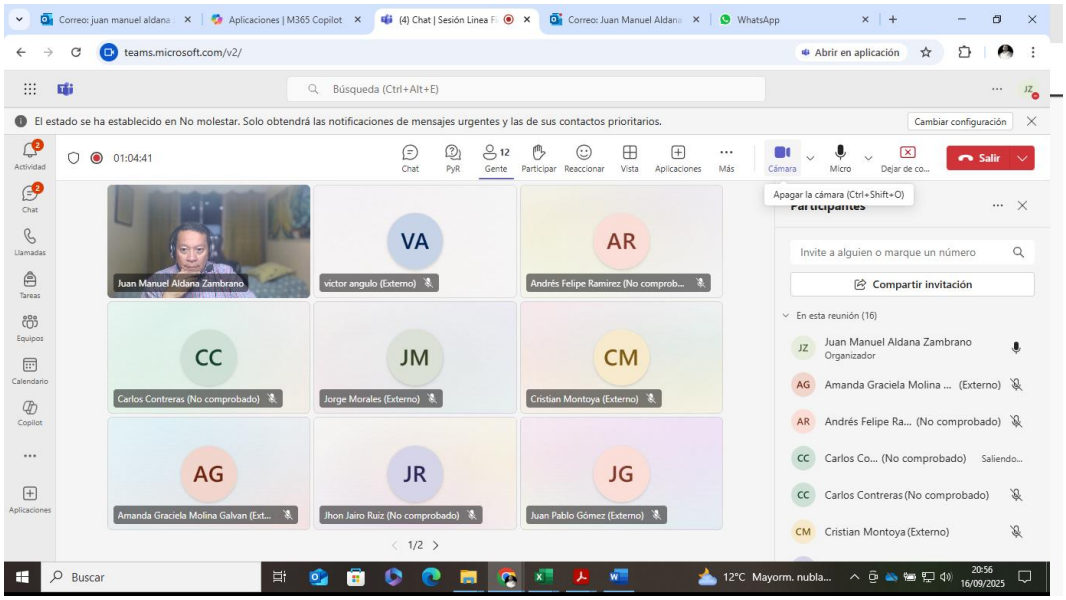
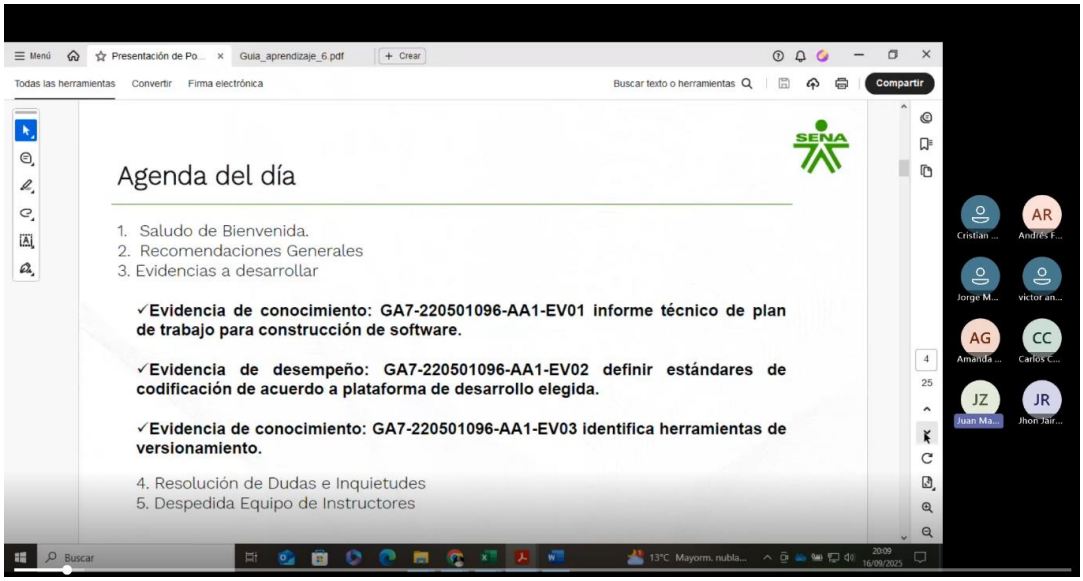
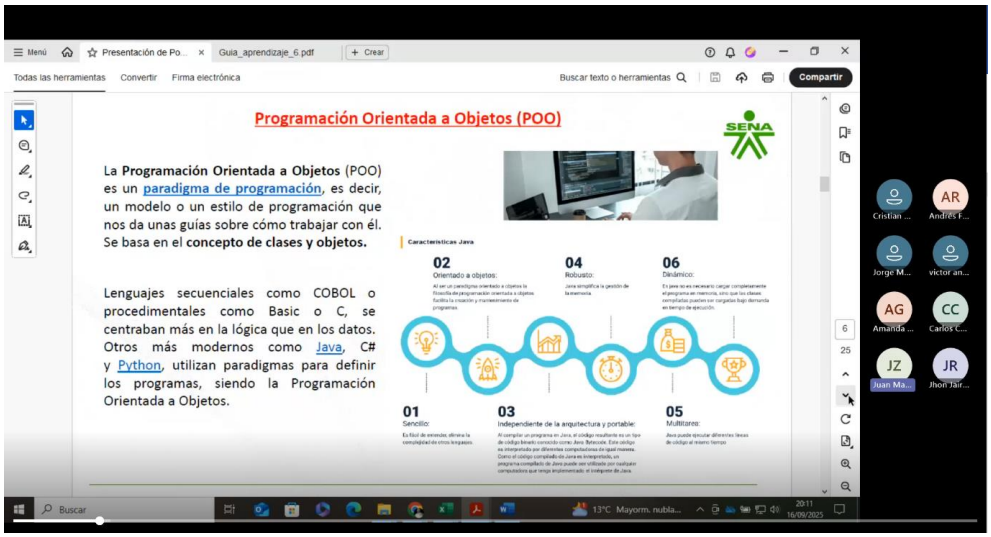
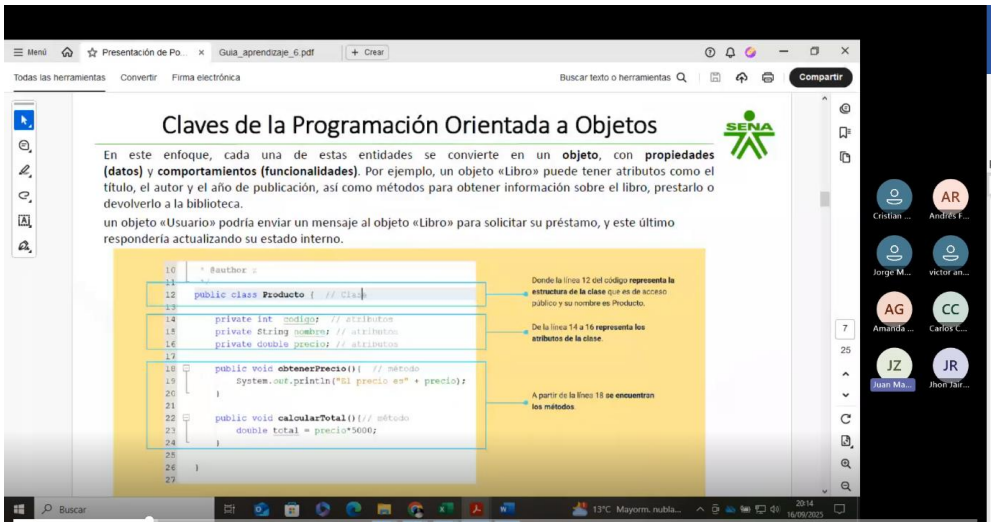
Ficha 3070297 - 3070301.

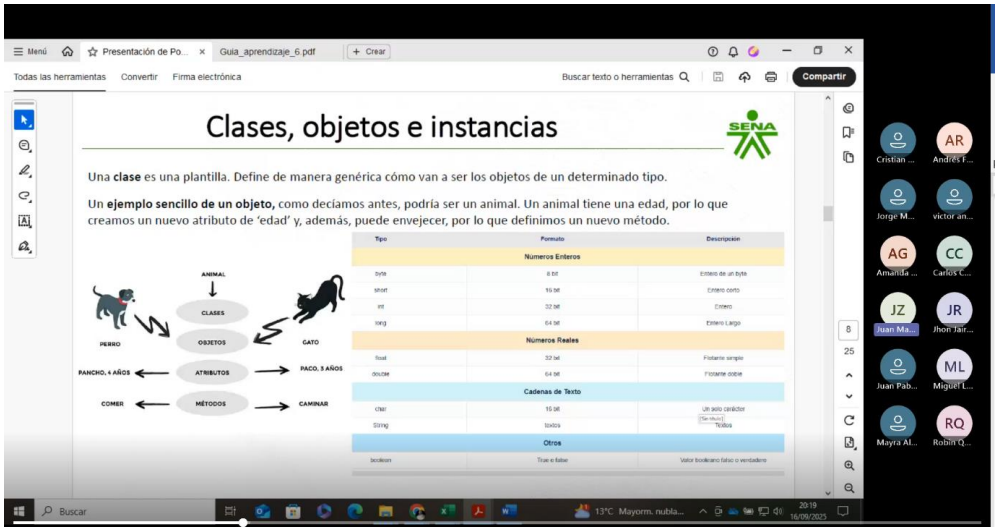
Grabación y resumen conferencia web – GA7-220501096-AA1-EV01, EV02 Y EV03. 16/09/2025. https://youtu.be/ZrwAhFv8zdc

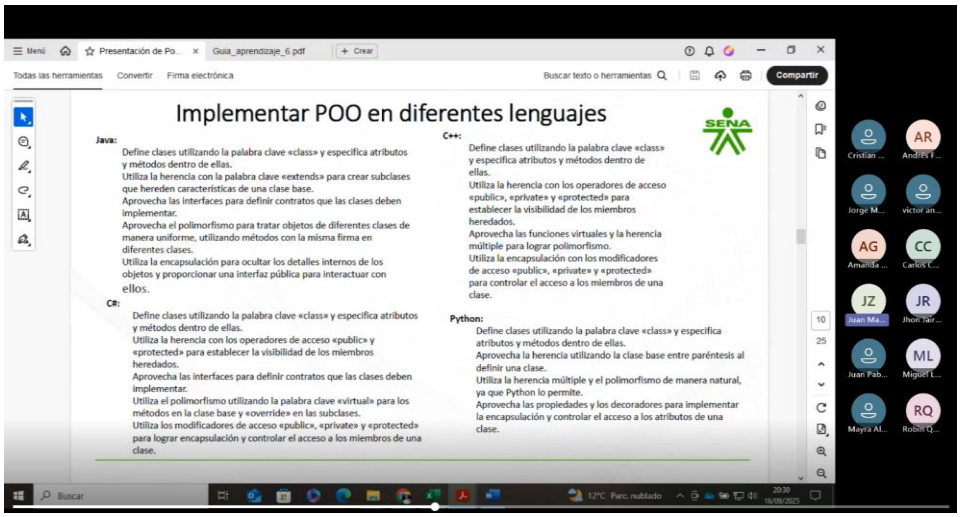


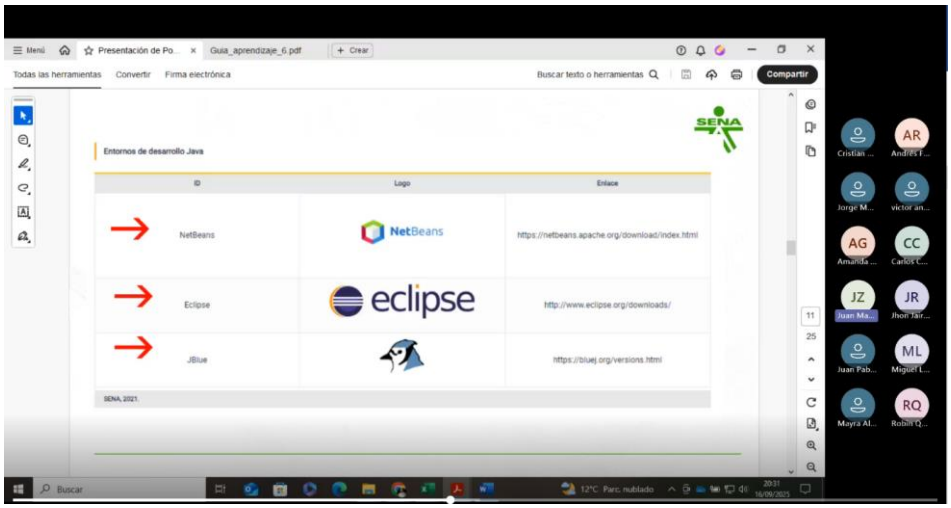


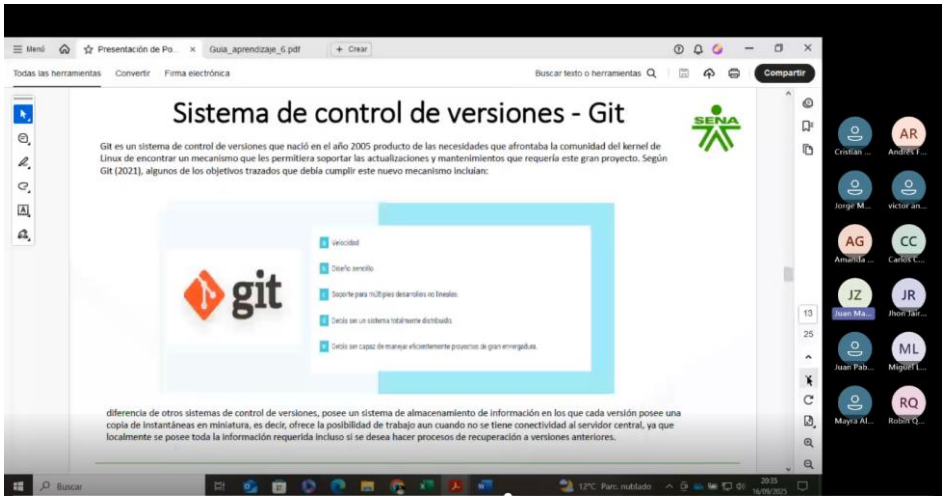

Programación Orientada a Objetos (POO)
La Programación Orientada a Objetos (POO) es un paradigma de programación, es decir, un modelo o un estilo de programación que nos da unas guías sobre cómo trabajar con él. Se basa en el concepto de clases y objetos.
Lenguajes secuenciales como COBOL o procedimentales como Basic o C, se centraban más en la lógica que en los datos. Otros más modernos como Java, C# y Python, utilizan paradigmas para definir los programas, siendo la Programación Orientada a Objetos.
Características Java
02 Orientado a objetos:
04 Robusto:
06 Dinámico:
01 Sencillo:
03 Independiente de la arquitectura y portable:
05 Multihilos:


Claves de la Programación Orientada a Objetos
En este enfoque, cada una de estas entidades se convierte en un objeto, con propiedades (datos) y comportamientos (funcionalidades). Por ejemplo, un objeto «Libro» puede tener atributos como el título, el autor y el año de publicación, así como métodos para obtener información sobre el libro, prestarlo o devolverlo a la biblioteca.
un objeto «Usuario» podría enviar un mensaje al objeto «Libro» para solicitar su préstamo, y este último respondería actualizando su estado interno.
Donde la línea 12 del código representa la estructura de la clase que es de acceso público y su nombre es Producto.
De la línea 14 a 16 representa los atributos de la clase.
A partir de la línea 18 se encuentran los métodos


Clases, objetos e instancias
Una clase es una plantilla. Define de manera genérica cómo van a ser los objetos de un determinado tipo.
Un ejemplo sencillo de un objeto, como decíamos antes, podría ser un animal. Un animal tiene una edad, por lo que creamos un nuevo atributo de 'edad' y, además, puede envejecer, por lo que definimos un nuevo método.
ANIMAL
CLASES
OBJETOS
ATRIBUTOS
MÉTODOS
PERRO
GATO
PANCHO, 4 AÑOS
PACO, 3 AÑOS
COMER
CAMINAR
Tipo
Formato
Descripción
Números Enteros
Números Reales
Cadenas de Texto
Otros


Implementar POO en diferentes lenguajes
Java:
Define clases utilizando la palabra clave «class» y especifica atributos y métodos dentro de ellas.
Utiliza la herencia con la palabra clave «extends» para crear subclases que hereden características de una clase base.
Aprovecha las interfaces para definir contratos que las clases deben implementar.
Aprovecha el polimorfismo para tratar objetos de diferentes clases de manera uniforme, utilizando métodos con la misma firma en diferentes clases.
Utiliza la encapsulación para ocultar los detalles internos de los objetos y proporcionar una interfaz pública para interactuar con ellos.
C#:
Define clases utilizando la palabra clave «class» y especifica atributos y métodos dentro de ellas.
Utiliza la herencia con los operadores de acceso «public» y «protected» para establecer la visibilidad de los miembros heredados.
Aprovecha las interfaces para definir contratos que las clases deben implementar.
Utiliza el polimorfismo utilizando la palabra clave «virtual» para los métodos en la clase base y «override» en las subclases.
Utiliza los modificadores de acceso «public», «private» y «protected» para lograr encapsulación y controlar el acceso a los miembros de una clase.
C++:
Define clases utilizando la palabra clave «class» y especifica atributos y métodos dentro de ellas.
Utiliza la herencia con los operadores de acceso «public», «private» y «protected» para establecer la visibilidad de los miembros heredados.
Aprovecha las funciones virtuales y la herencia múltiple para lograr polimorfismo.
Utiliza la encapsulación con los modificadores de acceso «public», «private» y «protected» para controlar el acceso a los miembros de una clase.
Python:
Define clases utilizando la palabra clave «class» y especifica atributos y métodos dentro de ellas.
Aprovecha la herencia utilizando la clase base entre paréntesis al definir una clase.
Utiliza la herencia múltiple y el polimorfismo de manera natural, ya que Python lo permite.
Aprovecha las propiedades y los decoradores para implementar la encapsulación y controlar el acceso a los atributos de una clase.


Entornos de desarrollo Java
ID
Logo
Enlace
NetBeans
https://netbeans.apache.org/download/index.html
Eclipse
http://www.eclipse.org/downloads/
JBlue
https://bluej.org/versions.html
SENA, 2021.


Sistema de control de versiones - Git
Git es un sistema de control de versiones que nació en el año 2005 producto de las necesidades que afrontaba la comunidad del kernel de Linux de encontrar un mecanismo que les permitiera soportar las actualizaciones y mantenimientos que requería este gran proyecto. Según Git (2021), algunos de los objetivos trazados que debía cumplir este nuevo mecanismo incluían:
git
diferencia de otros sistemas de control de versiones, posee un sistema de almacenamiento de información en los que cada versión posee una copia de instantáneas en miniatura, es decir, ofrece la posibilidad de trabajo aun cuando no se tiene conectividad al servidor central, ya que localmente se posee toda la información requerida incluso si se desea hacer procesos de recuperación a versiones anteriores.