ISTRED
INSTITUTO SUPERIOR TÉCNICO REY DAVID

PERIODO II
2023 - 2024

GUÍA DE APRENDIZAJE DEL ESTUDIANTE
INTRODUCCIÓN A SISTEMAS

ING. ANTHONY HERNÁNDEZ LEÓN

GESTIÓN DE BASE DE DATOS
1ER. SEMESTRE

Contenido

I. DATOS GENERALES DE LA ASIGNATURA 8

Información General 8

Requisitos 8

Descripción de la asignatura 8

Objetivo de la asignatura 9

Introducción de la Asignatura 9

I. DATOS GENERALES DE LA UNIDAD 10

Información General 10

Resultado de Aprendizaje de Unidad 10

Metodología de Unidad 10

DESARROLLO 11

Unidad 1: Introducción a sistemas Computacionales 11

Tema 1: Fundamentos Computacionales 11

Datos 11

La Información 12

Importancia de la Información 13

Computadoras e Información 14

Tecnologías de la Información 14

Conceptos de Computación 16

Búsqueda de la Información 16

Aplicaciones de Internet 17

Navegadores y Buscadores 18

ACTIVIDADES PRÁCTICAS 20
FORO DE DEBATE 21
PRÁCTICA INVESTIGATIVA 21
PRÁCTICA TEST 22
Tema 2: Anatomía de la Computadora Hardware 23
Funcionalidades de un Computador 23
Dispositivos de Entrada y de Salida 23
Dispositivos de Entrada 23
Dispositivos de Salida 24
Dispositivos de Entrada/Salida (E/S) 24
Definición de Servidor 25
Tipos de Servidores 25
Usos de Servidores 26
ACTIVIDADES PRÁCTICAS 27
FORO DE DEBATE 28
PRÁCTICA INVESTIGATIVA 28
PRÁCTICA TEST 29
II. DATOS GENERALES DE LA UNIDAD 2 30
DESARROLLO 31
Unidad 2: Software para procesamiento de Datos 31
Tema 1: Sistemas Operativos 31
Objetivo 32
Interrelación con el Hardware 33

Clasificación de sistemas Operativos 34
Servicios básicos de un Sistema Operativo 35
Interacción con el usuario 37
ACTIVIDADES PRÁCTICAS 39
FORO DE DEBATE 40
PRÁCTICA INVESTIGATIVA 40
PRÁCTICA TEST 41
Tema 2: Evolución de los Sistemas Operativos y sus aplicaciones 42
Familia Unix 42
Sistema Operativo Windows 42
Características principales: 42
Apple (macOS e iOS) 43
macOS (antes OS X) 43
iOS 43
Linux 43
Características principales: 43
Android 44
Características: 44
Programas de Aplicación 44
Usos del Software 44
Herramientas Ofimáticas 45
ACTIVIDADES PRÁCTICAS 46
FORO DE DEBATE 47

PRÁCTICA INVESTIGATIVA 47
PRÁCTICA TEST 48
III. DATOS GENERALES DE LA UNIDAD 3 49
Información General 49
Resultado de Aprendizaje de Unidad 49
Metodología de Unidad 49
DESARROLLO 50
Unidad 3: Introducción a Programación y Desarrollo de software 50
Tema 1: Introducción a Programación 50
Introducción 51
¿Cuál es el paradigma más usado? 52
Atributos y métodos 54
¿Qué es una clase? 55
Maquinas Programables 56
Programas de computadora 58
ACTIVIDADES PRÁCTICAS 60
FORO DE DEBATE 61
PRÁCTICA INVESTIGATIVA 61
PRÁCTICA TEST 62
Tema 2: Lenguajes de Programación 63
Lenguajes de Programación más usados 63
Ventajas los intérpretes: 65
Desventajas los intérpretes: 65

Evolución de los lenguajes de programación 65
Entornos de desarrollo 67
ACTIVIDADES PRÁCTICAS 70
FORO DE DEBATE 71
PRÁCTICA INVESTIGATIVA 71
PRÁCTICA TEST 72
IV. DATOS GENERALES DE LA UNIDAD 4 73
Resultado de Aprendizaje de Unidad 73
Metodología de Unidad 73
DESARROLLO 74
Unidad 4: Tipos de Programación 74
Tema 1: Programación Estructurada y Programación orientada a objetos 74
Programación Estructurada 74
Programación Orientada a Objetos 75
Principios fundamentales del POO: 75
La POO tiene una serie de ventajas, incluyendo: 76
Diferencias entre ellas 76
Ejemplos de lenguajes: 78
Clases y Objetos 78
Clases 78
Objetos 79
Clases y Objetos en la Programación Orientada a Objetos 80
Atributos y métodos de una clase 81

Atributos y métodos de una clase en la Programación Orientada a Objetos (POO) 83

ACTIVIDADES PRÁCTICAS 86

FORO DE DEBATE 87

PRÁCTICA INVESTIGATIVA 87

PRÁCTICA TEST 88

Tema 2: Desarrollo de Software 89

¿Qué es el software? 89

Componentes del software 89

Tipos de software 89

Importancia del software 90

Ciclo de vida del Desarrollo de Software 90

Fases del ciclo de vida del software 91

Planificación 91

Modelos de ciclo de Vida 94

Modelo en cascada 94

Ventajas 94

Desventajas 94

Modelo en V 94

Ventajas 95

Desventajas 95

Modelo de prototipos 95

Ventajas del modelo de prototipado 95

Desventajas del modelo de prototipado 95
Modelo en Espiral 96
Ventajas 96
Desventajas 96
Modelo de desarrollo incremental 96
Métodos Ágiles 96
ACTIVIDADES PRÁCTICAS 98
FORO DE DEBATE 99
PRÁCTICA INVESTIGATIVA 99
PRÁCTICA TEST 100
V. Bibliografía 101
Bibliografía básica 101
Bibliografía de consulta 101
Anexo 1: Modelo de Foro 102
Anexo 2: Esquema de Ensayo 103
Anexo 3: Esquema de Taller/Investigación 104
Anexo 4: Rúbrica de Foro 105
Anexo 5: Rubrica de Ensayo 106
Anexo 6: Rúbrica de Taller/Investigación 107

I. DATOS GENERALES DE LA ASIGNATURA

Información General

1.1.	Asignatura	Introducción a Sistemas					
1.2.	Carrera	Gestión de Bases de Datos					
1.3.	Código de asignatura	GBD1234					
1.4.	Créditos	2,5					
1.5.	Nivel	Primero					
1.6.	Detalle de horas	ACD	60	AA	12	APE	48

Requisitos

Prerrequisito		Correquisito	
Asignatura	Código	Asignatura	Código
N/A	N/A	N/A	N/A

Descripción de la asignatura

La asignatura "Introducción a Sistemas" desempeña un papel esencial en la formación del profesional superior en Gestión de Bases de Datos al proporcionar una base sólida de conocimientos y habilidades específicas para este campo. En primer lugar, esta asignatura permite a los estudiantes comprender los conceptos fundamentales de los sistemas de información y cómo se aplican a la gestión de bases de datos. Los estudiantes aprenden a identificar los componentes clave de un sistema de bases de datos, como las tablas, relaciones, consultas y la interfaz de usuario, así como las interacciones entre ellos. Además, se enfoca en el diseño y modelado de bases de datos, lo que es crucial para crear estructuras eficientes y funcionales que satisfagan las necesidades de organizaciones y empresas.

Los estudiantes también adquieren habilidades para analizar sistemas complejos de bases de datos y resolver problemas relacionados con la integridad y la seguridad de los datos. La asignatura promueve la aplicación de enfoques sistémicos en la gestión de bases de datos, lo que ayuda a los futuros profesionales a comprender cómo sus decisiones afectan a todo el sistema de información de una organización. En última

instancia, esta asignatura prepara a los estudiantes para enfrentar los desafíos del mundo laboral en el campo de las Bases de Datos, equipándolos con las habilidades necesarias para diseñar, administrar y mantener sistemas de bases de datos de manera eficiente y efectiva, contribuyendo así al éxito de las organizaciones en un entorno cada vez más dependiente de la información y los datos.

Objetivo de la asignatura

Proporcionar a los estudiantes una comprensión sólida y holística de los sistemas de información, preparándolos para aplicar enfoques sistémicos en la gestión de bases de datos y ser capaces de contribuir eficazmente a la toma de decisiones relacionadas con datos en organizaciones y empresas.

Introducción de la Asignatura

La asignatura "Introducción al Software" representa una puerta de entrada esencial al vasto mundo de la informática y el desarrollo de software en la educación superior. Este curso proporciona a los estudiantes una base sólida en los conceptos fundamentales relacionados con el software, su desarrollo y su papel central en la era digital. A medida que la tecnología continúa transformando nuestras vidas y sociedades, comprender cómo se crea y se utiliza el software se vuelve cada vez más crucial. A lo largo de este curso, exploraremos los conceptos básicos de la programación, la arquitectura de software, las metodologías de desarrollo y las tendencias tecnológicas emergentes. Además, aprenderemos a abordar problemas, diseñar soluciones y comunicar eficazmente nuestras ideas en el mundo del software. La asignatura "Introducción al Software" sienta las bases para un viaje emocionante y en constante evolución en el que los estudiantes se convertirán en creadores y solucionadores de problemas en el reino del software y la tecnología.

I. DATOS GENERALES DE LA UNIDAD

Información General

1.1.	Nivel	Primero					
1.2.	Nombre Unidad	Introducción a sistemas Computacionales.					
1.3.	Tema 1 – Unidad	Fundamentos Computacionales					
1.4.	Tema 2 – Unidad	Anatomía de la Computadora Hardware					
1.5.	Unidad organizacional	Unidad Profesional					
1.6.	Unidad	1					
1.7.	Total, Horas Unidad	30					
1.8.	Detalle de horas Unidad	ACD	15	AA	3	APE	12

Resultado de Aprendizaje de Unidad

Al finalizar esta unidad, los estudiantes serán capaces de utilizar herramientas fundamentales de software para el procesamiento de datos, incluyendo bases de datos y programas de análisis estadístico. Demostrarán habilidades prácticas en el manejo de herramientas como SQL, Excel y Python, aplicando técnicas para la extracción, transformación y carga de datos. Además, serán capaces de diseñar y ejecutar consultas de bases de datos, realizar análisis estadísticos básicos, interpretar resultados y crear visualizaciones efectivas para comunicar hallazgos de manera clara. Este resultado de aprendizaje busca proporcionar a los estudiantes una sólida comprensión práctica y teórica de las herramientas esenciales para la manipulación y análisis de datos.

Metodología de Unidad

La Metodología de se basa en un enfoque activo y colaborativo. Incluye sesiones de clase estructuradas en torno a temas específicos, lecturas y tareas, estudio de casos, proyectos prácticos y debates en clase. Los estudiantes también se involucran en trabajos en grupo y mantienen diarios de reflexión. Se fomenta la participación en debates y preguntas y respuestas, y se proporcionan recursos en línea para una comprensión más profunda. La evaluación se realiza mediante pruebas, proyectos y una evaluación final, con retroalimentación continua para guiar el aprendizaje.

DESARROLLO

Unidad 1: Introducción a sistemas Computacionales

Tema 1: Fundamentos Computacionales

Objetivo: Comprender los fundamentos de los sistemas, Los estudiantes deben ser capaces de definir y explicar los conceptos básicos de los sistemas, incluyendo componentes, relaciones, entradas y salidas, y cómo estos conceptos se aplican a los sistemas de información y las bases de datos.

Los fundamentos de la información y los datos son pilares esenciales en la era digital. Los datos representan hechos y cifras en bruto, mientras que la información es el resultado del procesamiento de estos datos, dotándolos de significado y contexto. El procesamiento de datos implica actividades como la organización y clasificación, transformando datos crudos en información útil. Este ciclo de información abarca desde la recolección y almacenamiento de datos hasta su transmisión y uso en la toma de decisiones. La calidad de los datos, que incluye precisión y confiabilidad, es crucial para garantizar la fiabilidad de la información generada. Además, la privacidad y seguridad de los datos son imperativos para proteger la información confidencial y cumplir con las normativas de privacidad. En conjunto, estos conceptos fundamentales desempeñan un papel esencial en la gestión y utilización efectiva de la información en un mundo cada vez más orientado a los datos.

Datos

Los datos son elementos fundamentales en la era de la información, constituyendo representaciones simbólicas de hechos, cifras, eventos o cualquier forma de información en su estado más crudo y sin procesar. Estos pueden manifestarse en diversas formas, como números, textos, imágenes, sonidos, entre otros. Los datos carecen de significado intrínseco y contexto por sí mismos, pero adquieren relevancia cuando se someten a procesos de interpretación y análisis.

En esencia, los datos actúan como componentes primarios que, una vez procesados y organizados, se convierten en información que se puede utilizar para tomar decisiones, generar conocimiento y facilitar la comunicación. La calidad de los datos, que engloba aspectos como precisión, integridad y confiabilidad, es crucial para asegurar que la información resultante sea precisa y confiable.

En un mundo cada vez más impulsado por la tecnología y la digitalización, los datos son activos valiosos que impulsan la toma de decisiones, la innovación y el avance en diversos campos. (M. Planas, 2004)

Imagen 3: Datos



Tomado de: https://www.itconsultors.com/ciencia-de-datos-que-es-para-que-sirve-ventajas-y-consejos

La Información

La información es el resultado del procesamiento de datos, donde los datos crudos se transforman en conocimiento significativo y utilizable. A diferencia de los datos aislados, la información está contextualizada, organizada y dotada de significado, lo que la convierte en un recurso esencial para la toma de decisiones, la comunicación y la generación de comprensión.

La información puede manifestarse en diversas formas, como informes, documentos, gráficos, mensajes o cualquier medio que transmita conocimiento. Su valor radica en su capacidad para proporcionar respuestas, soluciones o conocimientos en un contexto específico. La calidad de la información depende en gran medida de la calidad de los datos subyacentes y de la precisión, relevancia y actualidad de esta.

En un mundo cada vez más orientado hacia la información y la tecnología, la habilidad para gestionar y utilizar información de manera efectiva es crucial para el éxito en una amplia gama de sectores y disciplinas. (T, 1988)

Imagen 4: Información



Tomado de https://www.capterra.es/blog/639/software-base-de-datos-gratuitos-codigo-abierto

Importancia de la Información

La información, entendida como datos interrelacionados que brindan conocimiento útil y valioso, es el componente más relevante de todo sistema informático y de gestión. (GARCÍA, 2013). Además, es el activo más valioso para organizaciones, empresas y personas. Sin embargo, al cuidado de la información como activo no se le suele asignar la importancia que se merece o que se requiere, algunas veces por desconocimiento y otras por incompetencia del administrador de esta.

El trato descuidado de la información individual y organizacional puede llevar a su destrucción, pérdida, alteración y exposición no deseada de la misma La destrucción o pérdida conllevan a pérdidas de tiempo y dinero a veces incuantificables. Por otra parte, la alteración puede llevar a la toma de decisiones equivocadas, las cuales afectarían a los usuarios o medios para los cuales va a ser usada tal información. Aún más grave es la exposición o publicación no deseada de información, que implica pérdida de privacidad y confidencialidad, esto genera una desconfianza de los usuarios que depositan su información a las organizaciones y la vez pueden generar daños irreparables a las personas.

Conforme la humanidad y nuestra sociedad en se adentran en la Era de la Información y del Conocimiento, las teorías y técnicas aplicadas a la protección de la información se hacen más relevantes obtener para organizaciones, usuarios o individuos para proteger la información.

Computadoras e Información

Las computadoras y la información están entrelazadas en la era digital. Las computadoras, desde sus humildes inicios hasta las potentes máquinas que hoy en día conocemos, han revolucionado la forma en que almacenamos, procesamos y compartimos información. Son herramientas versátiles que abarcan desde dispositivos portátiles hasta supercomputadoras, permitiéndonos llevar a cabo tareas múltiples que nos permiten ahorrar tiempo a cada usuario, desde la comunicación hasta la investigación científica.

La información se presenta en diversas formas, como datos, texto, imágenes, videos y más. Las computadoras actúan como vías las cuales nos permiten acceder a esta información y utilizarla de maneras creativas y productivas. La capacidad de procesamiento de datos de las computadoras ha impulsado avances en campos como la inteligencia artificial, la medicina, la ciencia de datos y la ingeniería, etc.

La información es un recurso valioso para la toma de decisiones, la innovación y la resolución de problemas. Las computadoras nos permiten organizar, analizar y compartir esta información de manera eficaz y eficiente, lo que genera un impacto importante en la sociedad y los negocios.

Es decir, las computadoras y la información están en el centro de la revolución tecnológica en la actualidad, y su implementación continúa dando forma a nuestro mundo de maneras impactantes hace ya unas décadas (Castells, 1999).

Imagen 5: Computadoras e Información



Tomado de https://humanidades.com/redes-de-computadoras/

Tecnologías de la Información

Las tecnologías abarcan una amplia gama de herramientas y sistemas que abarcan desde computadoras y dispositivos personales hasta redes globales de comunicación.

Han transformado la forma en que vivimos, nos relacionamos y trabajamos (Almenara, 1998)

La digitalización de datos ha permitido tener acceso de manera rápida, eficaz y eficiente a la información en todo el mundo, facilitando la comunicación global y la colaboración en tiempo real. Además, las tecnologías de la información han impulsado la automatización de procesos, mejorando la eficiencia en industrias como la manufactura, la atención médica y la logística.

Las tecnologías de la información abarcan todos los dispositivos y plataformas, como:

Ordenadores: (Sistema conformado por programas y elementos electrónicos, que en su conjunto permiten procesar y ordenar información.)

Software: (conjunto de programas, instrucciones, algoritmos y partes visuales que nos permiten interactuar con el computador)

Sitios web: (espacio virtual que contiene toda la información que una persona o empresa quiere difundir a través de internet)

Servidores: (sistema que proporciona recursos, datos, servicios o programas a otros ordenadores, conocidos como clientes, a través de una red).

A medida que estas tecnologías avanzan, es de vital importancia abordar cuestiones relacionadas con la privacidad, la seguridad y la ética para que el impacto de estas sea positivo para los usuarios y se convierta en una herramienta sostenible y confiable. Es decir, las tecnologías de la información se han convertido en un motor de cambio y progreso en el mundo actual. (Daniel, 2007)

Imagen 6: Tecnologías de la Información



Tomado de https://tome.app/

Conceptos de Computación

Son los pilares relevantes que sustentan la era digital actual en la que nos encontramos. Comprenden una variedad de principios fundamentales que permiten el funcionamiento de las computadoras y sistemas de información. Si hablamos de conceptos claves, nos referiríamos a la programación, que es la creación de instrucciones que guían a las computadoras para realizar tareas específicas; los algoritmos, que son secuencias lógicas de pasos diseñados para resolver problemas; las estructuras de datos, que organizan y almacenan información de manera eficiente; y la abstracción, que simplifica problemas complejos al dividirlos en componentes más manejables. (Olvera, 2014)

Entender sobre estos conceptos es preciso en un mundo cada vez más digitalizado, ya que no solo son relevantes para los profesionales de la informática, sino que también son relevantes para los usuarios ya que fomentan un pensamiento lógico, que ayudara en la resolución de problemas en la vida cotidiana.

Los conceptos sobre computación son uno de los pitares en la tecnología moderna y actual, son cruciales para aprovechar al máximo las oportunidades que ofrece la revolución tecnológica actual. Son herramientas poderosas que capacitan a las personas para navegar y prosperar en la sociedad digital que día a día va en constante evolución.

Búsqueda de la Información

La búsqueda de información es la habilidad fundamental que un usuario en la era de digitalización en la que nos encontramos debería optar por tenerle presente y aplicarla a la hora de buscar información en la red.

La búsqueda de información implica la capacidad de localizar, recopilar y evaluar datos relevantes de diversas fuentes y unificarlos creando una información clara y entendible, como libros, sitios web, bases de datos, etc.

En un mundo donde la cantidad de información disponible es abrumadora y abundante, saber cómo buscar de manera eficaz y eficiente se ha convertido en una habilidad valiosa ya que esto implica aprender a sacar la información sobre algún tema en general de forma eficiente. Esto implica no solo ser experto en el uso de motores de búsqueda en línea, sino también en la capacidad de identificar entre fuentes

confiables y no confiables, verificar la veracidad de la información y unificar datos para obtener una comprensión completa de un tema.

La búsqueda de información no solo implica de veracidad de la información, sino también la actualidad de esta, la relevancia contextual, también implica el uso del uso de filtros, palabras claves y estrategias avanzadas para, para obtener la información más precisa, entendible y actualizada posible para los usuarios

La búsqueda de información es esencial en ámbitos académicos, profesionales y cotidianos, ya que nos permite tomar decisiones informadas, aprender y mantenernos actualizados en un mundo en constante cambio. (Bravo, 2002)

Imagen 7: Búsqueda de la información



Tomado de https://lizblogeducare.wordpress.com/2022/02/06/busqueda-de-recursos-en-internet/

Aplicaciones de Internet

Las aplicaciones de internet nos permiten trabajar y comunicarnos entre la sociedad, ya sea por medios como redes sociales que conectan a todo el mundo como hasta plataformas de streaming que ofrecen entretenimiento de variedad internet ha revolucionado la manera en la que interactuamos con el contenido y la información que este nos ofrece.

Si hablamos en el ámbito laboral, las aplicaciones de Internet facilitan la colaboración en línea, el teletrabajo y la gestión eficiente de proyectos. la educación en línea ha experimentado un crecimiento exponencial gracias a estas aplicaciones, ofreciendo oportunidades de aprendizaje a nivel global

Es decir, las aplicaciones de Internet han permeado prácticamente todos los aspectos de la vida en la actualidad, generando una revolución en la forma en que vivimos y nos relacionamos en este mundo digitalizado.

Imagen 8: Búsqueda de la información



Tomado de https://tome.app/

Navegadores y Buscadores

Los navegadores y buscadores son herramientas que no pueden faltar en el mundo digital actual. Los navegadores web, como:

- Chrome.
- Firefox.
- Safari.

Actúan como puertas de entrada a Internet, permitiéndonos acceder a una extensa gama de contenidos o información en línea. Estas aplicaciones nos brindan la capacidad de explorar sitios o páginas web, interactuar con aplicaciones en línea y disfrutar de servicios como el correo electrónico y las redes sociales.

Por otro lado, los buscadores, encabezados por:

- Google
- Bing

Y otros, son motores de búsqueda que nos ayudan a encontrar información específica en la red de Internet. Utilizan algoritmos sofisticados para indexar y clasificar millones de páginas web, lo que facilita la localización de recursos, respuestas a preguntas y datos relevantes.

En resumen, los navegadores y buscadores son herramientas cruciales para la navegación en línea y la búsqueda de información en la era digital. El uso adecuado de los navegadores y buscadores es fundamental para aprovechar al máximo la inmensa cantidad de conocimiento y recursos disponibles en la web que estas pueden brindar a cada usuario (Padilla Collado, 2014)

Imagen 9: Navegadores y Buscadores



Tomado de: https://www.aldia.unah.edu.pe/confundiendo-navegador-web-y-buscador-web/

ACTIVIDADES PRÁCTICAS

Tabla

Detalle de la Práctica

Practicas	Instrucciones	Duración de la Practica	Mecanismo de Evaluación
Foro de debate	Revisar Actividades Prácticas	0:30 mm	Por rúbricas
Práctica investiga: Taller/Ensayo/Investigación	Revisar Actividades Prácticas	3:30 h	Por rúbricas
Tutorías Prácticas	Revisar Actividades Prácticas	2:00 mm	Por participación en clase
Test evaluativo	Revisar Actividades Prácticas	0:30 mm	Formativa

Nota. Elaboración propia

Consideraciones Generales

1. Lee y revisa la Guía de Estudio.
2. Observa el video de la clase magistral.
3. Participa activamente en el FORO DEBATE.
4. Revisa en el calendario el día y hora en que se realizará las actividades prácticas y la tutoría.
5. Realizar la práctica propuesta por el docente.
6. Revisa las Rúbricas acorde a la actividad prácticas.
7. Toma los principales apuntes que vayas rescatando del texto.
8. Reflexiona sobre los temas tratados en la clase.
9. Sube el documento en formato PDF en la plataforma Moodle.
10. Revisa la calificación de tus actividades prácticas.

FORO DE DEBATE

Tema: ¿Por qué POO?

Instrucciones para Participar:

- Exponga su punto de vista acerca del tema.
- Comenta las respuestas de otros participantes.
- Promueve un ambiente respetuoso y constructivo.

PRÁCTICA INVESTIGATIVA

Taller: Elaboración de Ensayo de importancia de la Información y sus usos.

Consigna:

- El trabajo de investigación debe presentarse de manera estructurada y coherente para comunicar efectivamente los hallazgos y conclusiones.
- Asegúrese de que el estilo y formato del trabajo sigan las pautas de citación y presentación establecidas por la institución.
- Utilice un lenguaje claro y evite jergas o términos técnicos innecesarios.
- Revise y corrija la redacción y la gramática antes de la presentación final (ver anexo).

PRÁCTICA TEST

Antes de realizar el test, es importante considerar varios aspectos para aprovechar al máximo la experiencia.

Consideraciones para realizar el test

1. Lee y revisa el documento.

2. Observa el video de la clase magistral.

3. Revisa en el calendario el día y hora en que se debe de realizar el Test.

4. Realiza el Test de la unidad considerando lo Siguiente:

- No cambiarte entre paginas durante la realización del Test.
- Revisa que tu conexión a internet sea estable.
- Recuerda que todo el contenido está inmerso dentro del documento base.

5. Revisa tu puntuación, de inmediato al finalizar de la actividad.

Tema 2: Anatomía de la Computadora Hardware

Objetivo: Comprender la anatomía de la computadora y su hardware, analizando sus funcionalidades, dispositivos de entrada y salida, y componentes principales, así como la definición, tipos y usos de los servidores, con el propósito de fortalecer el conocimiento sobre la arquitectura informática y su aplicación en diferentes entornos tecnológicos.

El hardware de una computadora es el conjunto de componentes físicos que permiten su funcionamiento. Comprender su estructura y funciones es fundamental para el uso y mantenimiento de los sistemas informáticos.

Funcionalidades de un Computador

Un computador es un dispositivo electrónico diseñado para **procesar información** mediante la ejecución de instrucciones en forma de programas. Sus principales funcionalidades incluyen:

- **Procesamiento de Datos**: Conversión de datos en información útil mediante la CPU (Unidad Central de Procesamiento).
- **Almacenamiento de Información**: Uso de memorias (RAM, ROM, discos duros, SSD) para guardar y recuperar datos.
- **Interacción con el Usuario**: Uso de dispositivos de entrada y salida para permitir la comunicación con el sistema.
- **Conectividad**: Integración con redes para compartir información y acceder a servicios en línea.
- **Automatización de Tareas**: Ejecución de programas para realizar cálculos, edición de documentos, diseño gráfico, entre otros.

Dispositivos de Entrada y de Salida

Los dispositivos de entrada y salida permiten la comunicación entre el usuario y la computadora.

Dispositivos de Entrada

Son aquellos que permiten ingresar datos al sistema. Algunos ejemplos son:

- Teclado: Introducción de texto y comandos.
- Mouse: Navegación y selección de elementos en pantalla.
- Escáner: Digitalización de documentos e imágenes.
- Micrófono: Captura de sonido para grabación o videoconferencias.
- Cámara Web: Captura de imágenes y video en tiempo real.

Dispositivos de Salida

Son aquellos que muestran o transmiten información al usuario. Algunos ejemplos son:

- Monitor: Visualización de gráficos e interfaz de usuario.
- Impresora: Impresión de documentos físicos.
- Altavoces: Reproducción de audio.
- Proyector: Proyección de imágenes en pantallas grandes.

Dispositivos de Entrada/Salida (E/S)

Algunos dispositivos pueden cumplir ambas funciones:

- Pantallas táctiles: Permiten ingresar datos y visualizar información al mismo tiempo.
- Memorias USB: Almacenan y transfieren información.
- Discos Duros Externos: Guardan datos y permiten su recuperación.

Partes de un Computador

El hardware de una computadora está compuesto por varios componentes esenciales:

- CPU (Unidad Central de Procesamiento): Es el "cerebro" del computador, encargado de procesar instrucciones y realizar cálculos.
- Placa Madre (Motherboard): Tarjeta principal que conecta todos los componentes.
- Memoria RAM (Random Access Memory): Almacena temporalmente los datos y programas en ejecución.
- Dispositivos de Almacenamiento: Incluyen discos duros (HDD), unidades de estado sólido (SSD) y memorias USB.
- Fuente de Alimentación: Suministra energía a todos los componentes del sistema.

- Tarjetas de Expansión: Como tarjetas de video (GPU), tarjetas de sonido y tarjetas de red.
- Sistemas de Refrigeración: Incluyen ventiladores y disipadores de calor para evitar el sobrecalentamiento.

Definición de Servidor

Un servidor es un equipo informático diseñado para almacenar, procesar y distribuir datos o servicios a otros dispositivos llamados clientes. A diferencia de un computador convencional, un servidor suele operar de manera continua y estar optimizado para manejar grandes cargas de trabajo.

Características clave de un servidor:

- Alto rendimiento en procesamiento y almacenamiento.
- Capacidad de manejar múltiples conexiones simultáneas.
- Mayor seguridad y fiabilidad en la gestión de datos.
- Uso de sistemas operativos especializados como Windows Server, Linux Ubuntu Server, CentOS, Debian, entre otros.

Tipos de Servidores

Existen diferentes tipos de servidores según su función y aplicación:

- Servidor de Archivos: Almacena y gestiona archivos compartidos en una red.
- Servidor Web: Hospeda sitios web y aplicaciones en línea. Ejemplo: Apache, Nginx.
- Servidor de Bases de Datos: Maneja y almacena grandes volúmenes de datos estructurados. Ejemplo: MySQL, PostgreSQL, Oracle.
- Servidor de Correo Electrónico: Facilita el envío y recepción de correos electrónicos. Ejemplo: Microsoft Exchange, Postfix.
- Servidor de Juegos: Permite la conexión y gestión de videojuegos en línea. Ejemplo: Minecraft Server, Counter-Strike Server.

- Servidor DNS (Sistema de Nombres de Dominio): Traduce nombres de dominio en direcciones IP.
- Servidor Proxy: Actúa como intermediario entre los usuarios y la web para mejorar la seguridad y optimizar el tráfico de red.

Usos de Servidores

Los servidores desempeñan un papel crucial en múltiples áreas:

- Empresas y Negocios: Almacenamiento de datos, gestión de empleados y comunicaciones internas.
- Internet y Web Hosting: Funcionamiento de páginas web, blogs y comercio electrónico.
- Educación: Gestión de plataformas de aprendizaje virtual y almacenamiento de recursos educativos.
- Seguridad y Redes: Protección de información y monitoreo del tráfico en redes corporativas.
- Juegos en Línea: Alojamiento de partidas multijugador y distribución de contenido digital.
- Ciencia y Tecnología: Procesamiento de datos en proyectos de inteligencia artificial y análisis de grandes volúmenes de información.

El hardware de una computadora está compuesto por diversos componentes que trabajan en conjunto para procesar y gestionar información. Los dispositivos de entrada y salida facilitan la interacción con el usuario, mientras que los servidores desempeñan un papel esencial en la conectividad y gestión de datos en redes locales e internet. Comprender su estructura y funciones es fundamental para optimizar su rendimiento y solucionar problemas técnicos de manera eficiente.

ACTIVIDADES PRÁCTICAS

Tabla

Detalle de la Práctica

Practicas	Instrucciones	Duración de la Practica	Mecanismo de Evaluación
Foro de debate	Revisar Actividades Prácticas	0:30 mm	Por rúbricas
Práctica investiga: Taller/Ensayo/Investigación	Revisar Actividades Prácticas	3:30 h	Por rúbricas
Tutorías Prácticas	Revisar Actividades Prácticas	2:00 mm	Por participación en clase
Test evaluativo	Revisar Actividades Prácticas	0:30 mm	Formativa

Nota. Elaboración propia

Consideraciones Generales

1. Lee y revisa la Guía de Estudio.
2. Observa el video de la clase magistral.
3. Participa activamente en el FORO DEBATE.
4. Revisa en el calendario el día y hora en que se realizará las actividades prácticas y la tutoría.
5. Realizar la práctica propuesta por el docente.
6. Revisa las Rúbricas acorde a la actividad prácticas.
7. Toma los principales apuntes que vayas rescatando del texto.
8. Reflexiona sobre los temas tratados en la clase.
9. Sube el documento en formato PDF en la plataforma Moodle.
10. Revisa la calificación de tus actividades prácticas.

FORO DE DEBATE

Tema: Un computador está compuesto por varios elementos que trabajan en conjunto para procesar información y facilitar diversas tareas.

- ✓ ¿Por qué crees que es importante conocer las funcionalidades de un computador, sus partes y los dispositivos de entrada y salida?

Además, los servidores cumplen un papel clave en la gestión de datos y redes.

- ✓ ¿Cuáles son los principales tipos de servidores y qué papel desempeñan en el ámbito tecnológico?

Instrucciones para Participar:

- Exponga su punto de vista acerca del tema.
- Comenta las respuestas de otros participantes.
- Promueve un ambiente respetuoso y constructivo.

PRÁCTICA INVESTIGATIVA

Taller: Ensayo sobre las Funcionalidades de un Computador y los Servidores.

Escribe un ensayo de 1 a 2 páginas donde expliques:

- ✓ Las funcionalidades básicas de un computador y cómo ayudan en el procesamiento de datos.
- ✓ Los dispositivos de entrada y salida y su papel en la comunicación con el usuario.
- ✓ Definición y tipos de servidores, así como sus aplicaciones en la industria.

Consigna:

- El trabajo de investigación debe presentarse de manera estructurada y coherente para comunicar efectivamente los hallazgos y conclusiones.
- Asegúrese de que el estilo y formato del trabajo sigan las pautas de citación y presentación establecidas por la institución.
- Utilice un lenguaje claro y evite jergas o términos técnicos innecesarios.
- Revise y corrija la redacción y la gramática antes de la presentación final (ver anexo).

PRÁCTICA TEST

Antes de realizar el test, es importante considerar varios aspectos para aprovechar al máximo la experiencia.

Consideraciones para realizar el test

1. Lee y revisa el documento.

2. Observa el video de la clase magistral.

3. Revisa en el calendario el día y hora en que se debe de realizar el Test.

4. Realiza el Test de la unidad considerando lo Siguiente:

- No cambiarte entre paginas durante la realización del Test.
- Revisa que tu conexión a internet sea estable.
- Recuerda que todo el contenido está inmerso dentro del documento base.

5. Revisa tu puntuación, de inmediato al finalizar de la actividad.

II. DATOS GENERALES DE LA UNIDAD 2

Información General

1.1.	Nivel	Primero					
1.2.	Nombre Unidad	Software para el procesamiento de Datos					
1.3.	Tema 1 – Unidad	Sistemas Operativos					
1.4.	Tema 2 – Unidad	Evolución de los SO y sus aplicaciones					
1.5.	Unidad organizacional	Unidad Profesional					
1.6.	Unidad	2					
1.7.	Total, Horas Unidad	15					
1.8.	Detalle de horas Unidad	ACD	15	AA	3	APE	12

Resultado de Aprendizaje de Unidad

Al finalizar la unidad "Software para el procesamiento de Datos" de la materia de Introducción a Sistemas, los estudiantes serán capaces de comprender y aplicar los conceptos fundamentales del software utilizado para el manejo y análisis de datos. Esto incluye el conocimiento de diversas herramientas y programas de procesamiento de datos, como hojas de cálculo, bases de datos y software de análisis estadístico. Los estudiantes podrán realizar tareas básicas de manipulación de datos, como la entrada, limpieza, transformación y visualización de datos, así como entender la importancia de la calidad de los datos y la integridad en el contexto de la toma de decisiones informadas.

Metodología

La clase virtual grabada comienza con una introducción clara sobre lo que los estudiantes aprenderán en esa sesión. Se establecen los objetivos de aprendizaje para que los estudiantes sepan qué esperar y en qué enfocarse. El contenido de la asignatura se desarrolla dividiéndolo en secciones o módulos. Se utiliza una estructura organizada y lógica para abordar cada tema, combinando diferentes enfoques pedagógicos según el contenido, como explicaciones, ejemplos, demostraciones, gráficos y diagramas.

DESARROLLO

Unidad 2: Software para procesamiento de Datos

Tema 1: Sistemas Operativos

Objetivo: Diseñar y modelar bases de datos, Los estudiantes deben aprender a diseñar y modelar bases de datos de manera efectiva, identificando las necesidades de información de una organización y traduciéndolas en estructuras de bases de datos que sean eficientes y coherentes

- Objetivos
- Interrelación con el Hardware
- Clasificación de sistemas Operativos
- Servicios básicos de un SO
- Interacción con el usuario

Un sistema operativo es un programa de computadora o software que gestiona los recursos de hardware y software de una computadora. El sistema operativo es responsable de la interacción entre el usuario y el hardware, y también proporciona un entorno para ejecutar otros programas.

Los sistemas operativos se clasifican generalmente en dos categorías: sistemas operativos monousuario y sistemas operativos multiusuario. Los sistemas operativos monousuario permiten que un solo usuario acceda a la computadora a la vez. Los sistemas operativos multiusuario permiten que varios usuarios accedan a la computadora a la vez.

Otros tipos de sistemas operativos incluyen sistemas operativos de tiempo compartido, sistemas operativos de red y sistemas operativos móviles. Los sistemas operativos de tiempo compartido permiten que varios usuarios accedan a la computadora a la vez, pero cada usuario solo tiene acceso a una parte de la computadora a la vez. Los sistemas operativos de red permiten que las computadoras se comuniquen entre sí a través de una red. Los sistemas operativos móviles están diseñados para su uso en dispositivos móviles, como teléfonos inteligentes y tabletas.

Algunos de los sistemas operativos más comunes incluyen Windows, macOS, Linux y Android. Windows es el sistema operativo más popular para computadoras de escritorio y portátiles. macOS es el sistema operativo de Apple para sus computadoras

Mac. Linux es un sistema operativo de código abierto que es popular para servidores y dispositivos móviles. Android es el sistema operativo móvil más popular.

Las funciones principales de un sistema operativo incluyen:

- Gestión de memoria: El sistema operativo se encarga de asignar la memoria a los programas que se están ejecutando.
- Gestión de archivos: El sistema operativo se encarga de crear, modificar y eliminar archivos.
- Gestión de dispositivos: El sistema operativo se encarga de interactuar con los dispositivos de hardware, como el teclado, el mouse y la pantalla.
- Gestión de procesos: El sistema operativo se encarga de ejecutar varios programas a la vez.
- Gestión de seguridad: El sistema operativo se encarga de proteger la computadora de ataques maliciosos.

El sistema operativo es una parte esencial de cualquier computadora. Sin un sistema operativo, la computadora no podría funcionar.

Objetivo

El objetivo específico de los sistemas operativos es proporcionar una relación de abstracción y gestión entre el hardware del computador y las aplicaciones de software que los usuarios van a utilizar. Aquellos sistemas tienen la tarea de coordinar y controlar todos los recursos del sistema, como la CPU, la memoria, el almacenamiento y los dispositivos de entrada/salida, de manera segura y eficaz.

Los Sistemas Operativos permiten la multitarea, esto significa que pueden efectuar múltiples programas al mismo tiempo, asignando recursos de manera equitativa y evitando conflictos, también hacen fácil la interacción con el usuario a través de interfaces gráficas o de línea de comandos, ofreciendo una experiencia de usuario rápida y segura.

Es decir, los sistemas operativos tienen como propósito principal es optimizar el uso de los recursos de hardware, garantizar la estabilidad y la seguridad del sistema, proporcionando a los usuarios un entorno de trabajo estable y accesible para efectuar aplicaciones y llevar a cabo todo tipo de tareas computacionales.

Im**agen 1: Sistemas Operativos (Objetivo)**



Tomado de: https://pandorafms.com/blog/es/historia-de-los-sistemas-operativos/

Interrelación con el Hardware

El hardware se encuentra muy relacionado con el sistema operativo, principalmente con componentes como el microprocesador del sistema, ya que cada sistema operativo esta se encuentra diseñado trabajar en un tipo de ordenador.

Los sistemas operativos actúan como un intermediario permitiendo a las aplicaciones y los usuarios una interacción de manera eficiente con el hardware oculto. Esto se logra cuando el sistema operativo se encarga de administrar los componentes del hardware, como la CPU, la memoria RAM, el almacenamiento y los dispositivos de entrada/salida.

Organiza la asignación de recursos, da prioridad a las tareas que se están realizando y garantiza que múltiples programas puedan compartir los recursos de manera correcta sin generar conflictos. Además, el sistema operativo proporciona una capa de abstracción que oculta la complejidad del hardware subyacente, lo que permite a los desarrolladores de software escribir programas sin tener que preocuparse por los detalles específicos del hardware. (Rodríguez Jiménez, 2020)

Im**agen 2: Interrelación con el Hardware**



Tomado de: https://www.innovaciondigital360.com/software/que-es-un-sistema-operativo/

Clasificación de sistemas Operativos

Sistemas operativos monousuarios

Son aquellos que solo trabaja con un usuario, diseñados para uso personal o individual, estos sistemas operativos se suelen centrar en brindar al usuario una experiencia amigable y personalizada, suelen ser más simples y menos exigentes en recursos

Sistemas operativos multiusuarios

Están diseñados para admitir la interacción y el uso simultáneo de múltiples usuarios en un mismo sistema informático, pueden brindar servicio a más de un usuario a la vez, ya sea por medio de varias terminales conectadas a la computadora o por medio de sesiones remotas en una red de comunicaciones. Estos sistemas permiten que varios usuarios accedan y utilicen recursos compartidos, como la CPU, la memoria y los dispositivos de manera concurrente. Cada usuario tiene su propia sesión de trabajo aislada, lo que asegura la privacidad y la seguridad de sus datos y procesos.

Sistemas operativos monotarea

Estos Sistemas Operativos sólo permiten una tarea o programa a la vez por usuario, esto quiere decir que los componentes de hardware se designan a ejecutar, son simples y eficientes, pero tienen limitaciones en términos de productividad y flexibilidad ya requieren que los usuarios finalicen una tarea antes de iniciar otra. Existe el caso de sistemas multiusuario y monotarea lo que significa que pueden trabajar varios usuarios, pero una sola tarea a la vez.

Sistemas operativos multitarea

Son aquéllos que le permite al usuario realizar varias labores al mismo tiempo. Estos sistemas dividen el tiempo de la CPU entre diferentes procesos, Lo que aumenta significativamente la productividad y la eficiencia, también gestionan la prioridad de los procesos, garantizando que las tareas críticas reciban recursos adecuados y manteniendo el equilibrio entre las aplicaciones en ejecución mejorando su productividad.

Sistemas operativos uniproceso

Un sistema operativo uniproceso es aquél capaz de manejar solamente un procesador de la computadora diseñados para funcionar en computadoras que tienen una única unidad central de procesamiento (CPU), de manera que si la computadora tuviese más de uno le sería inútil. Los sistemas uniproceso aún son útiles en dispositivos y aplicaciones donde no se requiere un alto rendimiento de procesamiento paralelo.

Sistemas operativos multiproceso

Se refiere al número de procesadores del sistema, que es más de uno y éste es capaz de usarlos todos para distribuir su carga de trabajo. Estos sistemas permiten la ejecución simultánea de múltiples tareas, asignando eficientemente recursos de CPU a cada proceso, lo que brinda un mayor rendimiento y mayor capacidad de respuesta en sistemas de alto rendimiento, proporcionan una experiencia más fluida en aplicaciones que requieren un procesamiento intensivo, como la edición de video o la renderización 3D

Servicios básicos de un Sistema Operativo

Interfaz de Usuario

Casi todos los sistemas operativos disponen de una interfaz de usuario, es la capa visual y funcional que conecta al usuario con la computadora, este medio nos permite ejecutar aplicaciones, gestionar archivos y configurar opciones. Esta interfaz incluye menús, iconos, ventanas, barras de herramientas lo que vuelven menos complejas la navegación y las funciones de acceso del sistema (UI, user interface), que suele tomar diferentes formas. Uno de los tipos existentes es la interfaz de línea de comandos (CLI, command-line interface) que usa comandos de texto, y por otra parte se utiliza una interfaz gráfica de usuario (GUI, graphical user interface) compuesta por ventanas.

Ejecución de Programas

El sistema permite a los usuarios aprovechar al máximo la capacidad de un computador, tiene el poder de ejecutar múltiples programas de manera concurrente, lo que permite que los programas tengan los recursos necesarios sin interferir en otros. Supervisa que todo programa debe poder terminar su ejecución, de forma normal o anormal (indicando un error).

Cuando un usuario inicia un programa, el sistema operativo carga el código del programa en la memoria RAM y este comienza a ejecutarse. Dando paso a la interacción del programa con el hardware de la computadora permitiendo realizar una serie de operaciones, como cálculos, manipulación de datos y comunicación con otros programas o dispositivos. Al ejecutar un programa se comienza la asignación de recursos, como CPU, memoria y dispositivos de entrada/salida, para que el programa funcione de forma eficiente.

Operaciones de E/S

Un programa en ejecución puede necesitar llevar a cabo operaciones de E/S o de Entrada/Salida, estos programas facilitan la comunicación entre los sistemas operativos y los dispositivos periféricos, lo que implican transferencia de datos entre la memoria principal y los dispositivos de entrada o salida Las operaciones de E/S son esenciales para permitir que los usuarios interactúen con la computadora y para que los programas puedan comunicarse con los dispositivos externos. Por cuestión de eficiencia y protección, los usuarios no pueden controlar de modo directo los dispositivos de E/S.

Una operación de entrada se refiere al ingreso de datos de fuentes externas, como teclados, ratones, sensores, dispositivos de red o incluso archivos. Ingresando datos al sistema que pasan a ser procesados por programas o almacenados para su futuro uso. Mientras que las operaciones de salida implican la emisión de datos procesados hacia dispositivos externos, como monitores, impresoras, altavoces, o a través de redes.

Manipulación del sistema de archivos

Es el que se encarga de gestionar eficazmente los recursos del almacenamiento. Permitiendo a los usuarios modificar eliminar o mover archivos o directorios según su necesidad Obviamente, los programas necesitan leer y escribir en archivos y directorios. implica utilizar comandos y funciones específicas del sistema operativo para navegar por la estructura de carpetas, cambiar permisos de acceso, realizar copias de seguridad y recuperar datos. Algunos programas incluyen mecanismos de gestión de permisos para conceder o denegar el acceso a los archivos o directorios dependiendo de quien es el propietario.

Comunicaciones

Hay muchas circunstancias en las que un proceso necesita intercambiar información con otro. Dicha comunicación puede tener lugar entre procesos que se están ejecutando en la misma computadora o entre procesos en computadoras diferentes conectadas por red. Las comunicaciones se pueden implementar utilizando memoria compartida, procedimiento en el que el sistema operativo transfiere paquetes de información entre unos procesos y otros.

Los sistemas operativos proporcionan una serie de mecanismos de comunicación, como tuberías (pipes), sockets, colas de mensajes y memoria compartida, permitiendo a los procesos compartir información de manera segura y eficiente. Así como también administran la comunicación entre dispositivos de hardware y el software, a través de controladores de dispositivos, permitiéndoles interactuar con dispositivos como impresoras, discos duros, tarjetas de red y otros periféricos de manera evidente,

Detección de errores

Es la función que se encarga de garantizar la estabilidad y confiabilidad del sistema, atreves de mecanismos como el control de paridad en la memoria, la verificación de sumas de comprobación en archivos y la monitorización de la integridad de los datos durante las transferencias. Los errores pueden producirse en el hardware del procesador y de memoria, en un dispositivo de E/So en los programas de usuario. Para cada tipo de error, el sistema operativo debe llevar a cabo la operación apropiada para asegurar el funcionamiento correcto y coherente.

Asignación de recursos

Cuando hay varios usuarios, o hay varios trabajos ejecutándose al mismo tiempo, este servicio se encarga de distribuir eficientemente los recursos disponibles, como la CPU, la memoria, el almacenamiento y los dispositivos de entrada/salida, entre los diferentes procesos y aplicaciones que compiten por ellos. El sistema operativo gestiona muchos tipos diferentes de recursos; algunos pueden disponer de código de software especial que gestionen su asignación, mientras que otros pueden tener código que gestione de forma mucho más general su solicitud y liberación.

Interacción con el usuario

Esta función permite a las personas comunicarse y controlar las operaciones del sistema de manera correspondiente En un entorno de sistema operativo, dicha interacción se produce a través de interfaces de usuario, que pueden ser de línea de comandos o gráficas las interfaces de línea de comandos, como la terminal en sistemas Unix o el símbolo del sistema en Windows, permiten a los usuarios ingresar comandos de texto para ejecutar tareas específicas, proporcionando un alto grado de control y precisión.

Las interfaces gráficas de usuario (GUI) ofrecen una experiencia más visual, permitiendo a los usuarios interactuar con el sistema a través de ventanas, iconos y

elementos gráficos. Estas interfaces son las más comunes en sistemas operativos modernos como Windows, macOS y diversas distribuciones de Linux.

La interacción con el usuario también implica la gestión de dispositivos de entrada, como teclados, ratones, pantallas táctiles y micrófonos, así como dispositivos de salida como pantallas, altavoces y dispositivos de impresión. Los sistemas operativos deben garantizar que estos dispositivos se integren de manera efectiva y proporcionen una experiencia fluida y receptiva para el usuario, brindan. Mecanismos de retroalimentación y notificaciones, como mensajes de error, ventanas emergentes y sonidos, para informar al usuario sobre el estado de las operaciones y posibles problemas, facilitando la gestión de archivos y aplicaciones, permitiendo a los usuarios crear, abrir, editar y eliminar elementos de manera intuitiva. (fernandez, 2005)

ACTIVIDADES PRÁCTICAS

Tabla

Detalle de la Práctica

Practicas	Instrucciones	Duración de la Practica	Mecanismo de Evaluación
Foro de debate	Revisar Actividades Prácticas	0:30 mm	Por rúbricas
Práctica investiga: Taller/Ensayo/Investigación	Revisar Actividades Prácticas	3:30 h	Por rúbricas
Tutorías Prácticas	Revisar Actividades Prácticas	2:00 mm	Por participación en clase
Test evaluativo	Revisar Actividades Prácticas	0:30 mm	Formativa

Nota. Elaboración propia

Consideraciones Generales

1. Lee y revisa la Guía de Estudio.
2. Observa el video de la clase magistral.
3. Participa activamente en el FORO DEBATE.
4. Revisa en el calendario el día y hora en que se realizará las actividades prácticas y la tutoría.
5. Realizar la práctica propuesta por el docente.
6. Revisa las Rúbricas acorde a la actividad prácticas.
7. Toma los principales apuntes que vayas rescatando del texto.
8. Reflexiona sobre los temas tratados en la clase.
9. Sube el documento en formato PDF en la plataforma Moodle.
10. Revisa la calificación de tus actividades prácticas.

FORO DE DEBATE

Tema: Actualmente nuestros sistemas operativos son multitarea y multiproceso, especifique el ¿Por qué?

Instrucciones para Participar:

- Exponga su punto de vista acerca del tema.
- Comenta las respuestas de otros participantes.
- Promueve un ambiente respetuoso y constructivo.

PRÁCTICA INVESTIGATIVA

Taller: Clasificación de los Sistemas operativos e interacción con usuarios.

Consigna:

- El trabajo de investigación debe presentarse de manera estructurada y coherente para comunicar efectivamente los hallazgos y conclusiones.
- Asegúrese de que el estilo y formato del trabajo sigan las pautas de citación y presentación establecidas por la institución.
- Utilice un lenguaje claro y evite jergas o términos técnicos innecesarios.
- Revise y corrija la redacción y la gramática antes de la presentación final (ver anexo).

PRÁCTICA TEST

Antes de realizar el test, es importante considerar varios aspectos para aprovechar al máximo la experiencia.

Consideraciones para realizar el test

1. Lee y revisa el documento.

2. Observa el video de la clase magistral.

3. Revisa en el calendario el día y hora en que se debe de realizar el Test.

4. Realiza el Test de la unidad considerando lo Siguiente:

- No cambiarte entre paginas durante la realización del Test.
- Revisa que tu conexión a internet sea estable.
- Recuerda que todo el contenido está inmerso dentro del documento base.

5. Revisa tu puntuación, de inmediato al finalizar de la actividad.

Tema 2: Evolución de los Sistemas Operativos y sus aplicaciones

Objetivo: Analizar la evolución de los sistemas operativos y sus aplicaciones, comprendiendo su desarrollo histórico, características, tipos y funciones, con el propósito de reconocer su impacto en la computación moderna y su importancia en la gestión eficiente de los recursos informáticos.

Un Sistema Operativo (SO) es un conjunto de programas que gestiona los recursos de hardware y software de una computadora o dispositivo. A lo largo del tiempo, los sistemas operativos han evolucionado, adaptándose a las nuevas necesidades de los usuarios y la tecnología.

Familia Unix

Unix es una de las familias de sistemas operativos más antiguas y fundamentales en la historia de la informática. Creado en los años 70 por Ken Thompson y Dennis Ritchie en los laboratorios Bell, Unix se caracteriza por ser:

- **Multitarea y multiusuario**: Permite que múltiples usuarios trabajen simultáneamente en un mismo sistema.
- **Sistema basado en jerarquía de archivos**: Organiza la información en un esquema estructurado.
- **Altamente seguro y estable**: Usado en servidores, supercomputadoras y sistemas críticos.
- **Base de sistemas modernos**: Linux, macOS y muchas variantes derivan de Unix.

Ejemplo de sistemas Unix: **Solaris, BSD, AIX, HP-UX**.

Sistema Operativo Windows

Microsoft Windows es uno de los sistemas operativos más utilizados a nivel mundial. Fue lanzado en 1985 como una interfaz gráfica sobre MS-DOS y ha evolucionado con el tiempo.

Características principales:

- **Interfaz gráfica intuitiva**: Facilita el uso de la computadora con ventanas, iconos y menús.

- **Compatibilidad con una amplia variedad de hardware y software**.
- **Uso en entornos empresariales y domésticos**.
- **Integración con la nube y servicios en línea** (OneDrive, Microsoft 365).
- **Versiones populares:** Windows XP, Windows 7, Windows 10, Windows 11.

Apple (macOS e iOS)

Apple ha desarrollado su propia línea de sistemas operativos, destacando dos principales:

macOS (antes OS X)

- Sistema operativo exclusivo de computadoras Mac.
- Basado en Unix, con una interfaz optimizada para usuarios creativos y profesionales.
- Enfoque en seguridad, estabilidad y diseño intuitivo.
- Ejemplos de versiones: macOS Monterey, macOS Ventura.

iOS

- Sistema operativo móvil exclusivo de iPhone y iPad.
- Optimizado para dispositivos táctiles, con un ecosistema cerrado para mayor seguridad.
- Tienda de aplicaciones App Store con software verificado.

Linux

Linux es un sistema operativo de código abierto basado en Unix, desarrollado inicialmente por Linus Torvalds en 1991.

Características principales:

- Código abierto: Puede ser modificado y distribuido libremente.
- Seguridad y estabilidad: Usado en servidores, supercomputadoras y sistemas embebidos.
- Distribuciones variadas: Ubuntu, Debian, Fedora, CentOS, Arch Linux.
- Comunidad activa: Miles de desarrolladores contribuyen a su mejora continua.

Ejemplo de uso: Se encuentra en servidores web, infraestructura de redes y dispositivos IoT.

Android

Android es un sistema operativo basado en Linux, desarrollado por Google, y es el más usado en dispositivos móviles.

Características:

- **Código abierto y altamente personalizable**.
- **Optimizado para pantallas táctiles** (tabletas, teléfonos inteligentes).
- **Ecosistema amplio de aplicaciones** en Google Play Store.
- **Soporte para múltiples fabricantes** (Samsung, Xiaomi, Huawei, Motorola).

Ejemplo de versiones: Android 10, Android 11, Android 12, Android 13.

Programas de Aplicación

Los programas de aplicación son software diseñado para realizar tareas específicas. Se ejecutan sobre el sistema operativo y pueden incluir:

- Suites de oficina: Microsoft Office, LibreOffice, Google Docs.
- Edición de imágenes: Adobe Photoshop, GIMP.
- Desarrollo de software: Visual Studio, Eclipse, PyCharm.
- Navegadores web: Google Chrome, Mozilla Firefox, Microsoft Edge.
- Reproductores multimedia: VLC, Windows Media Player.

Usos del Software

El software se clasifica en diferentes categorías según su función:

- **Software de sistema**: Gestiona el hardware y los recursos del sistema (Ejemplo: Windows, Linux).
- **Software de aplicación**: Realiza tareas específicas para el usuario (Ejemplo: Word, Excel).

- **Software de desarrollo**: Facilita la programación y el diseño de aplicaciones (Ejemplo: Python, Java).
- **Software de entretenimiento**: Juegos, streaming de video y música (Ejemplo: Netflix, Steam, Spotify).
- **Software educativo**: Plataformas de aprendizaje como Moodle, Duolingo.

Herramientas Ofimáticas

Las herramientas ofimáticas son programas diseñados para optimizar tareas de oficina, como la creación de documentos, hojas de cálculo y presentaciones.

Ejemplos de software ofimático:

Herramienta	Ejemplo	Función
Procesador de textos	Microsoft Word, Google Docs, LibreOffice Writer	Creación y edición de documentos.
Hoja de cálculo	Microsoft Excel, Google Sheets, LibreOffice Calc	Análisis de datos y cálculos numéricos.
Presentaciones	Microsoft PowerPoint, Google Slides, Prezi	Creación de diapositivas para exposiciones.
Gestión de bases de datos	Microsoft Access, MySQL, PostgreSQL	Almacenamiento y gestión de información estructurada.
Correo electrónico	Microsoft Outlook, Gmail, Thunderbird	Comunicación y gestión de correos electrónicos.

La evolución de los sistemas operativos ha permitido que las computadoras y dispositivos móviles sean más accesibles, eficientes y seguros. Desde Unix hasta los modernos sistemas como Windows, macOS, Linux y Android, cada plataforma ha contribuido a la digitalización de múltiples sectores. El software de aplicación y las herramientas ofimáticas facilitan la productividad, permitiendo a los usuarios ejecutar tareas esenciales en el ámbito laboral, educativo y personal.

ACTIVIDADES PRÁCTICAS

Tabla

Detalle de la Práctica

Practicas	Instrucciones	Duración de la Practica	Mecanismo de Evaluación
Foro de debate	Revisar Actividades Prácticas	0:30 mm	Por rúbricas
Práctica investiga: Taller/Ensayo/Investigación	Revisar Actividades Prácticas	3:30 h	Por rúbricas
Tutorías Prácticas	Revisar Actividades Prácticas	2:00 mm	Por participación en clase
Test evaluativo	Revisar Actividades Prácticas	0:30 mm	Formativa

Nota. Elaboración propia

Consideraciones Generales

1. Lee y revisa la Guía de Estudio.
2. Observa el video de la clase magistral.
3. Participa activamente en el FORO DEBATE.
4. Revisa en el calendario el día y hora en que se realizará las actividades prácticas y la tutoría.
5. Realizar la práctica propuesta por el docente.
6. Revisa las Rúbricas acorde a la actividad prácticas.
7. Toma los principales apuntes que vayas rescatando del texto.
8. Reflexiona sobre los temas tratados en la clase.
9. Sube el documento en formato PDF en la plataforma Moodle.
10. Revisa la calificación de tus actividades prácticas.

FORO DE DEBATE

Tema: ¿Actualmente nuestros sistemas operativos son multitarea y multiproceso? Especifique el ¿Por qué?

Instrucciones para Participar:

- Exponga su punto de vista acerca del tema.
- Comenta las respuestas de otros participantes.
- Promueve un ambiente respetuoso y constructivo.

PRÁCTICA INVESTIGATIVA

Ensayo: Evolución de los Sistemas operativos.

Consigna:

- El trabajo de investigación debe presentarse de manera estructurada y coherente para comunicar efectivamente los hallazgos y conclusiones.
- Asegúrese de que el estilo y formato del trabajo sigan las pautas de citación y presentación establecidas por la institución.
- Utilice un lenguaje claro y evite jergas o términos técnicos innecesarios.
- Revise y corrija la redacción y la gramática antes de la presentación final (ver anexo).

PRÁCTICA TEST

Antes de realizar el test, es importante considerar varios aspectos para aprovechar al máximo la experiencia.

Consideraciones para realizar el test

1. Lee y revisa el documento.

2. Observa el video de la clase magistral.

3. Revisa en el calendario el día y hora en que se debe de realizar el Test.

4. Realiza el Test de la unidad considerando lo Siguiente:

- No cambiarte entre paginas durante la realización del Test.
- Revisa que tu conexión a internet sea estable.
- Recuerda que todo el contenido está inmerso dentro del documento base.

5. Revisa tu puntuación, de inmediato al finalizar de la actividad.

III. DATOS GENERALES DE LA UNIDAD 3

Información General

1.1.	Nivel	Primero					
1.2.	Nombre Unidad	Introducción a Programación y Desarrollo de software					
1.3.	Tema 1 – Unidad	Introducción a Programación					
1.4.	Tema 2 – Unidad	Lenguajes de Programación					
1.5.	Unidad organizacional	Unidad Profesional					
1.6.	Unidad	1					
1.7.	Total, Horas Unidad	30					
1.8.	Detalle de horas Unidad	ACD	15	AA	3	APE	12

Resultado de Aprendizaje de Unidad

Al finalizar esta unidad, los estudiantes serán capaces de utilizar herramientas fundamentales de software para el procesamiento de datos, incluyendo bases de datos y programas de análisis estadístico. Demostrarán habilidades prácticas en el manejo de herramientas como SQL, Excel y Python, aplicando técnicas para la extracción, transformación y carga de datos.

Metodología de Unidad

La Metodología de se basa en un enfoque activo y colaborativo. Incluye sesiones de clase estructuradas en torno a temas específicos, lecturas y tareas, estudio de casos, proyectos prácticos y debates en clase. Los estudiantes también se involucran en trabajos en grupo

y mantienen diarios de reflexión. Se fomenta la participación en debates y preguntas y respuestas, y se proporcionan recursos en línea para una comprensión más profunda. La evaluación se realiza mediante pruebas, proyectos y una evaluación final, con retroalimentación continua para guiar el aprendizaje.

DESARROLLO

Unidad 3: Introducción a Programación y Desarrollo de software

Tema 1: Introducción a Programación

Objetivo: Comprender los principios fundamentales de la programación, explorando conceptos básicos como algoritmos, estructuras de control y lenguajes de programación, con el propósito de desarrollar habilidades en la resolución de problemas computacionales y la creación de software eficiente.

La programación es el proceso de crear un programa informático, que es un conjunto de instrucciones que le dicen a una computadora qué hacer. Los programadores informáticos utilizan lenguajes de programación para escribir programas informáticos.

La programación es una habilidad esencial para cualquier persona que quiera trabajar en el campo de la informática. Los programadores informáticos son responsables de crear los softwares que utilizamos en nuestras vidas diarias, desde las aplicaciones que utilizamos en nuestros teléfonos hasta los sistemas operativos que utilizan las computadoras (Fernández, 2020).

La programación es una disciplina compleja que requiere una comprensión de los fundamentos de la informática, como la lógica, la teoría de la computación y la estructura de datos. También requiere habilidades de resolución de problemas y pensamiento crítico.

Los programadores informáticos suelen trabajar en equipos con otros programadores, diseñadores y analistas de sistemas. También pueden trabajar de forma independiente, creando sus propios programas o trabajando como consultores.

La programación es una profesión gratificante que ofrece muchas oportunidades de crecimiento y desarrollo. Los programadores informáticos tienen la oportunidad de trabajar en una amplia gama de proyectos, desde aplicaciones móviles hasta sistemas empresariales. También tienen la oportunidad de trabajar en nuevas tecnologías, como la inteligencia artificial y la realidad virtual (Tejado, 2021).

Algunos de los beneficios de la programación incluyen:

- Es una habilidad valiosa que puede abrirte muchas puertas profesionales.

- Es una disciplina desafiante que puede ayudarte a desarrollar tus habilidades de resolución de problemas y pensamiento crítico.

- Es una profesión gratificante que te permite crear cosas nuevas.

Introducción

La programación es muy amplia y existen muchas formas de programar. Es tan amplia, que a muchas personas les estresa aprender programación. Sin embargo, cada programador tiene un estilo diferente de programar, y en términos generales, a eso se le llama paradigma.

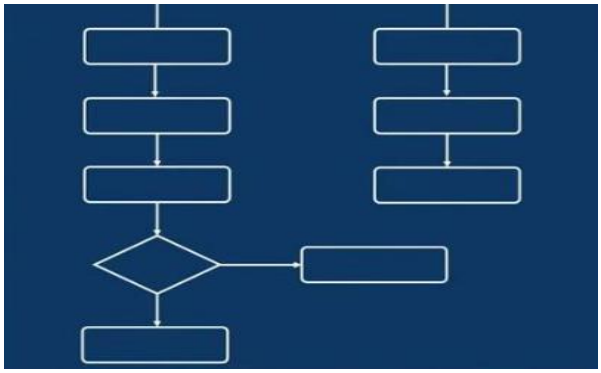
El paradigma con el que nos enseñan a programar, a la mayoría, es secuencial o estructurado. Es decir, las instrucciones van de arriba hacia abajo, una después de la otra. Simplemente damos una orden, luego otra, leemos unos datos, lo manipulamos con alguna operación, ponemos una condicional para validar ese resultado y, según esos resultados, mándanos una cosa u otra. De arriba hacia abajo, así aprendimos todos a programar.

Pero, cuando empiezas a trabajar con proyectos más grandes, te das cuenta que este paradigma no te ayuda mucho y allí se te complican las cosas. Vamos a poner un ejemplo muy simple: imagina que un cliente te pide una tienda en línea para vender zapatos. ¿Cómo vas hacer ese programa de arriba hacia abajo? toca pensar de una manera diferente porque tienes ciertas cosas:

- Los zapatos, que pueden tener un precio, color, una marca.
- Los productos deben poder filtrarse.
- Necesitas un carrito que se conecte con las pasarelas de pago para recibir el dinero.
- Tienes a los usuarios que van a hacer las compras.

Son muchas cosas que no pueden escribirse de arriba hacia abajo. Y aun así, como es la única manera en la que sabemos programar, lo hacemos de esa forma. Y cómo también he pasado por allí, sé que es muy difícil cambiar la mentalidad.

Imagen 1: Introducción a Programación



Fuentes: https://ed.team/blog/que-es-la-programacion-orientada-objetos

¿Cuál es el paradigma más usado?

El paradigma más usado en el mundo, (no el mejor, porque eso es relativo), es la programación orientada a objetos (POO). Es el más usado porque cada uno de estos elementos que necesita el sistema o la aplicación, como el carrito, el producto, el usuario, entre otros, es un objeto en este paradigma. Estos objetos tienen sus propios datos y su propio comportamiento. Por ejemplo:

Los zapatos son un objeto, y, por lo tanto, tienen estas dos propiedades:

- **Datos**: marca, precio, nombre.
- **Funcionalidad**: pueden ser comprados o agregados al carrito.

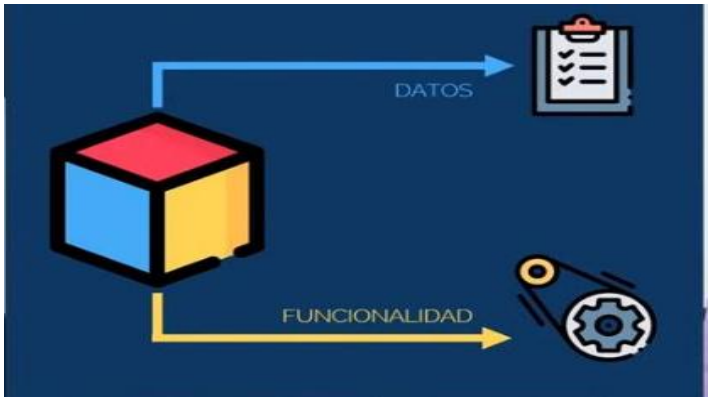
Los usuarios también se dividen igual:

- **Datos**: su nombre, número de tarjeta de crédito.
- **Funcionalidad**: comprar los productos.

Lo mismo pasa con el carrito:

- **Datos**: productos agregados al carrito o que usuarios están en el carrito.
- **Funcionalidad**: mandar esa orden de compra a la pasarela de pago, para que se procese y recibir el mensaje si el pago fue exitoso.

Imagen 2: Paradigmas de Programación



Fuentes: https://ed.team/blog/que-es-la-programacion-orientada-objetos

Cada uno de estos elementos en los que vamos dividiendo el sistema, es un objeto, y los objetos, tienen datos y funcionalidades. Esto es algo que repetiré muchísimo en esta materia, así que no lo vas a olvidar.

Con la Programación Orientada a Objetos pasamos de tener un código de arriba hacia abajo, en el que las funcionalidades están mezcladas y son difíciles de separar o escalar, a un sistema en el que tenemos los elementos (Objetos) separados y se comunican entre ellos:

- El usuario se comunica con el producto para comprarlo.
- El producto se comunica con el carrito para ser agregado.
- El carrito se comunica con la pasarela de pago y con el usuario.

Con la Programación Orientada a Objetos pasamos de tener un código de arriba hacia abajo en el que las funcionalidades están mezcladas y son difíciles de separar o escalar, a un sistema en el que tenemos los elementos (Objetos) separados y se comunican entre ellos:

- El usuario se comunica con el producto para comprarlo.
- El producto se comunica con el carrito para ser agregado.
- El carrito se comunica con la pasarela de pago y con el usuario.

De esta manera es más fácil manejar y mantener un sistema y hacerlo crecer. Si luego necesitáramos otra funcionalidad, podemos agregar otro objeto, o incluso, agregarle atributos o funcionalidad a los objetos que ya existen.

Atributos y métodos

Como se vio antes, los objetos tienen datos y funcionalidad y en la POO se les llama de esta manera:

- Datos → Atributos
- Funcionalidad → Métodos

Cada objeto tiene sus atributos y sus métodos. Te pondré un caso de vida real para que quede mucho más claro. Estamos programando la aplicación de cursos de sistemas, y queremos crear usuarios, entonces hacemos un proceso llamado abstracción. Significa pensar los atributos y métodos que debería de tener este usuario para la aplicación.

Luego de la reflexión, llegamos a la conclusión de que nuestro usuario debe tener: nombres, apellidos, contraseñas y premium (sería un valor que puede ser verdadero o falso).

A través de este mismo proceso de abstracción, definimos sus métodos: iniciar sesión, cerrar sesión, editar su perfil y contraseña, pasar a premium o publicar un artículo en la comunidad.

Imagen 3: Atributos y Métodos



Fuentes: https://ed.team/blog/que-es-la-programacion-orientada-objetos

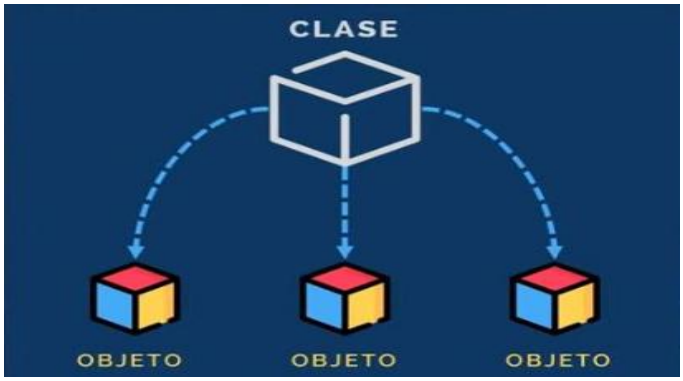
Obviamente en el proceso podemos daros cuenta que necesitamos también del atributo país, género, fecha de nacimiento y los agregamos. Ya tenemos la estructura y los agregamos sin problemas. Estos objetos se crean en código, obviamente, pero te imaginas que, para cada vez que un usuario quiera registrarse Al sistema (que son miles), ¿llamemos al programador y lo pongamos a escribir código para el usuario? no sería eficiente. Tendríamos una cola de cientos de usuarios esperando que el programador termine con el actual, para pasar al siguiente.

¿Cómo resolvemos este problema? usamos algo llamado clase.

¿Qué es una clase?

La clase es una plantilla, un molde, que tiene esa estructura básica del objeto (atributos: datos y métodos: funcionalidad). En el ejemplo anterior, no creamos realmente el objeto-usuario, creamos la plantilla (la clase). Entonces, cada vez que una persona se registra en el Sistema, realmente está usando la clase que ya creamos y que ya está en el código, para crear nuevos objetos (usuarios).

Imagen 4: Clases



Fuentes: https://ed.team/blog/que-es-la-programacion-orientada-objetos

Ese proceso de crear objetos a partir de una plantilla, llamada clase, se llama instanciar. Por eso es que cada uno de esos objetos, también se les llama instancia, y de esa manera, es que, con una sola clase, podemos crear decenas o cientos de usuarios sin tener que escribir código nuevamente. Solamente escribimos una vez la plantilla (Gómez-Díaz, 2022).

En este punto, debes tener claro, que un objeto tiene datos (atributos) y funcionalidad (métodos) y que, a través de una clase, podemos crear varios objetos. ¿Cómo es el proceso de usar estos objetos en una aplicación real?

Pilares de la POO

La programación orientada a objetos como paradigma, se basa en cuatro pilares fundamentales: abstracción, encapsulamiento, polimorfismo y herencia. Estos términos son la base de la POO y al comienzo, puede ser un poco confuso entenderlo (Carla M. V., 2021).

- **Encapsulación**. La implementación y el estado de cada objeto se mantienen de forma privada dentro de un límite definido o clase. Otros objetos no tienen acceso a esta clase o la autoridad para realizar cambios, pero pueden llamar a una lista de funciones o métodos públicos. Esta característica de ocultación de datos proporciona una mayor seguridad al programa y evita la corrupción de datos no intencionada.

- **Abstracción**. Los objetos solo revelan mecanismos internos que son relevantes para el uso de otros objetos, ocultando cualquier código de implementación innecesario. Este concepto ayuda a los desarrolladores a realizar cambios y adiciones más fácilmente a lo largo del tiempo.

- **Herencia**. Se pueden asignar relaciones y subclases entre objetos, lo que permite a los desarrolladores reutilizar una lógica común sin dejar de mantener una jerarquía única. Esta propiedad de OOP obliga a un análisis de datos más completo, reduce el tiempo de desarrollo y asegura un mayor nivel de precisión.

- **Polimorfismo**. Los objetos pueden adoptar más de una forma según el contexto. El programa determinará qué significado o uso es necesario para cada ejecución de ese objeto, reduciendo la necesidad de duplicar código.

Maquinas Programables

Una máquina programable es una máquina que puede ser controlada por un programa informático. Las máquinas programables se utilizan en una amplia gama de aplicaciones, desde la fabricación hasta la automatización de procesos.

Existen muchos tipos diferentes de máquinas programables. Algunos de los tipos más comunes incluyen:

- **Controladores lógicos programables (PLC):** Los PLC son dispositivos electrónicos que se utilizan para controlar procesos industriales.
- **Robótica**: La robótica es el uso de robots para automatizar tareas
- **Maquinas CNC**: Las máquinas CNC son máquinas herramienta que se utilizan para fabricar piezas de precisión.
- **Automatización de edificios**: La automatización de edificios utiliza tecnología para controlar los sistemas de un edificio, como la iluminación, la calefacción y la refrigeración.

Las máquinas programables ofrecen una serie de ventajas, que incluyen:

- **Flexibilidad**: Las máquinas programables se pueden reprogramar para realizar diferentes tareas.
- **Precisión**: Las máquinas programables pueden realizar tareas con una precisión constante.
- **Eficiencia**: Las máquinas programables pueden ayudar a aumentar la eficiencia de los procesos.

Las máquinas programables se utilizan en una amplia gama de industrias, que incluyen:

- **Fabricación**: Las máquinas programables se utilizan en la fabricación de una amplia gama de productos, desde automóviles hasta productos electrónicos.
- **Logística**: Las máquinas programables se utilizan en la logística para automatizar tareas como el almacenamiento y el transporte de mercancías.
- **Saludcare**: Las máquinas programables se utilizan en atención médica para automatizar tareas como la administración de medicamentos y las pruebas de diagnóstico.
- **Comercio minorista**: Las máquinas programables se utilizan en el comercio minorista para automatizar tareas como la venta y el cumplimiento de pedidos.

Las máquinas programables son una tecnología importante que se utiliza en una amplia gama de aplicaciones.

Programas de computadora

Los programas informáticos son conjuntos de instrucciones que le dicen a una computadora qué hacer. Los programas informáticos se utilizan para una amplia gama de tareas, desde jugar juegos hasta procesar datos.

Los programas informáticos se pueden clasificar en dos tipos principales: software de sistema y software de aplicación.

Software de sistema: El software de sistema es el software que se utiliza para controlar el funcionamiento de una computadora. Algunos ejemplos de software de sistema incluyen el sistema operativo, el controlador de dispositivo y el administrador de archivos.

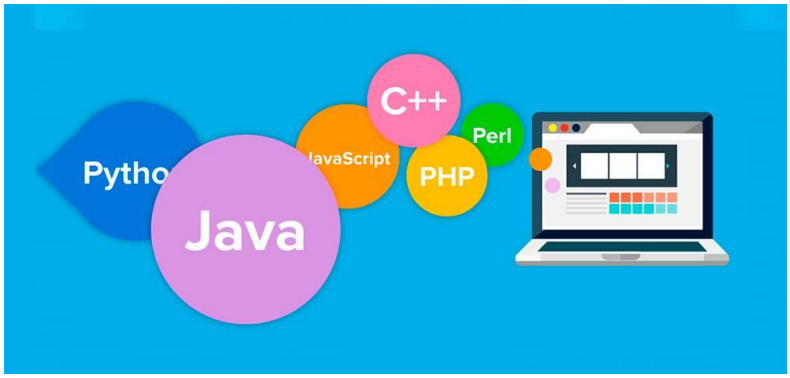
Software de aplicación: El software de aplicación es el software que se utiliza para realizar tareas específicas. Algunos ejemplos de software de aplicación incluyen el procesador de texto, la hoja de cálculo y el navegador web.

Los programas informáticos se pueden escribir en una variedad de lenguajes de programación. Algunos de los lenguajes de programación más comunes incluyen:

- **Python**: Python es un lenguaje de programación general que es fácil de aprender y usar.
- **Java**: Java es un lenguaje de programación orientado a objetos que es popular para el desarrollo de aplicaciones web y móviles.
- **C++**: C++ es un lenguaje de programación de alto rendimiento que es popular para el desarrollo de juegos y aplicaciones de software.
- **JavaScript**: JavaScript es un lenguaje de programación interpretado que se utiliza para el desarrollo de aplicaciones web.

Los programas informáticos se pueden distribuir de varias maneras. Algunos programas informáticos se distribuyen en forma de software comercial, que se vende a los usuarios. Otros programas informáticos se distribuyen de forma gratuita, como el software de código abierto.

Imagen 5: Lenguajes de programación



Fuentes: https://ed.team/blog/que-es-la-programacion-orientada-objetos

Los programas informáticos son una parte integral de nuestras vidas. Los utilizamos para trabajar, jugar, aprender e interactuar con el mundo que nos rodea.

ACTIVIDADES PRÁCTICAS

Tabla

Detalle de la Práctica

Practicas	Instrucciones	Duración de la Practica	Mecanismo de Evaluación
Foro de debate	Revisar Actividades Prácticas	0:30 mm	Por rúbricas
Práctica investiga: Taller/Ensayo/Investigación	Revisar Actividades Prácticas	3:30 h	Por rúbricas
Tutorías Prácticas	Revisar Actividades Prácticas	2:00 mm	Por participación en clase
Test evaluativo	Revisar Actividades Prácticas	0:30 mm	Formativa

Nota. Elaboración propia

Consideraciones Generales

1. Lee y revisa la Guía de Estudio.
2. Observa el video de la clase magistral.
3. Participa activamente en el FORO DEBATE.
4. Revisa en el calendario el día y hora en que se realizará las actividades prácticas y la tutoría.
5. Realizar la práctica propuesta por el docente.
6. Revisa las Rúbricas acorde a la actividad prácticas.
7. Toma los principales apuntes que vayas rescatando del texto.
8. Reflexiona sobre los temas tratados en la clase.
9. Sube el documento en formato PDF en la plataforma Moodle.
10. Revisa la calificación de tus actividades prácticas.

FORO DE DEBATE

Tema: ¿Por qué POO?

Instrucciones para Participar:

- Exponga su punto de vista acerca del tema.
- Comenta las respuestas de otros participantes.
- Promueve un ambiente respetuoso y constructivo.

PRÁCTICA INVESTIGATIVA

Ensayo: La Importancia de la Programación y las Máquinas Programables

Escribe un ensayo de 1 a 2 páginas en el que expliques:

- ✓ ¿Qué es la programación y cuáles son sus principales objetivos?
- ✓ ¿Qué son las máquinas programables y cómo han evolucionado en la historia de la computación?
- ✓ ¿Cómo influyen los programas de computadora en diferentes industrias?

Consigna:

- El trabajo de investigación debe presentarse de manera estructurada y coherente para comunicar efectivamente los hallazgos y conclusiones.
- Asegúrese de que el estilo y formato del trabajo sigan las pautas de citación y presentación establecidas por la institución.
- Utilice un lenguaje claro y evite jergas o términos técnicos innecesarios.
- Revise y corrija la redacción y la gramática antes de la presentación final (ver anexo).

PRÁCTICA TEST

Antes de realizar el test, es importante considerar varios aspectos para aprovechar al máximo la experiencia.

Consideraciones para realizar el test

1. Lee y revisa el documento.

2. Observa el video de la clase magistral.

3. Revisa en el calendario el día y hora en que se debe de realizar el Test.

4. Realiza el Test de la unidad considerando lo Siguiente:

- No cambiarte entre paginas durante la realización del Test.
- Revisa que tu conexión a internet sea estable.
- Recuerda que todo el contenido está inmerso dentro del documento base.

5. Revisa tu puntuación, de inmediato al finalizar de la actividad.

Tema 2: Lenguajes de Programación

Objetivo: Analizar los lenguajes de programación, su clasificación, características y aplicaciones, con el propósito de comprender su funcionamiento y seleccionar el lenguaje más adecuado según los requerimientos del desarrollo de software.

Lenguajes de Programación más usados

Un lenguaje de programación es un conjunto de reglas y convenciones que se utilizan para crear programas informáticos. Los programas informáticos son instrucciones que se le dan a una computadora para que realice una tarea específica.

Los lenguajes de programación se utilizan para crear una amplia gama de software, desde aplicaciones web y móviles hasta sistemas operativos y aplicaciones científicas.

Los lenguajes de programación más usados en la actualidad son los siguientes:

JavaScript: Es el lenguaje de programación más utilizado en el mundo, y es el principal lenguaje de scripting para el desarrollo web. Es un lenguaje de programación de alto nivel, interpretado y orientado a objetos.

Python: es un lenguaje de programación de alto nivel, interpretado y multiparadigma. Es un lenguaje de programación relativamente nuevo, pero ha ganado rápidamente popularidad en los últimos años. Es un lenguaje de programación fácil de aprender y usar, y es muy versátil, por lo que se puede utilizar para una amplia gama de tareas.

Java: Es un lenguaje de programación de alto nivel, compilado y orientado a objetos. Es un lenguaje de programación muy popular, y se utiliza para una amplia gama de aplicaciones, incluyendo desarrollo web, desarrollo de aplicaciones móviles y desarrollo de software empresarial.

C#: Es un lenguaje de programación de alto nivel, compilado y orientado a objetos. Es un lenguaje de programación similar a C++, pero es más fácil de aprender y usar. Se utiliza para una amplia gama de aplicaciones, incluyendo desarrollo web, desarrollo de aplicaciones móviles y desarrollo de software empresarial.

Rust: Es un lenguaje de programación compilado, de propósito general y multiparadigma, desarrollado por la Fundación Mozilla. Se utiliza para el desarrollo de sistemas operativos, realidad virtual y navegadores web. Valorado por su seguridad y

rendimiento, especialmente en el desarrollo de sistemas de bajo nivel (Enrique E. Condor Tinoco, 2017).

Importancia de los lenguajes de programación

Los lenguajes de programación son esenciales para el desarrollo de software. Sin los lenguajes de programación, sería imposible crear aplicaciones web, móviles, sistemas operativos y aplicaciones científicas.

Los lenguajes de programación son una herramienta poderosa que permite a los programadores crear software que satisface las necesidades de los usuarios (Challenger-Pérez, Díaz-Ricardo, & Becerra-García).

Compiladores e Interpretes

Compiladores

Un compilador es un programa informático que traduce el código fuente escrito en un lenguaje de programación en código máquina. El código máquina es un conjunto de instrucciones que puede ser ejecutado directamente por la computadora.

Ventajas de los compiladores:

Rendimiento: Los programas compilados suelen tener un mejor rendimiento que los programas interpretados, ya que el código máquina es ejecutado directamente por la computadora.

Portabilidad: Los programas compilados suelen ser más portátiles que los programas interpretados, ya que el código máquina no depende de la máquina en la que se ejecuta.

Eficiencia: Los compiladores pueden optimizar el código fuente para mejorar su rendimiento y eficiencia.

Desventajas los compiladores:

Complejidad: Los compiladores suelen ser programas complejos que requieren mucho tiempo y recursos para desarrollar.

Tiempo de compilación: La compilación de un programa puede llevar mucho tiempo, especialmente para programas grandes y complejos.

Errores: Los errores en el código fuente pueden provocar errores en el código máquina, que pueden ser difíciles de depurar.

Interpretes

Un intérprete es un programa informático que ejecuta el código fuente directamente. El intérprete no traduce el código fuente en código máquina, sino que simplemente lo ejecuta paso a paso.

Ventajas los intérpretes:

- Rapidez: Los programas interpretados suelen iniciarse más rápidamente que los programas compilados, ya que no es necesario compilar el código fuente.
- Facilidad de desarrollo: Los programas interpretados son más fáciles de desarrollar que los programas compilados, ya que no es necesario compilar el código fuente cada vez que se realiza un cambio.
- Ejecución interactiva: Los intérpretes permiten la ejecución interactiva de código, lo que facilita la depuración de errores.

Desventajas los intérpretes:

- Rendimiento: Los programas interpretados suelen tener un peor rendimiento que los programas compilados, ya que el código fuente debe ser ejecutado paso a paso.
- Portabilidad: Los programas interpretados pueden ser menos portátiles que los programas compilados, ya que dependen del intérprete que se utilice.
- Eficiencia: Los intérpretes no pueden optimizar el código fuente para mejorar su rendimiento y eficiencia.

Evolución de los lenguajes de programación

La evolución de los lenguajes de programación es un campo vasto y fascinante que ha experimentado múltiples cambios a lo largo de las décadas (Solís Macias, 2023).

Aquí proporcionaré un resumen general de las principales etapas en la evolución de los lenguajes de programación:

Lenguajes de máquina y ensamblador (década de 1940):

En los primeros días de la programación, las instrucciones se daban directamente en lenguaje de máquina, que estaba estrechamente vinculado con la arquitectura de la computadora.

Se desarrollaron lenguajes ensambladores para facilitar la programación y hacerla más comprensible para los programadores, utilizando mnemotécnicos en lugar de códigos numéricos.

Fortran (década de 1950):

Fortran (Formula Translation) fue el primer lenguaje de programación de alto nivel, diseñado para cálculos científicos y de ingeniería.

Introdujo conceptos como variables, bucles y funciones, haciendo que la programación fuera más accesible y menos centrada en la arquitectura de la máquina.

LISP y COBOL (década de 1950):

LISP se destacó como el primer lenguaje de programación diseñado específicamente para la inteligencia artificial, introduciendo conceptos como listas enlazadas.

COBOL (Common Business-Oriented Language) fue desarrollado para aplicaciones comerciales y empresariales.

ALGOL (década de 1960):

ALGOL (ALGOrithmic Language) influyó en muchos lenguajes posteriores y estableció la estructura básica de bloques para la programación estructurada.

ALGOL 60 fue especialmente influyente y sirvió como base para muchos lenguajes modernos.

C y Unix (década de 1970):

El lenguaje C, desarrollado por Dennis Ritchie, se convirtió en un lenguaje poderoso y eficiente que permitió un control cercano del hardware. Unix, desarrollado en C, contribuyó a la portabilidad de los programas entre sistemas.

Lenguajes de programación orientados a objetos (década de 1980-1990):

Lenguajes como C++, Smalltalk y Java introdujeron conceptos de programación orientada a objetos (OO), facilitando la creación de software modular y reutilizable.

Scripting y lenguajes interpretados (década de 1990):

Lenguajes como Python, Perl y Ruby se volvieron populares para tareas de scripting y desarrollo rápido de prototipos. La web impulsó el uso de JavaScript para programación del lado del cliente.

Lenguajes modernos y paradigmas (a partir de la década de 2000):

Surgieron lenguajes modernos como Python, Ruby, Swift y Kotlin, enfocándose en la simplicidad, legibilidad y productividad.

Se promovieron paradigmas de programación funcional, reactiva y concurrente. Lenguajes como Rust se centran en la seguridad y el rendimiento.

Machine Learning y Data Science (a partir de la década de 2010):

El auge de la inteligencia artificial y el aprendizaje automático llevó al aumento de lenguajes como Python y R en el ámbito de la ciencia de datos y el machine learning (Martínez Vaca, 2021) .

Imagen 7: Evolución de los lenguajes



Fuente: https://www.linkedin.com/pulse/evoluci%C3%B3n-de-uso-lenguajes-programaci%C3%B3n-evolution-use-sergio

Entornos de desarrollo

Un entorno de desarrollo del lenguaje de programación (IDE) es una aplicación informática que proporciona las herramientas y la infraestructura necesarias para desarrollar software. Los IDE suelen incluir un editor de texto, un compilador o intérprete, un depurador y un conjunto de herramientas de ayuda (Paredes Bravo, 2023).

Los IDE pueden ser muy útiles para los desarrolladores de software, ya que pueden ahorrar tiempo y esfuerzo. Los IDE pueden proporcionar las siguientes ventajas:

- **Productividad:** Los IDE pueden ayudar a los desarrolladores a ser más productivos, ya que proporcionan herramientas que pueden automatizar tareas comunes, como la compilación, la depuración y la documentación.
- **Eficiencia:** Los IDE pueden ayudar a los desarrolladores a ser más eficientes, ya que proporcionan una interfaz de usuario que está diseñada para facilitar la navegación y la edición de código.
- **Inteligencia:** Los IDE pueden ayudar a los desarrolladores a ser más inteligentes, ya que pueden proporcionar información contextual sobre el código, como la sintaxis, la semántica y los errores.

Tipos de entornos de desarrollo

Los IDE se pueden clasificar en dos tipos principales:

IDE de escritorio: Los IDE de escritorio son aplicaciones que se ejecutan en un ordenador personal. Son los tipos de IDE más comunes.

IDE en la nube: Los IDE en la nube son aplicaciones que se ejecutan en un servidor remoto. Se están volviendo cada vez más populares, ya que ofrecen una serie de ventajas, como la accesibilidad desde cualquier lugar y la facilidad de escalado.

Características de los entornos de desarrollo

Los IDE suelen incluir las siguientes características:

Editor de texto: El editor de texto es la herramienta principal para escribir código. Los IDE suelen proporcionar editores de texto con características avanzadas, como la autocompletado, la resaltado de sintaxis y la comprobación de errores.

Compilador o intérprete: El compilador o intérprete es la herramienta que convierte el código fuente en código máquina. Los IDE suelen proporcionar compiladores o intérpretes integrados.

Depuración: La depuración es el proceso de encontrar y corregir errores en el código. Los IDE suelen proporcionar herramientas de depuración que pueden ayudar a los desarrolladores a identificar y corregir errores.

Herramientas de ayuda: Los IDE suelen proporcionar herramientas de ayuda que pueden ayudar a los desarrolladores a aprender sobre el lenguaje de programación y las herramientas del IDE (López Solórzano, 2023).

ACTIVIDADES PRÁCTICAS

Tabla

Detalle de la Práctica

Practicas	Instrucciones	Duración de la Practica	Mecanismo de Evaluación
Foro de debate	Revisar Actividades Prácticas	0:30 mm	Por rúbricas
Práctica investiga: Taller/Ensayo/Investigación	Revisar Actividades Prácticas	3:30 h	Por rúbricas
Tutorías Prácticas	Revisar Actividades Prácticas	2:00 mm	Por participación en clase
Test evaluativo	Revisar Actividades Prácticas	0:30 mm	Formativa

Nota. Elaboración propia

Consideraciones Generales

1. Lee y revisa la Guía de Estudio.
2. Observa el video de la clase magistral.
3. Participa activamente en el FORO DEBATE.
4. Revisa en el calendario el día y hora en que se realizará las actividades prácticas y la tutoría.
5. Realizar la práctica propuesta por el docente.
6. Revisa las Rúbricas acorde a la actividad prácticas.
7. Toma los principales apuntes que vayas rescatando del texto.
8. Reflexiona sobre los temas tratados en la clase.
9. Sube el documento en formato PDF en la plataforma Moodle.
10. Revisa la calificación de tus actividades prácticas.

FORO DE DEBATE

Tema: Lenguaje de Programación más Usado.

Instrucciones para Participar:

- Exponga su punto de vista acerca del tema.
- Comenta las respuestas de otros participantes.
- Promueve un ambiente respetuoso y constructivo.

PRÁCTICA INVESTIGATIVA

Investigar: Evolución de los lenguajes de programación.

Consigna:

- El trabajo de investigación debe presentarse de manera estructurada y coherente para comunicar efectivamente los hallazgos y conclusiones.
- Asegúrese de que el estilo y formato del trabajo sigan las pautas de citación y presentación establecidas por la institución.
- Utilice un lenguaje claro y evite jergas o términos técnicos innecesarios.
- Revise y corrija la redacción y la gramática antes de la presentación final (ver anexo).

PRÁCTICA TEST

Antes de realizar el test, es importante considerar varios aspectos para aprovechar al máximo la experiencia.

Consideraciones para realizar el test

1. Lee y revisa el documento.

2. Observa el video de la clase magistral.

3. Revisa en el calendario el día y hora en que se debe de realizar el Test.

4. Realiza el Test de la unidad considerando lo Siguiente:

- No cambiarte entre paginas durante la realización del Test.
- Revisa que tu conexión a internet sea estable.
- Recuerda que todo el contenido está inmerso dentro del documento base.

5. Revisa tu puntuación, de inmediato al finalizar de la actividad.

IV. DATOS GENERALES DE LA UNIDAD 4

Información General

1.1.	Nivel	Primero					
1.2.	Nombre Unidad	Tipos de Programación					
1.3.	Tema 1 – Unidad	Programación Estructurada y Programación orientada a objetos					
1.4.	Tema 2 – Unidad	Desarrollo de Software					
1.5.	Unidad organizacional	Unidad Profesional					
1.6.	Unidad	4					
1.7.	Total, Horas Unidad	30					
1.8.	Detalle de horas Unidad	ACD	15	AA	3	APE	12

Resultado de Aprendizaje de Unidad

Al finalizar esta unidad, los estudiantes serán capaces de utilizar herramientas fundamentales de software para el procesamiento de datos, incluyendo bases de datos y programas de análisis estadístico. Demostrarán habilidades prácticas en el manejo de herramientas como SQL, Excel y Python, aplicando técnicas para la extracción, transformación y carga de datos.

Metodología de Unidad

La Metodología de se basa en un enfoque activo y colaborativo. Incluye sesiones de clase estructuradas en torno a temas específicos, lecturas y tareas, estudio de casos, proyectos prácticos y debates en clase. Los estudiantes también se involucran en trabajos en grupo

y mantienen diarios de reflexión. Se fomenta la participación en debates y preguntas y respuestas, y se proporcionan recursos en línea para una comprensión más profunda. La evaluación se realiza mediante pruebas, proyectos y una evaluación final, con retroalimentación continua para guiar el aprendizaje. Esta metodología busca que los estudiantes comprendan los conceptos fundamentales y puedan aplicar eficazmente las metodologías de desarrollo de software en situaciones prácticas.

DESARROLLO

Unidad 4: Tipos de Programación

Tema 1: Programación Estructurada y Programación orientada a objetos

Objetivo: Comparar los enfoques de Programación Estructurada y Programación Orientada a Objetos (POO), analizando sus principios, características y aplicaciones, con el propósito de desarrollar habilidades en el diseño y desarrollo de software eficiente y bien estructurado.

Programación Estructurada

La programación estructurada es un paradigma de programación que se basa en la descomposición de un problema en un conjunto de subproblemas más pequeños y manejables. Los subproblemas se resuelven utilizando una combinación de estructuras de control básicas, como la secuencia, la selección y la repetición.

La programación estructurada se desarrolló en la década de 1960 como una respuesta a la creciente complejidad de los programas informáticos. Antes de la programación estructurada, los programas se solían escribir utilizando un enfoque de programación secuencial, en el que cada instrucción se ejecutaba en orden. Este enfoque podía conducir a programas difíciles de entender y mantener (DURAN MUÐO, 2007).

La programación estructurada introduce tres estructuras de control básicas:

- **Secuencia:** La estructura de secuencia indica que las instrucciones se ejecutan en orden.
- **Selección**: La estructura de selección permite al programa elegir entre dos o más rutas de ejecución, en función de una condición.
- **Repetición**: La estructura de repetición permite al programa ejecutar un conjunto de instrucciones repetidamente, en función de una condición.

La programación estructurada también introduce el concepto de modularización, que es la práctica de dividir un programa en módulos más pequeños y manejables. Los módulos se pueden diseñar y desarrollar de forma independiente, lo que facilita la comprensión y el mantenimiento del programa (Pimentel, González, & Zavala, 2024).

La programación estructurada ha demostrado ser efectiva para desarrollar software de alta calidad y es la base de muchos lenguajes de programación modernos. Lenguajes

como C, Pascal y Ada fueron diseñados originalmente con principios de programación estructurada en mente. Aunque han surgido nuevos paradigmas de programación, como la programación orientada a objetos y la programación funcional, los principios de la programación estructurada siguen siendo relevantes y se integran en muchos enfoques de desarrollo de software actuales (Luis López Goytia, 2014).

Programación Orientada a Objetos

La programación orientada a objetos (POO) es un paradigma de programación que se basa en la idea de que los programas informáticos se pueden modelar a partir de objetos del mundo real. Un objeto es una entidad que tiene propiedades y métodos. Las propiedades son datos que describen el objeto, y los métodos son operaciones que el objeto puede realizar. (Fernández, 2020)

La POO se desarrolló en la década de 1970 como una respuesta a las limitaciones de la programación estructurada. La programación estructurada se basa en la idea de que los programas informáticos se pueden dividir en un conjunto de subproblemas más pequeños y manejables. Sin embargo, la programación estructurada puede ser insuficiente para algunos problemas complejos, ya que puede ser difícil modelar objetos del mundo real utilizando estructuras de control básicas (Goicoechea, 2021).

La POO introduce una serie de conceptos nuevos que permiten a los programadores modelar objetos del mundo real de forma más efectiva.

Principios fundamentales del POO:

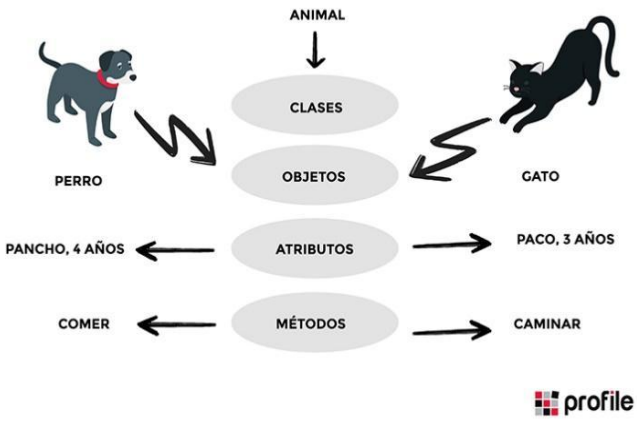
- **Clases**: Una clase es una plantilla para crear objetos. Una clase define las propiedades y métodos de un objeto.
- **Objetos**: Un objeto es una instancia de una clase. Un objeto tiene las propiedades y métodos definidos por su clase.
- **Encapsulación**: La encapsulación es el proceso de ocultar las propiedades y métodos de un objeto de los usuarios externos.
- **Herencia**: La herencia es el proceso de permitir que una clase herede las propiedades y métodos de otra clase.
- **Polimorfismo**: El polimorfismo es el proceso de permitir que objetos de diferentes clases respondan a un mismo mensaje de forma diferente.

La POO tiene una serie de ventajas, incluyendo:

- Facilidad de comprensión: Los programas orientados a objetos suelen ser más fáciles de entender que los programas estructurados, ya que utilizan una estructura que se asemeja al mundo real.
- Facilidad de mantenimiento: Los programas orientados a objetos suelen ser más fáciles de mantener que los programas estructurados, ya que se pueden dividir en objetos más pequeños.
- Reducción de errores: Los programas orientados a objetos suelen tener menos errores que los programas estructurados, ya que utilizan la encapsulación para ocultar los datos sensibles.

La POO es el paradigma de programación dominante en la actualidad. Es utilizado por la mayoría de los lenguajes de programación modernos, incluidos Java, C++, Python y JavaScript.

Imagen 1: Progamacion orientada a objetos



Fuente: https://profile.es/blog/que-es-la-programacion-orientada-a-objetos/

Diferencias entre ellas

Enfoque

- Programación Estructurada: Se enfoca en la organización del código en estructuras de control básicas como secuencias, selecciones e iteraciones.
- Programación Orientada a Objetos: Se enfoca en la organización del código en objetos que encapsulan datos y comportamiento.

Encapsulación

- Programación Estructurada: No hay encapsulación de datos y comportamiento.
- Programación Orientada a Objetos: Los datos y el comportamiento se encapsulan en objetos.

Reutilización

- Programación Estructurada: La reutilización de código es limitada.
- Programación Orientada a Objetos: La reutilización de código es alta a través de la herencia y la composición.

Flexibilidad

- Programación Estructurada: La flexibilidad es limitada para cambios en los requisitos.
- Programación Orientada a Objetos: La flexibilidad es alta para cambios en los requisitos.

Complejidad

- Programación Estructurada: Es menos compleja de aprender y usar.
- Programación Orientada a Objetos: Es más compleja de aprender y usar.

Ejemplos de lenguajes:

Programación Estructurada: C, Pascal

Programación Orientada a Objetos: Java, Python, C++

La programación estructurada es ideal para programas simples y bien definidos, mientras que la programación orientada a objetos es ideal para programas complejos y que requieren flexibilidad y reutilización de código.

Tabla1: de diferencia de POO vs PE

Característica	**Programación Estructurada**	**Programación Orientada a Objetos**
Enfoque	Estructuras de control	Objetos
Encapsulación	No	Sí
Reutilización	Baja	Alta
Flexibilidad	Baja	Alta
Complejidad	Baja	Alta
Ejemplos de lenguajes	C, Pascal	Java, Python, C++

Fuente: Elaboración Propia

En general, la programación orientada a objetos es un paradigma más potente y flexible que la programación estructurada. Sin embargo, la programación orientada a objetos también puede ser más compleja y costosa (Lasso-Cardona, 2022).

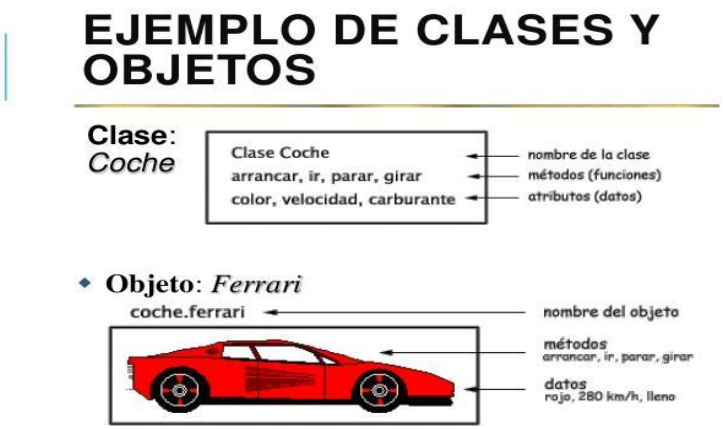
Clases y Objetos

Clases

Una clase es una plantilla para la creación de objetos de datos según un modelo predefinido. Las clases se utilizan para representar entidades o conceptos. Cada clase es un modelo que define un conjunto de variables (el estado), y métodos apropiados para operar con dichos datos (el comportamiento). Cada objeto creado a partir de la clase se denomina instancia de la clase. (BARNES & KÖLLING, 2007)

Las clases son un pilar fundamental de la programación orientada a objetos. Permiten abstraer los datos y sus operaciones asociadas al modo de una caja negra. Los lenguajes de programación que soportan clases difieren sutilmente en su soporte para diversas características relacionadas con clases. La mayoría soportan diversas formas de herencia.

Imagen 2: Clases



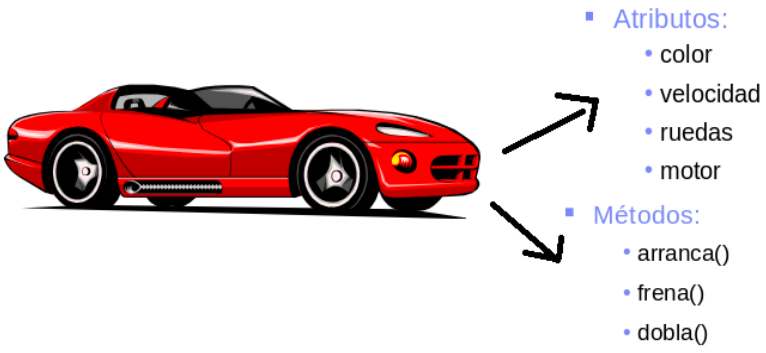
Fuente: https://aprendiendoarduino.wordpress.com/2019/09/11/clases-y-objetos-2/

Objetos

La programación orientada a objetos trabaja de esta manera. Todo el programa está construido en base a diferentes componentes (Objetos), cada uno tiene un rol específico en el programa y todos los componentes pueden comunicarse entre ellos de formas predefinidas.

Los Objetos de Software, al igual que los objetos del mundo real, también tienen características y comportamientos. Un objeto de software mantiene sus características en una o más «variables», e implementa su comportamiento con «métodos». Un método es una función o subrutina asociada a un objeto.

Imagen 3: Objetos



Fuente: https://compu2poo.wordpress.com/2013/11/29/programacion-orientada-a-objetos-poo-conceptos-basicos/

Clases y Objetos en la Programación Orientada a Objetos

En la programación orientada a objetos (POO), las clases y los objetos son dos conceptos fundamentales. (BARNES & KÖLLING, 2007)

Clases

- Son plantillas que definen la estructura y el comportamiento de un tipo de objeto.
- Especifican los atributos (datos) y los métodos (funciones) que tendrá un objeto.
- Actúan como un molde para crear objetos con características similares.

Objetos

- Son instancias de una clase.
- Contienen los atributos y métodos definidos en la clase.
- Representan entidades del mundo real en el programa.

Ejemplo

- **Clase** Persona
- Atributos: nombre, apellido, edad
- Métodos: saludar(), caminar()

Objeto persona1

- Instancia de la clase Persona

- Atributos: nombre = "Juan", apellido = "Pérez", edad = 30
- Métodos: saludar(), caminar()

Relación entre clases y objetos:

- Las clases se utilizan para crear objetos.
- Los objetos heredan las características de la clase a la que pertenecen.
- Los objetos pueden modificar sus propios atributos.

Ventajas de usar clases y objetos:

- Reutilización de código: Las clases se pueden reutilizar para crear diferentes objetos con características similares.
- Modularidad: El código se organiza en módulos independientes (clases) que facilitan su comprensión y mantenimiento.
- Encapsulación: Los datos y métodos se encapsulan en objetos, lo que permite ocultar detalles de implementación.
- Flexibilidad: La POO permite adaptar el código a cambios en los requisitos de forma más sencilla.

Ejemplos de clases y objetos en la vida real:

- Clase Coche: marca, modelo, color, velocidad
- Objeto coche1: marca = "Toyota", modelo = "Corolla", color = "rojo", velocidad = 100 km/h
- Clase Persona: nombre, apellido, edad, profesión
- Objeto persona2: nombre = "María", apellido = "Gómez", edad = 25, profesión = "doctora"

En resumen, las clases y los objetos son pilares fundamentales de la POO que permiten crear programas más eficientes, flexibles y fáciles de mantener. (BARNES & KÖLLING, 2007)

Atributos y métodos de una clase

Atributos

Los atributos son las propiedades o características de un objeto. Pueden ser de tipo primitivo, como un número o una cadena, o de tipo compuesto, como una lista o una matriz.

Los atributos se definen en la clase y se pueden acceder a ellos desde cualquier método de la clase.

Métodos

Los métodos son las acciones que puede realizar un objeto. Pueden ser métodos de acceso, que se utilizan para obtener o establecer los valores de los atributos, o métodos de comportamiento, que realizan acciones sobre el objeto.

Los métodos también se definen en la clase y se pueden acceder a ellos desde cualquier objeto de la clase.

Tipos de atributos

Los atributos pueden ser de dos tipos:

Atributos de instancia: Son los atributos que se definen en la clase y que se asignan a cada instancia de la clase.

Atributos de clase: Son los atributos que se definen en la clase y que son compartidos por todas las instancias de la clase.

Tipos de métodos

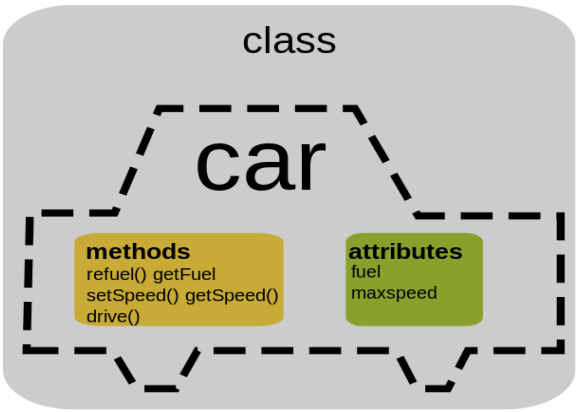
Los métodos también pueden ser de dos tipos:

Métodos de instancia: Son los métodos que se definen en la clase y que se pueden llamar desde cualquier instancia de la clase.

Métodos de clase: Son los métodos que se definen en la clase y que solo se pueden llamar desde la clase.

Los atributos y métodos son dos componentes fundamentales de una clase en la POO. Los atributos representan las propiedades o características de un objeto, mientras que los métodos representan las acciones que puede realizar un objeto.

Imagen 4: métodos y atributos



Fuente: https://edea.juntadeandalucia.es/bancorecursos/file/cb77d5d9-b0a7-4c50-98f4-28073b221392/1/esan_2021062412_9203322.zip/41_clases_y_objetos_atributos_y_metodos.html?temp.hn=true&temp.hb=true

Atributos y métodos de una clase en la Programación Orientada a Objetos (POO)

En la POO, las clases y los objetos son dos conceptos fundamentales. Las clases son plantillas que definen la estructura y el comportamiento de un tipo de objeto, mientras que los objetos son instancias de una clase que contienen los atributos y métodos definidos en la clase. (Salcedo Quiñones, 2021)

Los atributos y métodos son dos elementos clave de una clase:

Atributos

- Son las propiedades o características que definen un objeto.
- Representan los datos que un objeto necesita para funcionar.
- Ejemplo: En una clase Persona, los atributos podrían ser nombre, apellido y edad.

Métodos

- Son las acciones o comportamientos que un objeto puede realizar.
- Ejemplo: En una clase Persona, los métodos podrían ser saludar(), caminar() y comer().

Los atributos y métodos trabajan juntos para definir el comportamiento y las características de un objeto.

A continuación, se presenta una tabla que resume las principales diferencias entre atributos y métodos:

Tabla1: de diferencia de POO vs PE

Característica	Atributos	Métodos
Definición	Propiedades o características de un objeto	Acciones o comportamientos de un objeto
Tipo de dato	Variables	Funciones
Ejemplo	nombre, apellido, edad	Saludar(), caminar(), comer()

Fuente: Elaboración Propia

Visibilidad de los atributos y métodos

Los atributos y métodos pueden ser públicos, privados o protegidos:

- Públicos: Accesibles desde cualquier lugar del programa.
- Privados: Accesibles solo desde la clase en la que se definen.
- Protegidos: Accesibles solo desde la clase en la que se definen y sus clases hijas.

Ejemplo de definición de atributos y métodos en Python:

Python

```
class Persona:
    # Atributos públicos
    nombre = ""
    apellido = ""
    # Constructor (método especial para inicializar un objeto)
    def __init__(self, nombre, apellido):
        self.nombre = nombre
```

```
        self.apellido = apellido


    # Método público
    def saludar(self):
        print("Hola, mi nombre es", self.nombre, self.apellido)


# Creación de un objeto
persona1 = Persona("Juan", "Pérez")


# Acceso a un atributo público
print(persona1.nombre)


# Llamada a un método público
persona1.saludar()
```

En este ejemplo

- La clase Persona tiene dos atributos públicos: nombre y apellido.
- La clase Persona tiene un método público: saludar().
- Se crea un objeto persona1 de la clase Persona.
- Se accede al atributo público nombre del objeto persona1.
- Se llama al método público saludar() del objeto persona1.

los atributos y métodos son esenciales para definir el comportamiento y las características de los objetos en la POO.

ACTIVIDADES PRÁCTICAS

Tabla

Detalle de la Práctica

Practicas	Instrucciones	Duración de la Practica	Mecanismo de Evaluación
Foro de debate	Revisar Actividades Prácticas	0:30 mm	Por rúbricas
Práctica investiga: Taller/Ensayo/Investigación	Revisar Actividades Prácticas	3:30 h	Por rúbricas
Tutorías Prácticas	Revisar Actividades Prácticas	2:00 mm	Por participación en clase
Test evaluativo	Revisar Actividades Prácticas	0:30 mm	Formativa

Nota. Elaboración propia

Consideraciones Generales

1. Lee y revisa la Guía de Estudio.
2. Observa el video de la clase magistral.
3. Participa activamente en el FORO DEBATE.
4. Revisa en el calendario el día y hora en que se realizará las actividades prácticas y la tutoría.
5. Realizar la práctica propuesta por el docente.
6. Revisa las Rúbricas acorde a la actividad prácticas.
7. Toma los principales apuntes que vayas rescatando del texto.
8. Reflexiona sobre los temas tratados en la clase.
9. Sube el documento en formato PDF en la plataforma Moodle.
10. Revisa la calificación de tus actividades prácticas.

FORO DE DEBATE

Tema: Programación Estructurada (PE) o Programación Orientada a Objetos (POO).

Instrucciones para Participar:

- Exponga su punto de vista acerca del tema.
- Comenta las respuestas de otros participantes.
- Promueve un ambiente respetuoso y constructivo.

PRÁCTICA INVESTIGATIVA

Taller: Diferencia entre Programación Estructurada (PE) y Programación Orientada a Objetos (POO).

Consigna:

- El trabajo de investigación debe presentarse de manera estructurada y coherente para comunicar efectivamente los hallazgos y conclusiones.
- Asegúrese de que el estilo y formato del trabajo sigan las pautas de citación y presentación establecidas por la institución.
- Utilice un lenguaje claro y evite jergas o términos técnicos innecesarios.
- Revise y corrija la redacción y la gramática antes de la presentación final (ver anexo).

PRÁCTICA TEST

Antes de realizar el test, es importante considerar varios aspectos para aprovechar al máximo la experiencia.

Consideraciones para realizar el test

1. Lee y revisa el documento.

2. Observa el video de la clase magistral.

3. Revisa en el calendario el día y hora en que se debe de realizar el Test.

4. Realiza el Test de la unidad considerando lo Siguiente:

- No cambiarte entre paginas durante la realización del Test.
- Revisa que tu conexión a internet sea estable.
- Recuerda que todo el contenido está inmerso dentro del documento base.

5. Revisa tu puntuación, de inmediato al finalizar de la actividad.

Tema 2: Desarrollo de Software

¿Qué es el software?

El software, también llamado soporte lógico, es la parte intangible de un dispositivo electrónico que le da vida. Se compone de instrucciones y datos que controlan el hardware, procesan información y permiten realizar tareas específicas.

Existen diferentes tipos de software, como el software de sistema que controla el funcionamiento básico del dispositivo, el software de aplicación que permite realizar tareas específicas como navegar por internet o escribir un documento, y el software de programación que los programadores usan para crear otros programas. (Moncaut, Baum, & Robert, 2021)

El software es esencial para el funcionamiento de cualquier dispositivo electrónico, desde un computador hasta un teléfono inteligente. Sin software, el hardware no sería más que un conjunto de componentes sin vida.

Algunos ejemplos de software son los sistemas operativos (Windows, macOS, Linux), las aplicaciones (Microsoft Office, Google Chrome, Spotify) y los juegos (Fortnite, League of Legends). (martinez, 2015)

Componentes del software

- **Programas**: Son conjuntos de instrucciones que le dicen al dispositivo qué hacer. Algunos ejemplos de programas son los sistemas operativos, las aplicaciones y los juegos.
- **Datos**: Son la información que el software procesa y almacena. Algunos ejemplos de datos son documentos, imágenes, música y videos.
- **Documentación**: Son las instrucciones y guías que ayudan a los usuarios a entender cómo usar el software.

Tipos de software

- **Software de sistema**: Es el software que controla el funcionamiento básico del dispositivo. Algunos ejemplos de software de sistema son los sistemas operativos, los controladores de dispositivos y las utilidades.

- **Software de aplicación**: Es el software que permite a los usuarios realizar tareas específicas. Algunos ejemplos de software de aplicación son los procesadores de texto, las hojas de cálculo, los navegadores web y los juegos.
- **Software de programación**: Es el software que los programadores usan para crear otros programas. Algunos ejemplos de software de programación son los lenguajes de programación, los compiladores y los IDEs.

Imagen 1: Desarrollo de Software



Fuente: https://coins-colombia.com/2023/03/25/beneficios-del-desarrollo-de-software-a-la-medida/

Importancia del software

El software es esencial para el funcionamiento de cualquier dispositivo electrónico. Sin software, el hardware no sería más que un conjunto de componentes sin vida. El software nos permite realizar una gran variedad de tareas, desde navegar por internet hasta jugar videojuegos. (Chaves, 2006)

Ciclo de vida del Desarrollo de Software

El ciclo de vida del desarrollo de software abarca el desarrollo de nuevo software desde la etapa de planificación inicial hasta la implementación y el mantenimiento a largo plazo. Es una herramienta de mapeo que ayuda a los desarrolladores de software a medida que crean un nuevo software.

Diferentes metodologías SDLC se adaptarán mejor a diferentes tipos de proyectos. Cada modelo enfatiza los diferentes beneficios de SDLC.

¿Por qué es importante SDLC?

El software suele ser desarrollado por equipos de ingenieros y desarrolladores. Es fundamental que cada persona que trabaje en el proyecto siga el mismo proceso. Sin esta consistencia, sería casi imposible desarrollar un software exitoso y entregarlo al cliente a tiempo. (GÓMEZ, MARÍN, & DÍAZ, 2014)

SDLC le da a cada proyecto un framework individualizado para trabajar dentro. Mantiene a todo el equipo de desarrollo en la misma página para que el proceso permanezca ordenado y eficiente y resulte en productos de software de alta calidad y bajo costo.

Imagen 2: Ciclo de vida del Desarrollo de Software



Fuente: https://vidasoft.es/blog/desarrollo-producto/ciclo-de-vida-del-desarrollo-de-software-fases-ejemplos-modelos-y-mejores-practicas/

Fases del ciclo de vida del software

El ciclo de vida del software (SDLC) es un marco que describe las etapas involucradas en el desarrollo de software. Este proceso ayuda a garantizar que el software se desarrolle de manera eficiente y efectiva, y que cumpla con las necesidades de los usuarios.

Las fases del SDLC pueden variar según el modelo específico, pero generalmente se dividen en las siguientes etapas:

Planificación

- Definición de los objetivos del proyecto de software.

- Creación de un plan para alcanzar los objetivos, incluyendo la estimación del tiempo, recursos y presupuesto.
- Identificación de los riesgos y las estrategias para mitigarlos.

Análisis

- Comprensión de las necesidades de los usuarios y los requisitos del software.
- Recopilación de información sobre el entorno en el que se utilizará el software.
- Análisis de los datos y la información para definir las funcionalidades del software.

Diseño

- Creación de un diseño detallado del software, incluyendo la arquitectura del sistema, la interfaz de usuario y la base de datos.
- Especificación de las características y funcionalidades del software.
- Diseño de los diferentes módulos del software.

Implementación

- Escritura del código del software en un lenguaje de programación específico.
- Prueba del código para detectar y corregir errores.
- Integración de los diferentes módulos del software.

Pruebas

- Pruebas del software para asegurarse de que cumple con los requisitos y funciona como se esperaba.
- Realización de diferentes tipos de pruebas, como pruebas unitarias, pruebas de integración, pruebas de sistema y pruebas de aceptación.
- Corrección de errores encontrados durante las pruebas.

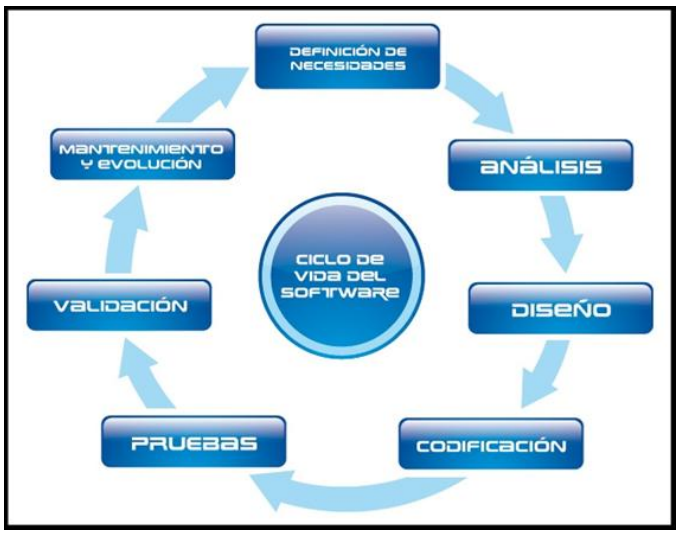
Implementación

- Implementación del software en producción y ponerlo a disposición de los usuarios.
- Entrenamiento a los usuarios sobre cómo usar el software.
- Soporte técnico a los usuarios.

Mantenimiento

- Corrección de errores que se encuentran después de la implementación del software.
- Realización de mejoras en el software para agregar nuevas funcionalidades o mejorar las existentes.
- Actualización del software para mantenerlo compatible con los nuevos sistemas operativos y tecnologías.

Imagen 3: Ciclo de vida del Desarrollo de Software



Fuente: https://inlogiq.com/diferentes-modelos-de-ciclos-de-vida-del-software/

Cada una de estas fases es importante para el desarrollo exitoso del software. La planificación ayuda a garantizar que el proyecto se desarrolle de manera organizada y eficiente. El análisis ayuda a comprender las necesidades de los usuarios y a definir

los requisitos del software. El diseño ayuda a crear un software que sea fácil de usar y que cumpla con las necesidades de los usuarios. La implementación implica escribir el código del software y probarlo para asegurarse de que funciona correctamente. Las pruebas ayudan a identificar y corregir errores en el software. La implementación pone el software a disposición de los usuarios y el mantenimiento ayuda a garantizar que el software siga funcionando correctamente y se actualiza con nuevas funcionalidades.

Modelos de ciclo de Vida

Un modelo de ciclo de vida del software (SDLC) es un marco que describe las etapas involucradas en el desarrollo del software. El modelo SDLC ayuda a garantizar que el software se desarrolle de manera eficiente y efectiva, y que cumpla con las necesidades de los usuarios.

Existen varios modelos SDLC diferentes, cada uno con sus propias ventajas y desventajas. Algunos de los modelos SDLC más comunes incluyen:

Modelo en cascada

El modelo en cascada es un método de gestión que divide un proyecto en distintas fases secuenciales. Todas ellas funcionan de forma lineal, es decir, que cada parte del proyecto se completa antes de empezar con la siguiente. Este modelo es simple y fácil de usar, ya que todo está bien organizado y las fases no pueden mezclarse entre sí. (Gamboa, 2018)

Ventajas

- Es fácil de entender y seguir.
- Es adecuado para proyectos pequeños y bien definidos.

Desventajas

- No es flexible y puede ser difícil adaptarse a cambios en los requisitos.
- No es adecuado para proyectos grandes y complejos.

Modelo en V

El modelo en V está inspirado en el método cascada. Se basa en la gestión de proyectos de forma lineal, pero consta de una fase descendente y otra ascendente.

- Fase descendente: Tareas de diseño y desarrollo.
- Fase ascendente: Control de calidad.

Ventajas

- Es más flexible que el modelo en cascada.
- Permite una mejor detección de errores en las primeras etapas del desarrollo.

Desventajas

- Puede ser más complejo de implementar que el modelo en cascada.
- No es adecuado para proyectos con requisitos cambiantes.

Modelo de prototipos

El modelo de prototipo permite al usuario obtener una visión previa de las principales características del software. Debe ser construido en poco tiempo y la aplicación representará los aspectos del proyecto que serán visibles para el cliente final.

Una vez testeada por los usuarios, aparecerán necesidades que se habían pasado por alto y esto conlleva a la realización de nuevas versiones hasta que se desarrolle el sistema o producto final.

Ventajas del modelo de prototipado

- Permite obtener comentarios de los usuarios en las primeras etapas del desarrollo.
- Puede ayudar a mejorar la calidad del software final.
- Puede reducir el riesgo de errores y problemas.
- Puede ayudar a ahorrar tiempo y dinero.

Desventajas del modelo de prototipado

- Puede ser más costoso que otros modelos SDLC.
- Puede no ser adecuado para proyectos complejos.

- Puede ser difícil de mantener el prototipo actualizado con los cambios en los requisitos.

Modelo en Espiral

El modelo de desarrollo en espiral se organiza en distintas fases. La espiral conforma un conjunto de actividades, en la que cada bucle o iteración representa un conjunto. En un primer momento, se definen los objetivos con el fin de empezar con la fase de desarrollo. A continuación, se evalúan las posibles mejoras, finalizando con la fase de mantenimiento. (Olivera & Alonso, 2021)

Ventajas

- Es fácil de entender y seguir.
- Es adecuado para proyectos pequeños y bien definidos.

Desventajas

- No es flexible y puede ser difícil adaptarse a cambios en los requisitos.
- No es adecuado para proyectos grandes y complejos.

Modelo de desarrollo incremental

El modelo incremental se diferencia por dividir sus tareas en iteraciones, trabajando objetivos en pequeños tramos específicos. Estas iteraciones permiten una fácil administración y están sujetas a cambios y modificaciones según las necesidades que puedan ir surgiendo.

Las herramientas más conocidas que utilizan este tipo de modelo de gestión de proyectos de software son Scrum y Kanban, también llamadas Metodologías Ágiles.

Métodos Ágiles

Los métodos ágiles plantean métodos de gestión de proyectos menos rígidos, donde la interacción del cliente es clave. Estos métodos mejoran enormemente la productividad y adquieren las siguientes características:

- Entregas frecuentes y revisión del trabajo por pasos.
- Adaptación a posibles cambios tras recibir la interacción del cliente.

- Organización de reuniones, donde se plantean avances y se hace balance de los resultados.
- Mejoras del proyecto de forma continua.

Entre los métodos agiles más destacados tenemos:

Scrum - El Método Scrum se caracteriza en trabajar por ciclos o Sprints. Es una metodología que exige equipos multidisciplinares, donde cada miembro del equipo puede cumplir varias tareas. Este método consta de una jerarquía de roles, el equipo se reúne diariamente con el fin de aunar criterios y realizar las adaptaciones necesarias**.** Scrum es un marco ágil para el desarrollo, mantenimiento y entrega de software, que se basa en la iteración incremental y la adaptación continua. Es un enfoque colaborativo que involucra a equipos auto-organizados para entregar valor de manera continua al cliente. (Trigás Gallego, 2012)

ACTIVIDADES PRÁCTICAS

Tabla

Detalle de la Práctica

Practicas	Instrucciones	Duración de la Practica	Mecanismo de Evaluación
Foro de debate	Revisar Actividades Prácticas	0:30 mm	Por rúbricas
Práctica investiga: Taller/Ensayo/Investigación	Revisar Actividades Prácticas	3:30 h	Por rúbricas
Tutorías Prácticas	Revisar Actividades Prácticas	2:00 mm	Por participación en clase
Test evaluativo	Revisar Actividades Prácticas	0:30 mm	Formativa

Nota. Elaboración propia

Consideraciones Generales

1. Lee y revisa la Guía de Estudio.
2. Observa el video de la clase magistral.
3. Participa activamente en el FORO DEBATE.
4. Revisa en el calendario el día y hora en que se realizará las actividades prácticas y la tutoría.
5. Realizar la práctica propuesta por el docente.
6. Revisa las Rúbricas acorde a la actividad prácticas.
7. Toma los principales apuntes que vayas rescatando del texto.
8. Reflexiona sobre los temas tratados en la clase.
9. Sube el documento en formato PDF en la plataforma Moodle.
10. Revisa la calificación de tus actividades prácticas.

FORO DE DEBATE

Tema: ¿Que metodología ágil es su favorita y por qué?

Instrucciones para Participar:

- Exponga su punto de vista acerca del tema.
- Comenta las respuestas de otros participantes.
- Promueve un ambiente respetuoso y constructivo.

PRÁCTICA INVESTIGATIVA

Taller: Taller de Evolución y ciclos de Vida del software.

Consigna:

- El trabajo de investigación debe presentarse de manera estructurada y coherente para comunicar efectivamente los hallazgos y conclusiones.
- Asegúrese de que el estilo y formato del trabajo sigan las pautas de citación y presentación establecidas por la institución.
- Utilice un lenguaje claro y evite jergas o términos técnicos innecesarios.
- Revise y corrija la redacción y la gramática antes de la presentación final (ver anexo).

PRÁCTICA TEST

Antes de realizar el test, es importante considerar varios aspectos para aprovechar al máximo la experiencia.

Consideraciones para realizar el test

1. Lee y revisa el documento.

2. Observa el video de la clase magistral.

3. Revisa en el calendario el día y hora en que se debe de realizar el Test.

4. Realiza el Test de la unidad considerando lo Siguiente:

- No cambiarte entre paginas durante la realización del Test.
- Revisa que tu conexión a internet sea estable.
- Recuerda que todo el contenido está inmerso dentro del documento base.

5. Revisa tu puntuación, de inmediato al finalizar de la actividad.

V. Bibliografía

Bibliografía básica

G.M.R., W. (S.A.). Informatica Para No Iniciados. Ediciones Deusto, S.A.

Hernández Bejarano, M., & Baquero Rey, L. E. (2023). Programación Orientada A Objetos En Java: Buenas Prácticas. EDICIONES DE LA U.

Winters, T., Manshreck, T., & Wright, H. (2022). Ingeniería de Software en Google: Lecciones sobre Programación Aprendidas a lo largo del tiempo. MARCOMBO.

Trejos Buritica, O. I. (2021). Lógica de Programación: Solucionario en pseudocódigo/ Ejercicios resueltos,. EDICIONES DE LA U.

Trejos Buritica, O. I. (2023). Lógica de Programación. 2 edición. EDICIONES DE LA U .

Bibliografía de consulta

Prieto Taborda, M. A., Bermón Angarita, L., & Ramírez Castañeda, L. A. (2019). Diseño, desarrollo y evaluación de un recurso educativo digital para la introducción a la Administración de Sistemas Informáticos. Revista Virtual Universidad Católica del Norte. Obtenido de https://www.redalyc.org/journal/1942/194259585004/194259585004.pdf.

Guerrero, A. (2022). Introducción a Sistemas ERP. Universidad Pablo de Olavide. Artículo obtenido de : https://dialnet.unirioja.es/servlet/articulo?codigo=8651236

Davara, M. (1994). Introducción a los Sistemas Expertos. Revista Iberoamericana de Derecho Informático. Artículo Obtenido de: https://dialnet.unirioja.es/servlet/articulo?codigo=257148

Anexo 1: Modelo de Foro

Título del Foro:

[Nombre del Tema o Pregunta]

Desarrollo:

[Breve descripción del tema o pregunta]

Reglas del Foro:

Regla 1: Antes de participar en el debate, realiza una investigación exhaustiva sobre el tema en cuestión. Comprender los hechos, datos y argumentos relacionados con el tema fortalecerá tu posición y mejorarán la calidad de la discusión.

Regla 2: Muestra respeto hacia tus compañeros de debate, incluso si no estás de acuerdo con sus opiniones. Escucha activamente sus argumentos y evita interrupciones. Expresa tus puntos de vista de manera respetuosa y constructiva.

Regla 3: Contribuye de manera equitativa al debate, evitando la monopolización de la conversación. Respeta el tiempo asignado y asegúrate de que otros también tengan la oportunidad de expresar sus opiniones.

Anexo 2: Esquema de Ensayo

Caratula: (Escribe aquí los datos referentes a la materia, nombres completos, tema, periodo, Institución Educativa y logos)

Título:

(Escribe aquí el título de tu ensayo)

Introducción:

Desarrollo:

Conclusión:

Bibliografía

Anexo 3: Esquema de Taller/Investigación

Caratula:

(Escribe aquí los datos referentes a la materia, tema, periodo, Institución Educativa y logos)

Título:

(Escribe aquí el título principal de tu investigación)

Temas a investigar:

- **Tema 1:**
- **Tema 2:**
- **Tema 3:**

Conclusión:

Bibliografía

Anexos

Anexo 4: Rúbrica de Foro

INDICADORES	MS 5 PUNTOS	S 3 PUNTOS	PS 2 PUNTOS	PUNTAJE
Comprensión del tema	La respuesta argumentada demuestra conocimiento concreto del tema tratado.	La respuesta argumentada demuestra conocimiento general de la mayoría del tema.	La respuesta argumentada demuestra un conocimiento limitado del tema.	20 %
Cuestionamientos y criterios	Presentó las dudas e inquietudes que se le presentaron durante el desarrollo del tema, o al menos sacó diferentes conclusiones, que las expuso en el debate.	Presenta al menos 2 preguntas o argumentos, aunque demuestra que no ha comprende ni revisado en profundidad el material.	No presenta dudas ni argumentos, aunque conozca del tema, no aporta al debate.	20 %
Redacción y ortografía	No presenta errores ortográficos ni de redacción.	Presenta algunos errores ortográficos y/o de redacción.	Presenta muchos errores de redacción y de ortografía.	20 %
Interacción: relación/ confrontación con intervenciones de otros/as	Comparte activamente con de sus compañeros los temas planteados y resuelve dudas de sus compañeros con argumentos valederos en más de 2 intervenciones.	Comparte con de sus compañeros los temas planteados, argumentando en al menos de 2 intervenciones.	Plantea dudas a sus compañeros con el tema, sin embargo, no participa ni responde con argumentos lógicos. Solo interviene una vez.	20 %
Participación	Participa con más de 2 intervenciones, además de su contestación.	Participa al menos con 5 Intervenciones, además de su contestación.	No interviene en otra participación a más de la suya.	20 %
			TOTAL=	100 %

Puntaje: 5 puntos

Anexo 5: Rubrica de Ensayo

INDICACIONES	MS 10 PUNTOS	S 6 PUNTOS	PS 2 PUNTOS	PUNTAJE
Enfoque o Idea principal	La idea principal nombra el tema del ensayo y esquematiza los puntos principales a discutir.	La idea principal nombra el tema del ensayo.	La idea principal esquematiza algunos o todos los puntos a discutir, pero no menciona el tema.	20 %
Transiciones o Conectores	Una variedad bien pensada de transiciones fue usada. Estas muestran claramente como están las ideas conectadas.	Las transiciones muestran cÃ³mo están las ideas conectadas, pero hay muy poca variedad.	Algunas transiciones funcionan bien, pero la conexiÃ³n entre las ideas no es clara.	20 %
Conclusión	La conclusión es fuerte y dejó al lector con una idea absolutamente clara de la posición del autor. Un parafraseo efectivo de la idea principal empieza la conclusión.	La conclusión es evidente. La posición del autor es parafraseada en las primeras dos oraciones de la conclusión.	La posición del autor es parafraseada en la conclusión, pero no al principio de la misma.	20 %
Gramática y ortografía	El autor no comete errores de gramática ni de ortografía que distraen al lector del contenido del ensayo.	El autor comete 1 a 2 errores gramaticales u ortográficos que distraen al lector del contenido del ensayo.	El autor comete 3 a 4 errores gramaticales u ortográficos que distraen al lector del contenido del ensayo.	20 %
Fuentes	Todas las fuentes usadas para las citas, las estadísticas y los hechos son creíbles y están citadas correctamente.	Todas las fuentes usadas para las citas, las estadísticas y los hechos son creíbles y la mayoría está citada correctamente.	La mayoría de las fuentes usadas para las citas, las estadísticas y los hechos es creíble y está citada correctamente.	20 %
			TOTAL=	100 %

Puntaje: 10 puntos

Anexo 6: Rúbrica de Taller/Investigación

INDICADORES	MS 10 PUNTOS	S 6 PUNTOS	PS 2 PUNTOS	PUNTAJE
Organización de las ideas	El texto es coherente y tiene sentido toda la lectura.	Presenta ideas aisladas, no enlazadas entre sí.	El texto no tiene coherencia y las ideas no están claras.	30 %
Ortografía	Tiene 4 o menos faltas ortográficas	Tiene entre 5 y 9 faltas ortográficas.	Tiene más de 10 faltas ortográficas.	30 %
Calidad de la información presentada	La información está claramente relacionada con el tema y proporciona varios niveles de relación con sus diferentes entornos.	La información da respuesta a la consigna principal del artículo, mas no aporta niveles de relaciones de la comunicación con el entorno próximo.	La información proporcionada guarda relación con las categorías, aunque su calidad argumentativa es deficiente.	40 %
			TOTAL=	100 %

Puntaje: 10 puntos