-13700K: 5800
  - Core i9-13900K: 9500
- AMD:
  - Ryzen 5 5600G: 1500
  - Ryzen 5 7600X: 3000
  - Ryzen 7 7700X: 6000
  - Ryzen 9 7950X3D: 9000

RAM
- Adata 8 GB: 500
- Adata 16 GB: 1000
- Adata 32 GB: 1500
- Kingston 8 GB: 550
- Kingston 16 GB: 1300
- Kingston 32 GB: 2000

Discos duros
- HDD:
  - Western Digital Blue 500 GB: 450
  - Western Digital Blue 1 TB: 600
  - Seagate Barracuda 1 TB: 550
  - Seagate Barracuda 2 TB: 800
- SSD:
  - Kingston 500 GB (SSD): 500

  - Kingston 1 TB (SSD): 800
  - Kingston 2 TB (SSD): 1600
  - Kingston 4 TB (SSD): 3200

Fuente de poder
- EVGA 800 W: 2200
- EVGA 1000 W: 3800
- EVGA 1500 W: 9500
- Corsair 800 W: 2500
- Corsair 1200 W: 5500
- Corsair 1500 W: 8000
- XPG 500 W: 1000
- XPG 700 W: 1500
- XPG 1000 W: 3000

Motherboard
- ROG Maximus Z790 Hero: 14000
- TUF Gaming B760-Plus: 3800
- MEG Z790 Godlike: 25000
- MAG B760 Tomahawk: 4200

Gabinete
- H6 Flow ATX (NZXT): 2800
- Lancer ATX (Yeiyian): 1500

GPU
- GTX 1660: 3500
- RTX 3060: 6500
- RTX 4070: 13000
- RTX 4080: 25000
- RTX 4090: 40000

Patron **Builder**

Nuestro patron builder se implementa mediante la clase ArmadaBuilder, que contiene todos los metodos para agregar las piezas que armaran a la nueva PC de tipo Compu, mientras que DirectorConstructor con ayuda de este builder permite crear varias combinaciones de PC como la economica y la premium que ya se encuentran ahi.
Esto lo hace paso a paso llamando al builder hasta terminar de ensamblar la computadora, regresando un objeto de tipo Compu o Compunent.
Pero no solo queda ahi, puesto que estas dos clases tambien son la base fundamental cuando se quiere crear una PC personalizada.

La razon por la que implementamos este patron es porque justamente como se llama el patron, una computadora se va construyendo paso a paso con las piezas que se quieren ir agregando.
Ademas de que este patron es flexible a la hora de crear distintas PCs, pues todas cumplen con la misma logica de construccion; solo sus piezas son las que cambian.
Tambien al haber tantas piezas disponibles para elegir, hace al programa mas mantenible y menos engorroso a la hora de elegir con que se quiere construir la PC.

Patron **Decorator**

El patron Decorator en el proyecto fue implementado mediante nuestra clase PCDecorada, un tipo de PC que recibe un Compunent (un objeto de tipo Compu al

principio para empezar a decorar) y luego un Programa.
Cada decorador agregado se envuelve a la PC base, agregando programas a una lista y actualizando el precio total de la PC cada vez que esto sucede.
La estructura permite poder encadenar estos decoradores, agregando uno detras de otro.

La razon por la que nos parecio factible utilizar este patron fue para evitar crear subclases para las combinaciones de los programas, ademas de que de cierta forma consideramos una analogia con la practica 2, pues en esta se hacia builder con un helado (base) al cual se le decoraba con varios toppings, pero en este caso es una computadora (base) a la que se decora con programas.
Ademas, el uso de este patron nos permitio agregar funcionalidades sin tener que modificar la clase Compu.

Programas que se pueden anadir:
- Windows 10, Precio: 500
- WSLTerminal: Linux, Precio: 500
- Adobe Photoshop 2025, Precio: 500
- Office 2024, Precio: 500
- AutoCAD 2025, Precio: 250

Patron **Proxy**

Nuestro patron Proxy tuvo que ser implementado de manera obligatoria, pero dada la problematica del proyecto sobre sucursales distantes, fue una buena eleccion y no resulto complicado saber en que parte debia aplicarse.

Primero, se definio la interfaz remota SucursalRemota, que contiene los metodos que una sucursal puede ejecutar a distancia.
Las sucursales concretas representan los objetos reales ubicados en distintos lugares (por ejemplo, Sucursal Jalisco o Sucursal Yucatan), mientras que SucursalProxy actua como un intermedio entre el cliente y dichas sucursales.
El servidor RMI tambien fue parte importante del patron, ya que tiene los objetos reales y gestiona su registro en el sistema RMI, permitiendo que los clientes puedan localizarlos mediante el proxy.

El proposito principal del patron proxy en el proyecto fue controlar el acceso a los objetos remotos, manejar errores o caidas de comunicacion entre sucursales sin que el cliente lo note, y proteger la interaccion directa con los objetos reales, asegurando que toda comunicacion ocurra de forma controlada a traves del proxy.

Instrucciones de Compilación y Ejecución

  Compilamos con:
  ```
  javac *.java
  ```
  Primero ejecutamos en una ventana de terminal con:
  ```
  java ServidorRMI
  ```
  Despues en una segunda ventana de terminal con:
  ```
  java App
  ```

NOTA: Comando utilizado en Linux y con la versión Java 23 (JDK 23)

Instrucciones de Ejecución para Docker:
Primero:
```
  sudo docker build -t app .
```
Despues:
```
sudo docker run -it --name rmi-test --rm app
```

Descripción del Programa

El proyecto MonosChinosMX es un sistema de venta y armado de computadoras que permite al usuario seleccionar entre PCs prediseñadas o construir una completamente personalizada desde 0.

El proyecto usa varios patrones de diseño como Abstract Factory, Builder, Decorator y Proxy, para estructurar el sistema de forma modular, escalable y flexible. Además, emplea RMI (Remote Method Invocation) para simular la comunicación entre sucursales remotas, haciendo que el cliente pueda comprar o distribuir computadoras entre distintas ubicaciones del país sin acceder directamente a los objetos remotos.