



Área:
Servicios

Nivel:
Técnico
Básico

VICEMINISTERIO DE EDUCACIÓN ALTERNATIVA Y ESPECIAL
DIRECCIÓN GENERAL DE EDUCACIÓN ALTERNATIVA

Guía de trabajo

SISTEMAS INFORMÁTICOS

Educación Técnica Tecnológica y Productiva



ESTADO PLURINACIONAL DE
BOLIVIA

MINISTERIO
DE EDUCACIÓN

**MINISTERIO DE EDUCACIÓN
EDUCACIÓN TÉCNICA TECNOLÓGICA Y PRODUCTIVA
CARRERA SISTEMAS INFORMÁTICOS
NIVEL TÉCNICO BÁSICO**

Edgar Pary Chambí
MINISTRO DE EDUCACIÓN

Viviana Mamani Laura
VICEMINISTRA DE EDUCACIÓN ALTERNATIVA Y ESPECIAL

Ximena Aguirre Calamani
DIRECTORA GENERAL DE EDUCACIÓN ALTERNATIVA

EDICIÓN, DISEÑO E ILUSTRACIÓN:
Viceministerio de Educación Alternativa y Especial
Dirección General de Educación Alternativa

Cómo citar este documento:

Ministerio de Educación. "Educación Técnica Tecnológica y Productiva - Carrera Sistemas Informáticos". La Paz, Bolivia.

Depósito legal:
4 - 1 - 362 - 2023 P.O.

Impreso:

EDITORIAL DEL ESTADO PLURINACIONAL DE BOLIVIA 

LA VENTA DE ESTE DOCUMENTO ESTÁ PROHIBIDA

Av. Arce, Nro. 2147
www.minedu.gob.bo

Índice

Presentación	4
Orientaciones metodológicas	5
Módulo I. Taller de sistemas operativos I	6
Unidad temática N° 1. Introducción a los sistemas operativos	6
Unidad temática N° 2. Máquinas virtuales	14
Unidad temática N° 3. Administración de recursos	18
Unidad temática N° 4. Administración plataforma windows	24
Módulo II. Matemática para la informática	28
Unidad temática N° 1. Nomenclatura informática	28
Unidad temática N° 2. Lógica matemática	35
Unidad temática N° 3. Progresiones, sucesiones y series numéricas	41
Unidad temática N° 4. Álgebra booleana	49
Unidad temática N° 5. Teoría de conjuntos	53
Módulo III. Programación I	59
Unidad temática N° 1. Diseño de algoritmos	59
Unidad temática N° 2. Lenguaje de programación	68
Unidad temática N° 3. Estructuras básicas de programación	71
Módulo IV. Hardware de computadoras I	74
Unidad temática N° 1. Evolución y generación de computadoras	74
Unidad temática N° 2. Partes de una computadora	80
Bibliografía	98

Presentación

El Ministerio de Educación, a través del Viceministerio de Educación Alternativa y Especial y la Dirección General de Educación Alternativa, presenta las Guías de Trabajo en Educación Técnica Tecnológica y Productiva. Estos materiales revisten una singular relevancia, porque serán entregados por primera vez a los participantes que se encuentran desarrollando sus procesos formativos en los Centros de Educación Alternativa.

Los facilitadores de las carreras Técnicas Tecnológicas y Productivas de los Centros de Educación Alternativa lideraron la producción de las guías de trabajo, basándose en sus experiencias y en las orientaciones del currículo. Los contenidos y actividades propuestos tienen como objetivo brindar una formación que trascienda lo cognitivo, abarcando también las dimensiones del ser, saber, hacer y decidir, en el marco del Modelo Educativo Sociocomunitario Productivo, establecido en la Ley de Educación N° 070 “Avelino Siñani – Elizardo Pérez” y están vinculadas a las vocaciones y potencialidades productivas de cada región.

Estos materiales se constituyen en una herramienta para formar a las personas jóvenes y adultas, no solo para el mundo laboral, sino también para ser actores de cambio, promotores de la innovación y generadores de emprendimientos productivos, contribuyendo a la reactivación económica.

Es fundamental destacar el papel dinámico que desempeñan las Personas Jóvenes y Adultas en la configuración de las transformaciones sociales. En este contexto, la formación Técnica, Tecnológica y Productiva es un tema central y prioritario, con el desafío de avanzar hacia la reactivación económica, la igualdad social y la eliminación de la pobreza. Todo ello se aborda desde un enfoque transformador e inclusivo para una educación plural.

Finalmente, este documento se configura como una herramienta de orientación, punto de partida esencial para el desarrollo de los procesos formativos. Los facilitadores deben enriquecer, contextualizar los contenidos y las propuestas de actividades según su experiencia profesional y las demandas particulares de los participantes.

Edgar Pary Chambi

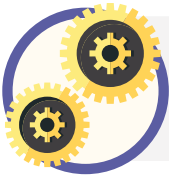
MINISTRO DE EDUCACIÓN

Orientaciones para uso del texto

Para aprovechar al máximo esta guía y lograr el desarrollo de las actividades propuestas, utilizamos la siguiente iconografía que indica el inicio de los momentos metodológicos y las actividades correspondientes.



Objetivo holístico: orienta el proceso formativo articulado a las dimensiones Ser, Saber, Hacer y Decidir.



Práctica: conocimientos previos a partir de nuestra experiencia y realidad, antes de abordar los contenidos.



Teoría: comprendemos conceptos y categorías que posibiliten profundizar el debate que nos proponga cada Unidad Temática.



Valoración: nos apropiamos de criterios que nos permitan profundizar en nuestra reflexión y análisis de la realidad a partir de los contenidos.



Producción: promovemos la aplicación creativa del conocimiento, donde los participantes compartirán los resultados de su proceso formativo.



Actividades: desarrollamos actividades que incluyan consignas concretas y precisas que faciliten la internalización de los conocimientos adquiridos.



Escanear código QR: nos invita a explorar temáticas complementarias a los contenidos desarrollados. Al escanearlo, podremos acceder a una variedad de recursos audiovisuales.

Módulo I

Taller de sistemas operativos I



Objetivo holístico del módulo

Reconocemos la importancia de los sistemas operativos y desarrollamos un conocimiento profundo e integral de los sistemas operativos, incluyendo su diseño, funcionalidad, administración de recursos y técnicas de gestión, para optimizar el rendimiento del sistema, mejorar la seguridad y la eficiencia, y solucionar problemas técnicos, a través de la instalación y configuración de diferentes sistemas operativos, y la gestión de máquinas virtuales en diversos contextos, desde personales hasta empresariales.



Unidad temática N° 1. Introducción a los sistemas operativos



Comprobemos qué sistema operativo tienen nuestros dispositivos móviles y qué sistema operativo tienen las computadoras de nuestra institución.

Para saber el sistema operativo de nuestro celular seguimos los siguientes pasos:

Configuración > Acerca del teléfono > Versión de Android y buscar versión de núcleo o versión de kernel

Detallar las características y la información que muestra el dispositivo

- Sistema operativo.
- Versión de núcleo o versión de kernel.
- Número de compilación.



Realicemos una analogía del funcionamiento de los sistemas operativos y compartámoslo con los compañeros.



Sistemas operativos

Definición de sistema operativo

Un sistema operativo es un conjunto de programas y herramientas que controlan y coordinan el funcionamiento de una computadora o dispositivo electrónico. Es el software fundamental que se encarga de gestionar los recursos de hardware,

como la memoria, la CPU, el almacenamiento y los dispositivos de entrada y salida, y proporciona una interfaz de usuario para que los usuarios interactúen con la computadora.



Historia de los sistemas operativos

La historia de los sistemas operativos ha evolucionado de manera constante desde los primeros días de la computación, en la década de 1950, hasta la actualidad. A continuación, se detallan algunas de las características más importantes y las etapas clave en el desarrollo de los sistemas operativos.

Etapa 1. Sistemas operativos Batch Processing (1950-1960)

En la década de 1950, los sistemas operativos Batch Processing fueron desarrollados para manejar grandes volúmenes de trabajo en los mainframes de IBM. Estos sistemas permitían que los trabajos se agruparan en lotes y se procesarían de forma secuencial, lo que mejoraba la eficiencia del procesamiento. Los programadores también podían enviar comandos para controlar el flujo de trabajo a través del sistema, lo que permitía la automatización de tareas.

Etapa 2. Sistemas operativos Multiprogramming (1960-1970)

En la década de 1960, los sistemas operativos Multiprogramming se desarrollaron para permitir que varias tareas se ejecutaran al mismo tiempo en una sola computadora. Estos sistemas operativos permitían que la CPU pasara de una tarea a otra rápidamente, lo que mejoraba aún más la eficiencia del procesamiento. Los sistemas operativos como el IBM OS/360 y el DEC TOPS-10 fueron pioneros en esta etapa.

Etapa 3. Sistemas operativos de tiempo compartido (1970-1980)

A finales de la década de 1960, se desarrollaron los sistemas operativos de tiempo compartido, que permitían a múltiples usuarios acceder a una sola computadora al mismo tiempo. Estos sistemas operativos tenían una arquitectura de software y hardware más sofisticada que los sistemas Batch Processing y Multiprogramming. El sistema operativo Unix, que fue desarrollado en los laboratorios Bell de AT&T, fue uno de los sistemas operativos más populares en esta época.

Etapa 4. Sistemas operativos personales (1980-1990)

A mediados de la década de 1980, surgieron los sistemas operativos personales, como MS-DOS de Microsoft y Mac OS de Apple. Estos sistemas operativos ofrecían una interfaz de usuario más amigable, lo que permitió a los usuarios interactuar con la computadora de una manera más intuitiva. Además, estos sistemas operativos permitieron a los usuarios instalar y ejecutar aplicaciones de software de terceros, lo que permitió una mayor personalización del uso de la computadora.

Etapa 5. Sistemas operativos modernos (1990-actualidad)

En la década de 1990, los sistemas operativos modernos, como Windows 95 de Microsoft y MacOS X de Apple, se lanzaron con características avanzadas y una interfaz de usuario mejorada. Los sistemas operativos móviles, como Android de Google e iOS de Apple, surgieron en la década de 2000 y revolucionaron la forma en que las personas utilizan la tecnología móvil. Además, la inteligencia artificial y la computación en la nube han impulsado aún más la eficiencia y la funcionalidad de los sistemas operativos. En la actualidad, los

sistemas operativos continúan evolucionando y mejorando constantemente para satisfacer las demandas en constante cambio de los usuarios y las empresas.

Estructura y servicios básicos del sistema operativo

La estructura de un sistema operativo se puede dividir en varias capas, cada una de las cuales proporciona diferentes servicios y funcionalidades básicas. A continuación, se describen algunas de las capas y servicios más importantes de un sistema operativo:

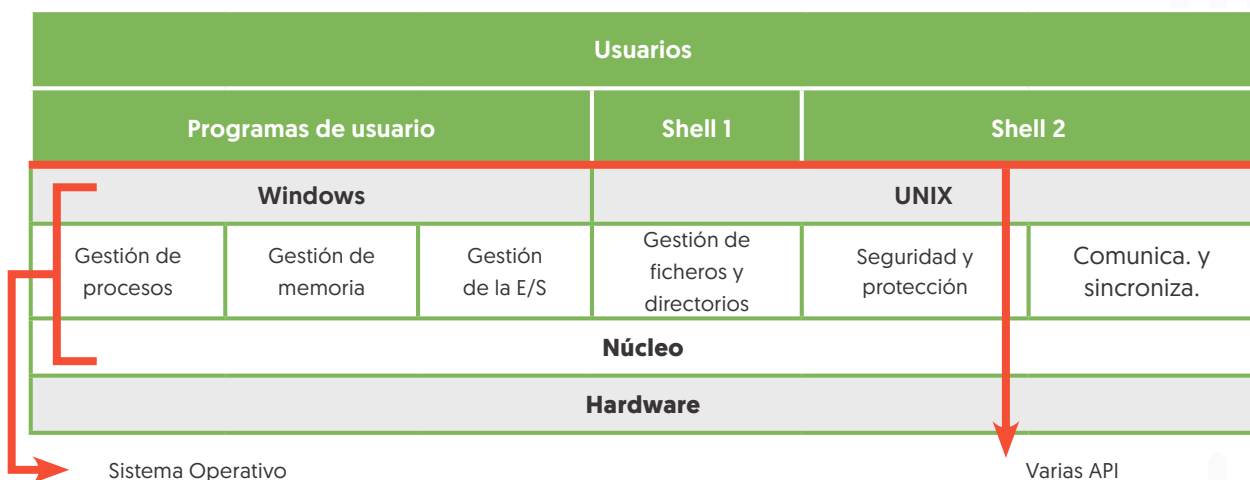
Kernel. Es la capa más baja y es responsable de interactuar directamente con el hardware del sistema. El kernel controla los dispositivos de entrada/salida, la memoria, el procesamiento y la gestión de archivos. Es el corazón del sistema operativo y proporciona servicios básicos, como la administración de procesos, la administración de memoria, la administración de archivos y la administración de dispositivos.

Servicios de administración de procesos. El sistema operativo controla y administra los procesos que se ejecutan en el sistema. Proporciona servicios para iniciar, detener y administrar los procesos, y para controlar el uso de la CPU por parte de los procesos.

Servicios de administración de memoria. El sistema operativo administra la memoria del sistema y proporciona servicios para asignar y liberar memoria, administrar la memoria virtual y compartir memoria entre procesos.

Servicios de administración de archivos. El sistema operativo administra los archivos y directorios del sistema, y proporciona servicios para crear, eliminar, abrir y cerrar archivos. También proporciona servicios de gestión de permisos y control de acceso a los archivos.

Servicios de entrada/salida. El sistema operativo administra los dispositivos de entrada/salida del sistema, y proporciona servicios para acceder a ellos y controlar su uso. Esto incluye dispositivos como teclados, ratones, discos duros, impresoras y dispositivos de red.



Servicios de seguridad. El sistema operativo proporciona servicios para garantizar la seguridad del sistema y la privacidad de los usuarios. Esto incluye servicios de autenticación, autorización y control de acceso, así como servicios de encriptación y seguridad de la red.

Shell. Es la interfaz de usuario del sistema operativo. Permite a los usuarios interactuar con el sistema operativo a través de una línea de comandos o una interfaz gráfica de usuario [GUI]. El shell acepta comandos y los transmite al kernel para su ejecución.

Recursos administrados por los sistemas operativos

Los sistemas operativos son responsables de administrar una amplia variedad de recursos en un sistema de computadora. Algunos de los recursos administrados por los sistemas operativos son:

Procesos. Los sistemas operativos administran los procesos que se ejecutan en un sistema de computadora. Los procesos son programas en ejecución que utilizan los recursos del sistema, como la CPU, la memoria y los dispositivos de entrada/salida.

Memoria. Los sistemas operativos administran la memoria del sistema, incluyendo la asignación y liberación de memoria a los procesos, la gestión de la memoria virtual y la memoria caché.

Archivos. Los sistemas operativos administran los archivos y directorios del sistema, incluyendo la creación, eliminación, apertura y cierre de archivos. También controlan el acceso y los permisos de los archivos y proporcionan servicios de búsqueda y gestión de archivos.

Dispositivos de entrada/salida. Los sistemas operativos administran los dispositivos de entrada/salida, incluyendo teclados, ratones, monitores, impresoras y dispositivos de red.

Redes. Los sistemas operativos administran la comunicación de red, incluyendo el control de acceso a la red, la configuración de la red, la conexión a Internet y la gestión de los servicios de red.

Seguridad. Los sistemas operativos administran la seguridad del sistema, incluyendo la autenticación y el control de acceso de los usuarios, la protección contra virus y malware, y la encriptación de datos.

Tiempo. Los sistemas operativos administran el tiempo del sistema, incluyendo la sincronización del reloj del sistema, la gestión de las interrupciones del temporizador y la programación de tareas en segundo plano.

Tipos de sistemas operativos

Existen varios tipos de sistemas operativos, cada uno diseñado para diferentes dispositivos y entornos. A continuación, se describen algunos de los tipos más comunes:

Sistemas operativos de escritorio. Estos sistemas operativos están diseñados para su uso en computadoras de escritorio y portátiles. Algunos ejemplos incluyen Microsoft Windows, MacOS y Linux.

Sistemas operativos móviles. Estos sistemas operativos están diseñados para su uso en dispositivos móviles, como teléfonos inteligentes y tabletas. Algunos ejemplos incluyen Android, iOS y Windows Phone.



Sistemas operativos de servidores. Estos sistemas operativos están diseñados para su uso en servidores y centros de datos. Algunos ejemplos incluyen Microsoft Windows Server, Linux y Unix.

Sistemas operativos embebidos. Estos sistemas operativos están diseñados para su uso en dispositivos embebidos, como cámaras, dispositivos médicos y sistemas de control industrial. Algunos ejemplos incluyen FreeRTOS, QNX y VxWorks.

Sistemas operativos en tiempo real. Estos sistemas operativos están diseñados para aplicaciones que requieren una respuesta en tiempo real, como sistemas de control de procesos y sistemas de seguridad. Algunos ejemplos incluyen RTOS, VxWorks y Windows Embedded Compact.

Sistemas operativos de red. Estos sistemas operativos están diseñados para su uso en enrutadores, interruptores y otros dispositivos de red. Algunos ejemplos incluyen Cisco IOS, Juniper Junos y OpenWrt.

Sistemas operativos en la nube. Estos sistemas operativos están diseñados para su uso en infraestructuras de nube, como Amazon Web Services, Microsoft Azure y Google Cloud Platform.

Procesos. Los procesos son una parte importante de los sistemas operativos, y su gestión es fundamental para el correcto funcionamiento del sistema. A continuación, se detallan con más precisión algunos aspectos clave de la gestión de procesos en los sistemas operativos.

Creación de procesos. La creación de un proceso normalmente se inicia con una acción de usuario, como hacer clic en un icono o ejecutar un comando desde la línea de comandos. Esta acción indica al sistema operativo que debe cargar el código del programa en la memoria y asignarle los recursos necesarios, como el espacio de direcciones de memoria y los descriptores de archivos. Además, el sistema operativo debe establecer valores para varios contadores, punteros y listas.

Planificación de procesos. Hay muchos algoritmos de planificación, cada uno con sus ventajas y desventajas. Algunos algoritmos dan prioridad a los procesos que requieren menos tiempo de CPU, otros pueden dar prioridad a los procesos que requieren más entrada/salida, y otros pueden simplemente rotar entre todos los procesos disponibles (round-robin). La elección del algoritmo depende en gran medida de los requerimientos específicos del sistema.

Comunicación de procesos. Los mecanismos de IPC (Inter-process communication) permiten que los procesos compartan datos, se envíen señales entre sí, o coordinen acciones. Los mecanismos de IPC incluyen tuberías (pipes), señales (signals), semáforos, memoria compartida, y colas de mensajes, entre otros.

Terminación de procesos. Cuando un proceso completa su ejecución, debe ser eliminado de manera eficiente para liberar sus recursos y mantener un rendimiento óptimo del sistema. Si un proceso se encuentra en un estado de error, se puede convertir en un proceso zombie que consume recursos sin realizar ningún trabajo útil. El sistema operativo debe identificar y eliminar estos procesos para evitar problemas.

Sincronización de procesos. En sistemas con múltiples procesos o hilos, se puede requerir que ciertas operaciones se ejecuten en un orden específico. La sincronización de procesos se utiliza para garantizar que estos requisitos se cumplan. Los mecanismos de sincronización incluyen semáforos, monitores y variables de condición.

Seguridad de procesos. Los sistemas operativos utilizan una variedad de mecanismos para proteger los procesos y sus recursos. Esto puede incluir espacios de direcciones de memoria separados para cada proceso, permisos de acceso a los archivos y a las operaciones de red, y la capacidad de limitar la cantidad de recursos que un proceso puede utilizar.

Gestión de memoria. La gestión de memoria es crucial para el rendimiento y la estabilidad del sistema operativo. El sistema operativo debe asignar memoria a los procesos cuando lo necesiten, y liberarla cuando ya no la necesiten. Además, el sistema operativo puede utilizar técnicas como el intercambio de memoria (swapping) y la paginación para maximizar la utilización de la memoria.

Estados de un proceso. Cada proceso puede tener los siguientes estados:

Nuevo (New). Este es el estado inicial de un proceso, cuando se ha emitido un comando para crear un proceso, pero la creación aún no ha finalizado. Durante este estado, el sistema operativo realiza varias tareas, incluyendo asignar un identificador único al proceso y asignar espacio en la memoria para el nuevo proceso.

Listo (Ready). Una vez que el proceso ha sido creado, se mueve al estado "Listo". Aquí, el proceso está en la memoria y listo para ejecutarse, pero está esperando su turno para obtener acceso a la

CPU. El sistema operativo mantiene una cola de todos los procesos en el estado "Listo" y utiliza un algoritmo de programación para determinar qué proceso se ejecutará a continuación.

En ejecución (Running). Cuando un proceso ha sido seleccionado por el planificador del sistema operativo y se le ha dado acceso a la CPU, el proceso pasa al estado "En ejecución". Durante este estado, el código del proceso se está ejecutando activamente.

Bloqueado (Blocked). Un proceso puede entrar en el estado "Bloqueado" si necesita esperar a que ocurra un evento específico, como una operación de entrada/salida (I/O) o la liberación de un recurso que está siendo utilizado por otro proceso. Mientras el proceso está bloqueado, no puede avanzar en su ejecución, aunque la CPU esté disponible. Una vez que se completa la operación de I/O o se libera el recurso necesario, el proceso puede pasar de nuevo al estado "Listo".

Terminado (Terminated). Cuando un proceso ha finalizado su ejecución, pasa al estado "Terminado". El sistema operativo luego realizará la tarea de limpieza, que puede incluir la liberación de la memoria utilizada por el proceso y el cierre de cualquier archivo o conexión de red que el proceso haya dejado abierta.

Suspendido (Suspended). En algunos sistemas, los procesos pueden moverse a un estado "Suspendido". Esto suele suceder si el sistema necesita liberar memoria para otros procesos. En este estado, el proceso se guarda en un almacenamiento de respaldo (normalmente el disco duro), liberando así memoria para otros procesos. Cuando el sistema tiene recursos suficientes, el proceso suspendido puede ser reanudado, es decir, movido de nuevo a memoria y al estado "Listo".

Procesos ligeros e hilos. En términos técnicos, un hilo es la unidad más pequeña de procesamiento que puede ser planificada y ejecutada por un sistema operativo. Aunque los hilos existen dentro de un proceso y comparten recursos con otros hilos en el mismo proceso, cada hilo tiene su propio contador de programa (que indica dónde se encuentra en su ejecución), su propio conjunto de registros de la CPU y su propia pila de llamadas (que se utiliza para mantener un registro de las funciones que ha llamado).

Contexto compartido. Los hilos en un mismo proceso comparten el mismo espacio de memoria, lo que significa que pueden acceder a las mismas variables y estructuras de datos. Esto puede hacer que la comunicación entre hilos sea extremadamente rápida, ya que no se requiere ningún mecanismo especial de comunicación entre procesos (IPC). Sin embargo, este compartir también puede conducir a condiciones de carrera, donde dos hilos intentan modificar la misma variable al mismo tiempo, lo que puede conducir a resultados inesperados. Para evitar estas condiciones de carrera, los programadores utilizan mecanismos de sincronización como mutexes, semáforos y monitores.

Multitarea más eficiente. Dado que los hilos dentro de un proceso comparten el mismo espacio de memoria, la creación de hilos y el cambio de contexto entre hilos es a menudo mucho más rápido

que entre procesos. Esto permite que un programa realice varias tareas en paralelo de manera más eficiente. Por ejemplo, en un servidor web, cada conexión podría ser manejada por un hilo separado, permitiendo al servidor manejar múltiples conexiones simultáneamente.

Creación y gestión más eficientes. Crear un hilo es generalmente más rápido y requiere menos recursos del sistema que crear un proceso completo. Esto se debe a que un nuevo proceso requiere su propio espacio de memoria y su propio conjunto de recursos del sistema, mientras que un hilo sólo requiere un contador de programa, un conjunto de registros y una pila de llamadas, todos los cuales son mucho menos costosos en términos de recursos del sistema.

Rendimiento y capacidad de respuesta. La multithreading puede mejorar el rendimiento de ciertas aplicaciones al permitirles realizar trabajo mientras esperan operaciones de E/S (como la lectura de un archivo o la recepción de datos de red) para completarse. También puede mejorar la capacidad de respuesta de las aplicaciones con interfaz de usuario, al permitir que se realice trabajo en segundo plano sin interrumpir la interacción del usuario.



Analicemos lo aprendido

Del análisis realizado en la práctica 1, revisemos cuál es el sistema operativo más antiguo y el sistema operativo más actualizado, con referencia del sistema operativo Android actual.



Unidad temática N° 2. Máquinas virtuales



Preguntas problematizadoras

- ¿Jugamos o conocimos alguna vez el videojuego Mario Kart [original] en una computadora? **Sí / No**
- ¿Sabemos qué es una consola de video? **Sí / No**
- ¿Podríamos diferenciar una máquina virtual de un emulador? **Sí / No**



Máquinas virtuales

Introducción

Las máquinas virtuales (MV) desempeñan un papel esencial en muchos aspectos de la informática, desde la administración de servidores hasta la seguridad y el desarrollo de software.

¿Cómo funcionan las máquinas virtuales?

Las máquinas virtuales operan mediante la creación de un entorno aislado que imita una computadora física, con su propio sistema operativo y software. que se encarga de la creación, ejecución y gestión de las MV.

Tipos de máquinas virtuales

- **Máquinas virtuales de sistema.** Proporcionan una representación virtual de todo el hardware que necesita un sistema operativo para funcionar. Esto permite a los usuarios ejecutar sistemas operativos diferentes en la misma máquina física. Cada máquina virtual proporciona un entorno completo y privado que incluye la CPU, memoria, dispositivos de entrada/salida y red, en el que se puede ejecutar un sistema operativo y sus aplicaciones como si se estuvieran ejecutando directamente en el hardware físico.
- **Máquinas virtuales de proceso.** También conocidas como máquinas virtuales de aplicación, se diseñan para ejecutar un proceso o aplicación específica. Proporcionan un entorno de ejecución que es independiente del sistema operativo y del hardware subyacentes. Esto permite que la misma aplicación se ejecute de manera idéntica en diferentes plataformas. Un ejemplo es la Máquina Virtual de Java (JVM).

Ventajas y desventajas de las máquinas virtuales

Las máquinas virtuales ofrecen muchas ventajas, que incluyen el aislamiento (si una MV falla, no afecta a las demás), la eficiencia (permite ejecutar varios sistemas operativos y aplicaciones en un solo servidor físico), la flexibilidad (permite mover y copiar MV fácilmente de un hardware a otro) y la seguridad (permite probar software y configuraciones en un entorno aislado).

Software para máquinas virtuales

Existen varios tipos de software que te permiten crear y administrar máquinas virtuales (MV). Estos se conocen como hipervisores o monitores de máquinas virtuales. Aquí te dejo algunos ejemplos:

- **VirtualBox.** Este es un software de virtualización de código abierto proporcionado por Oracle. VirtualBox es muy popular para la virtualización en equipos de escritorio debido a su facilidad de uso y a que admite una amplia gama de sistemas operativos huéspedes.
- **VMware.** VMware ofrece varios productos de virtualización, entre ellos VMware Workstation para equipos de escritorio y VMware vSphere para servidores. VMware es conocido por su robustez y es ampliamente utilizado en entornos empresariales.
- **Hyper-V.** Es el hipervisor de Microsoft incluido en las versiones Pro y Enterprise de Windows. Hyper-V también está disponible para servidores con Windows Server. Este hipervisor se integra bien con otros productos de Microsoft y es popular en entornos que ya utilizan una gran cantidad de software de Microsoft.
- **KVM (Kernel-based Virtual Machine).** Es una solución de virtualización de código abierto para Linux. KVM se integra directamente con el kernel de Linux y permite a los usuarios ejecutar múltiples máquinas virtuales con diferentes sistemas operativos.
- **QEMU.** Es un emulador de código abierto y gratuito que es capaz de emular una amplia variedad de hardware. Aunque QEMU puede funcionar por sí solo, a menudo se utiliza junto con KVM para aprovechar las capacidades de virtualización del hardware.
- **Parallels desktop.** Este es un software de virtualización popular para Mac. Permite a los usuarios de Mac ejecutar aplicaciones de Windows en su entorno macOS.
- **Docker.** Aunque Docker no es un hipervisor en el sentido tradicional, se utiliza para la virtualización a nivel de sistema operativo (conocida como contenedores). Los contenedores de Docker son más ligeros que las máquinas virtuales tradicionales y comparten el kernel del sistema operativo anfitrión.

Máquinas virtuales - hipervisor tipo 1 y 2

El término "hipervisor" se refiere al software que crea, ejecuta y gestiona máquinas virtuales (MV). Los hipervisores se clasifican en dos tipos: Tipo 1 (o hipervisor de nivel de hardware) y Tipo 2 (o hipervisor de nivel de sistema operativo).

Hipervisores tipo 1

Los hipervisores de Tipo 1 son programas de software que se ejecutan directamente en el hardware de un sistema, sin necesidad de un sistema operativo subyacente. Como resultado, tienen control directo sobre el hardware y pueden administrar eficientemente los recursos del sistema entre las máquinas virtuales. Al tener menos capas de abstracción entre las máquinas virtuales y el hardware, los hipervisores de Tipo 1 pueden proporcionar una mejor eficiencia y rendimiento que los de Tipo 2.

Algunos ejemplos de hipervisores de tipo 1 incluyen:

- **VMware ESXi.** Este es un producto empresarial que se utiliza comúnmente en entornos de centros de datos. ESXi es parte de la suite de productos de VMware vSphere, que proporciona una plataforma completa para administrar y orquestar máquinas virtuales en grandes infraestructuras.
- **Microsoft Hyper-V.** Es el hipervisor de Tipo 1 de Microsoft. Está incluido con algunas versiones de Windows Server y permite ejecutar múltiples sistemas operativos en un solo hardware.
- **Xen.** Es un hipervisor de código abierto que se utiliza a menudo en entornos de servidores y en la nube. Xen es conocido por su eficiencia y es utilizado por varias empresas de servicios en la nube, como Amazon Web Services (AWS) para su servicio EC2.

Hipervisores tipo 2

Los hipervisores de Tipo 2 se ejecutan como una aplicación dentro de un sistema operativo existente. Como resultado, no tienen acceso directo al hardware y en su lugar deben hacer llamadas al sistema operativo anfitrión para acceder a los recursos del sistema. Aunque esta capa adicional de abstracción puede introducir algo de sobrecarga y reducir el rendimiento, los hipervisores de Tipo 2 son generalmente más fáciles de instalar y usar que los de Tipo 1, y a menudo ofrecen mejor soporte para dispositivos periféricos y características de uso general.

Algunos ejemplos de hipervisores de Tipo 2 incluyen:

- **VMware workstation/fusion.** Workstation es la versión para Windows y Linux, mientras que Fusion es la versión para Mac. Estos productos están diseñados para uso en estaciones de trabajo y permiten a los usuarios ejecutar múltiples sistemas operativos en su máquina local.
- **Oracle virtualBox.** VirtualBox es un hipervisor de código abierto y gratuito que se puede instalar en una variedad de sistemas operativos anfitriones, incluyendo Windows, Linux y Mac.
- **Parallels desktop.** Es un hipervisor de tipo 2 popular entre los usuarios de Mac, ya que permite ejecutar aplicaciones de Windows en macOS.



Analicemos lo aprendido

En nuestros cuadernos realicemos el análisis de cuáles son los beneficios de las máquinas virtuales.



Apliquemos lo aprendido

1. Realicemos un debate respondiendo las siguientes preguntas:

- ¿Por qué aprender a usar máquinas virtuales ?
- ¿Qué áreas de la informática debemos conocer sobre máquinas virtuales?
- ¿Qué sistemas operativos podemos instalar en máquinas virtuales?
- ¿Qué tipo de hipervisor es VirtualBox?

2. Investiguemos y realicemos un análisis de los siguientes conceptos sobre licencias de software

- Licencia de software libre.
- Licencia de software privativo o propietario.



Unidad temática N° 3. Administración de recursos



Realicemos un análisis

Desde el administrador de tareas del sistema operativo que estamos usando, verifiquemos y mostremos al facilitador los procesos que se están realizando.

Gestión de la CPU. Aprende a monitorizar la utilización de la CPU y a administrar los procesos para evitar el uso excesivo de la CPU. Podemos usar herramientas de gestión de tareas en el sistema operativo que utilicemos.

Gestión de la memoria. Aprende a monitorizar el uso de la memoria y a ajustar la asignación de la memoria para mejorar el rendimiento. Esto podría implicar ajustar la cantidad de memoria asignada a ciertos procesos o programas.

Gestión del almacenamiento. Aprende a administrar el espacio en disco, a organizar los archivos de manera eficiente y a realizar mantenimientos regulares.



Administración de recursos

La administración de recursos es una de las tareas fundamentales de un sistema operativo. Esta gestión se realiza en varios recursos clave como la CPU (unidad central de procesamiento), la memoria, el almacenamiento y los dispositivos de entrada/salida [E/S]. Aquí está una descripción detallada de cómo se administra cada uno de estos recursos:

Administración del procesador

La administración de recursos del procesador, también conocida como planificación de procesos, es uno de los aspectos clave de un sistema operativo. Su objetivo es asignar eficientemente los recursos de la CPU entre los diferentes procesos e hilos de ejecución que se están ejecutando en un sistema.

Procesador

Estos son algunos de los aspectos clave de la administración de recursos del procesador:

Algoritmos de planificación. Estos son utilizados por el sistema operativo para determinar qué proceso o hilo se debe ejecutar en la CPU en un momento dado. Hay varios algoritmos de planificación diferentes, incluyendo el primero en llegar, primero en ser atendido (FIFO), el más corto primero (SJF), la rotación (Round Robin), la prioridad, y otros.

Multiprocesamiento. Con los sistemas de multiprocesamiento, la CPU puede ejecutar varios procesos simultáneamente, aumentando así el rendimiento y la eficiencia. Los sistemas modernos suelen tener múltiples núcleos de CPU, y los sistemas operativos modernos pueden asignar diferentes procesos o hilos a diferentes núcleos.

Interrupciones y señales. Las interrupciones son eventos que detienen temporalmente la ejecución de un proceso para que el sistema operativo pueda atender algo más importante. Las señales son una forma de interrupción que los procesos pueden usar para comunicarse entre sí o con el sistema operativo.

Tiempo compartido. Este es un método de planificación de procesos en el que la CPU se alterna entre diferentes procesos, dedicando un pequeño período de tiempo a cada uno. Esto permite a varios procesos parecer que se están ejecutando al mismo tiempo, incluso en una CPU de un solo núcleo.

Control de concurrencia. Este es un método para asegurar que los procesos que necesitan acceder a los mismos recursos (como los archivos o la memoria) puedan hacerlo de manera segura, sin interferir entre sí.

Administración de la memoria

La gestión de la memoria es una de las funciones más críticas de un sistema operativo. Su objetivo es administrar y coordinar la memoria de la computadora, asignando porciones a los programas cuando los necesitan y liberándose cuando ya no son necesarias. La gestión efectiva de la memoria puede aumentar significativamente la velocidad y la eficiencia del sistema.

Estas son algunas de las principales funciones y técnicas de gestión de memoria:

Asignación de memoria. Cuando un proceso se ejecuta, necesita espacio en la memoria para almacenar su código, sus variables y su pila de ejecución. El sistema operativo es responsable de asignar memoria a estos procesos cuando se necesitan y liberar esa memoria cuando ya no se necesitan.

Paginación. La paginación es una técnica de gestión de memoria que divide la memoria en bloques físicos de tamaño fijo llamados páginas. Cuando un proceso se ejecuta, sus necesidades de memoria se dividen en bloques de tamaño de página. Esto permite que la memoria se use de manera más eficiente, ya que se pueden asignar y desasignar páginas individuales.

Segmentación. La segmentación es otra técnica de gestión de memoria que divide la memoria en segmentos lógicos de diferentes tamaños, cada uno de los cuales tiene una tarea específica. Por ejemplo, un proceso puede tener un segmento para su código, un segmento para sus datos y un segmento para su pila. Los segmentos pueden crecer o encogerse según sea necesario, lo que proporciona una mayor flexibilidad que la paginación.

Memoria virtual. La memoria virtual permite que un proceso use más memoria de la que está físicamente disponible en el sistema. Esto se logra almacenando temporalmente los datos de la memoria en el disco cuando no se están utilizando, luego se pueden cargar de nuevo en la memoria cuando se necesitan. Esto permite que se ejecuten programas más grandes y más procesos simultáneamente, a costa de una disminución en la velocidad debido al tiempo requerido para el intercambio de datos entre la memoria y el disco.

Gestión de caché y búfer. Los sistemas operativos a menudo utilizan cachés y búferes para acelerar el acceso a la memoria. Una caché almacena temporalmente los datos que se han utilizado recientemente o que se utilizan con frecuencia, mientras que un búfer almacena temporalmente los datos que se están transfiriendo entre dos lugares.

Protección de memoria: La protección de la memoria evita que un proceso acceda a la memoria asignada a otro proceso sin permiso. Esto es esencial para la seguridad y la estabilidad del sistema, ya que impide que un proceso dañe los datos de otro proceso o del sistema operativo.

Administración de discos

La gestión de almacenamiento es otra responsabilidad crucial de un sistema operativo. En el nivel más básico, el sistema operativo controla el hardware de almacenamiento físico, pero también proporciona una estructura lógica para los datos y controla cómo se accede a los datos, cómo se almacenan, cómo se organizan y cómo se comparten. Aquí están algunos de los conceptos clave:

Sistema de archivos. En el corazón de la gestión de almacenamiento se encuentra el sistema de archivos. Este es el método que utiliza el sistema operativo para organizar y almacenar datos en un dispositivo de almacenamiento. Los sistemas de archivos pueden ser de diferentes tipos, como FAT32, NTFS, ext4, HFS+, etc., y cada uno tiene sus propias ventajas y desventajas en términos de eficiencia, seguridad y compatibilidad.

E/S de disco. Los sistemas operativos manejan todas las operaciones de entrada y salida [E/S] que involucran almacenamiento. Esto incluye la lectura y escritura de datos en el almacenamiento, así como otras operaciones como la creación, el cambio de nombre y la eliminación de archivos y directorios.

Administración de espacio. El sistema operativo es responsable de llevar un registro de qué partes del almacenamiento están en uso y qué partes están libres. Cuando se crea un nuevo archivo, el

sistema operativo debe encontrar un espacio libre en el almacenamiento para él. Cuando se elimina un archivo, el sistema operativo marca su espacio como libre.

Caché de disco y búferes. Al igual que con la gestión de la memoria, los sistemas operativos a menudo utilizan cachés y búferes para acelerar las operaciones de almacenamiento. Un caché de disco almacena los datos recientemente leídos del almacenamiento, mientras que un búfer almacena los datos que están a punto de escribirse.

Seguridad y permisos de acceso. Los sistemas operativos a menudo implementan medidas de seguridad que controlan quién puede acceder a qué datos. Esto generalmente se maneja mediante un sistema de permisos que determina quién puede leer, escribir o ejecutar cada archivo.

Redundancia y recuperación de errores. Algunos sistemas operativos implementan técnicas para proteger contra la pérdida de datos. Esto puede incluir la redundancia, donde los datos se almacenan en más de un lugar, y la recuperación de errores, donde el sistema operativo intenta recuperar o corregir los datos si se producen errores de almacenamiento.

Todos estos factores juntos permiten que un sistema operativo administre de manera eficiente el almacenamiento y aseguren que los datos estén organizados, accesibles y seguros.

Administración de entradas y salidas

La gestión de dispositivos de entrada/salida [E/S] es otra función vital de un sistema operativo. Los dispositivos de E/S pueden ser cualquier cosa, desde un mouse y un teclado hasta una impresora, un monitor o un disco duro. Estos dispositivos proporcionan la principal interfaz de interacción entre el usuario y el sistema informático.

A continuación, se muestran algunos de los aspectos clave de la gestión de dispositivos de E/S:

Controladores de dispositivos. Cada dispositivo de E/S necesita un programa específico, conocido como controlador de dispositivo, que sirve como intermediario entre el sistema operativo y el dispositivo. El controlador de dispositivo sabe cómo comunicarse con el dispositivo y traduce las solicitudes generales del sistema operativo en instrucciones específicas para ese dispositivo.

Asignación de dispositivos. Cuando un programa necesita usar un dispositivo de E/S, el sistema operativo debe decidir cómo asignar ese dispositivo. Algunos dispositivos pueden ser compartidos entre varios programas, mientras que otros pueden ser asignados exclusivamente a un solo programa en un momento dado.

Planificación de E/S. Al igual que el sistema operativo debe planificar qué proceso obtiene acceso a la CPU y cuándo, también debe planificar qué procesos obtienen acceso a los dispositivos de E/S y cuándo. Los algoritmos de planificación para E/S pueden ser tan simples como FCFS (primero en

llegar, primero en ser servido] o pueden ser más complejos, con prioridades basadas en la urgencia de la tarea, el tipo de tarea, etc.

Búferes, cachés y colas. Para mejorar el rendimiento de E/S, los sistemas operativos a menudo utilizan técnicas como los búferes, las cachés y las colas. Un búfer almacena temporalmente datos de E/S para manejar diferencias en la velocidad de operación entre la CPU y los dispositivos de E/S. Una caché almacena datos recientemente o frecuentemente utilizados para reducir los tiempos de acceso. Las colas se utilizan para mantener un registro de las solicitudes de E/S que están esperando para ser atendidas.

Manejo de errores de E/S. Los sistemas operativos también deben ser capaces de manejar errores que ocurren durante las operaciones de E/S. Esto puede incluir cosas como errores de lectura/escritura, fallos de conexión y otros problemas de hardware.

La gestión de dispositivos de E/S es una parte esencial de la funcionalidad del sistema operativo, permitiendo la interacción eficiente entre el hardware, el software y el usuario.



Analicemos lo aprendido

Partiendo de la teoría podemos analizar y explicar cuáles son los procesos que se están realizando en nuestra computadora desde el administrador de tarea de Windows.



Apliquemos lo aprendido

Podemos instalar y analizar cuáles de los siguientes navegadores consume más recursos de nuestra computadora abriendo solo una página en todos los navegadores:

	CPU	Memoria RAM	GPU
Mozilla Firefox			
Microsoft Edge			
Google Chrome			
Brave			
Opera			



Unidad temática N° 4. Administración plataforma Windows



Investiguemos desde la realidad

De las propiedades de sistema operativo Windows, analicemos la versión de qué sistema operativo tenemos, además de las especificaciones técnicas del hardware de la computadora.



Plataforma Windows

Características del sistema operativo

El sistema operativo Windows, desarrollado por Microsoft, es uno de los sistemas operativos más utilizados en el mundo. Ha habido muchas versiones a lo largo de los años, incluyendo Windows XP, Windows 7, Windows 8, Windows 10, y más recientemente, Windows 11. Cada versión ha añadido nuevas características y mejoras, pero hay algunas características comunes y fundamentales que son consistentes en todas las versiones de Windows. Aquí detallamos algunas de ellas:

Interfaz gráfica de usuario (GUI). Una de las características más notables de Windows es su interfaz gráfica de usuario. Esto incluye una barra de tareas, un menú de inicio, y la capacidad de abrir y gestionar programas en ventanas que pueden ser redimensionadas, minimizadas, maximizadas y cerradas.

Compatibilidad de software y hardware. Windows es compatible con una amplia gama de hardware y software. Muchos desarrolladores de software crean programas específicamente para Windows debido a su popularidad, y la mayoría de los fabricantes de hardware aseguran que sus productos sean compatibles con Windows.

Soporte multitarea. Windows permite ejecutar y gestionar múltiples aplicaciones a la vez. Esto permite a los usuarios cambiar fácilmente entre diferentes tareas y programas.

Sistema de archivos. Windows utiliza el sistema de archivos NTFS (New Technology File System) de forma predeterminada, aunque también es compatible con FAT32 y exFAT. NTFS es un sistema de archivos robusto que soporta permisos de archivo, compresión, cifrado, y recuperación de errores.

Gestión de red. Windows ofrece una amplia gama de características de gestión de red, permitiendo a los usuarios configurar conexiones de red, compartir archivos e impresoras en una red, y más.

Seguridad. Windows incluye una serie de características de seguridad, como Windows Defender, un programa antivirus incorporado; Firewall de Windows; Control de Cuentas de Usuario (UAC), que puede prevenir cambios no autorizados al sistema; y BitLocker, una herramienta de cifrado de disco.

Herramientas y accesorios integrados. Windows viene con una serie de herramientas y accesorios útiles, como Microsoft Edge para navegación web, Cortana para búsqueda y asistencia personal, y varias otras herramientas para tareas como visualización de imágenes, reproducción de medios, y edición de texto.

Actualizaciones regulares. Microsoft lanza regularmente actualizaciones para Windows, que incluyen correcciones de seguridad, mejoras de rendimiento, y nuevas características.

Compatibilidad con periféricos. Windows ofrece soporte nativo para una amplia gama de dispositivos periféricos, como teclados, ratones, impresoras, y cámaras.

Interoperabilidad. Windows puede interoperar con otros sistemas, como Linux y MacOS, a través de redes y protocolos estándar.

Gestión del sistema operativo

Gestionar un sistema operativo Windows implica una serie de tareas que garantizan el funcionamiento eficiente, seguro y productivo del sistema. Aquí hay un desglose de las tareas más comunes que un administrador de Windows podría realizar:

Instalación y configuración del sistema operativo. Esto implica instalar el sistema operativo Windows en nuevas máquinas, configurar los ajustes iniciales, establecer conexiones de red y crear cuentas de usuario.

Instalación y actualización de software. Los administradores de Windows también instalan y actualizan software, incluyendo navegadores web, suites de productividad, antivirus y otras aplicaciones necesarias para las operaciones comerciales o individuales.

Gestión de seguridad. Los administradores deben configurar y supervisar la seguridad del sistema operativo. Esto puede incluir la configuración de cortafuegos, la instalación y actualización de software antivirus, la gestión de permisos de usuario y el monitoreo regular de la seguridad del sistema.

Gestión de redes. Esto implica configurar y mantener conexiones de red, compartir recursos en la red [como impresoras o archivos], y resolver problemas de red.

Monitorización del sistema. Los administradores también deben supervisar el rendimiento del sistema, identificar y resolver problemas, y mantener el sistema en funcionamiento sin problemas. Las herramientas de administración del sistema y los registros de eventos pueden ayudar a realizar esta tarea.

Gestión de backups y recuperación de datos. Es esencial mantener copias de seguridad regulares de los datos del sistema y ser capaz de recuperarlos en caso de un fallo del sistema o una pérdida de datos.

Gestión de actualizaciones del sistema. Microsoft lanza regularmente actualizaciones y parches de seguridad para su sistema operativo que deben ser instalados para mantener el sistema seguro y al día.

Gestión de usuarios y grupos. Esto implica crear y administrar cuentas de usuario, establecer permisos y políticas para cada cuenta, y garantizar que los usuarios puedan acceder a los recursos que necesitan.

Gestión de discos y almacenamiento. Esto incluye la gestión de particiones de disco, la gestión del uso del espacio en disco y la configuración de las políticas de almacenamiento.

Soporte técnico. Los administradores de Windows suelen proporcionar soporte técnico a los usuarios, lo que puede incluir la resolución de problemas de software, la asistencia para el uso de aplicaciones y la resolución de problemas de hardware.

Todas estas tareas requieren un conocimiento sólido del sistema operativo Windows, así como una comprensión de los conceptos de TI en general. Además, cada organización puede tener necesidades específicas que requieran tareas de gestión adicionales.



Analicemos lo aprendido

Realicemos una valoración entre los diferentes sistemas operativos Windows y cuáles fueron los más estables y son a la fecha más aptos para los usuarios.



Apliquemos lo aprendido

Desarrollemos los siguientes conceptos:

- Partes del escritorio y el explorador de archivos.
- Comparación de FAT32, exFAT y NTFS.
- Programas alternativos a Windows Defender.
- Instalación de Windows xp en una máquina virtual.
- Instalación de Windows 7 en una máquina virtual.
- Instalación de Windows 10 en una máquina virtual.
- Instalación de Windows 11 en una máquina virtual.

Módulo II

Matemática para la informática



Objetivo holístico del módulo

Desarrollamos las habilidades y conocimientos necesarios para aprender sobre teoría de conjuntos y lógica matemática, que son fundamentales en la programación y el diseño de algoritmos, para fortalecer habilidades y mantenerse actualizado con los avances y las aplicaciones matemáticas en el campo de la informática, reconociendo que la tecnología evoluciona rápidamente y siempre hay nuevos desarrollos en esta área.



Respondamos las siguientes preguntas

¿Conocemos los números binarios?

¿Sabríamos explicar cómo funciona un lector de CD o DVD?

¿Conocemos cómo se pueden intercambiar de colores en una computadora a otra sin que cambie la tonalidad?



Unidad temática N° 1. Nomenclatura informática

Definiciones y aplicaciones de la matemática en computación

Matemática en computación

La matemática para la computación es una rama más de la matemática que se enfoca en los conceptos y técnicas matemáticas que se utilizan en la informática y la computación. Esta disciplina incluye áreas como la teoría de la computación, la teoría de grafos, la lógica matemática, la teoría de la información, la estadística y la probabilidad, entre otras.

La matemática para la computación se utiliza para diseñar y analizar algoritmos, desarrollar sistemas informáticos eficientes,



modelar sistemas complejos y procesos de datos, y para comprender la complejidad de los problemas computacionales. También se utiliza para la criptografía, la seguridad informática y en la creación de sistemas de inteligencia artificial y aprendizaje automático. En resumen, la matemática es fundamental para la computación y es esencial para el desarrollo de la tecnología informática moderna.

Aplicación

La matemática ayuda a comprender el funcionamiento de las computadoras y las nomenclaturas (bits, bytes) que acompañan tales como: capacidad de almacenamiento, transferencia, manejo de datos, colores y otros. También permiten entender y desarrollar en diferentes áreas de la informática los cuales detallamos a continuación:

Diseño y análisis de algoritmos. La matemática es esencial para el diseño y análisis de algoritmos, ya que se utilizan técnicas matemáticas para medir la eficiencia de los algoritmos y para demostrar su corrección.

Criptografía y seguridad informática. La criptografía se basa en principios matemáticos, como la teoría de números y la teoría de grupos, para proteger la información en línea. La matemática también es esencial en la seguridad informática, donde se utilizan técnicas matemáticas para detectar y prevenir ataques informáticos.

Modelado y simulación. La matemática se utiliza para modelar sistemas complejos y procesos de datos, y para simular su comportamiento. Esto es especialmente importante en campos como la ingeniería y la física, donde se utilizan modelos matemáticos para simular el comportamiento de sistemas físicos.

Inteligencia artificial y aprendizaje automático. La matemática es esencial en el desarrollo de sistemas de inteligencia artificial y aprendizaje automático. Se utilizan técnicas matemáticas como el álgebra lineal, la probabilidad y la estadística para entrenar y ajustar modelos de aprendizaje automático.

Teoría de la computación. La teoría de la computación se basa en la matemática para entender y analizar la complejidad de los algoritmos y los problemas computacionales. La matemática es esencial para el desarrollo de la teoría de la computación y para la comprensión de la complejidad.

Sistemas de numeración

Los sistemas de numeración son sistemas que se utilizan para representar y manipular números. Existen varios sistemas de numeración, incluyendo:

Sistema de numeración decimal. El sistema decimal es el sistema de numeración más común en el mundo, utiliza diez dígitos (0-9) es nuestro sistema de numeración base.

1, 2, 3, 4, 5, 6, 7, 8, 9, 0

Sistema de numeración romano. El sistema romano utiliza letras para representar números y se utiliza en algunos contextos históricos y culturales, que seguramente lo llevaste en primaria.

I, II, III, IV, V, VI, VII, VIII, IX, X, XX, XXX, L, C, D, M

Sistema de numeración binario. El sistema binario utiliza sólo dos dígitos [0 y 1] y se utiliza en la electrónica digital y la informática es el lenguaje base de las computadoras.

0, 1, 10, 11, 100, 101, 110, 111

Sistema de numeración octal. El sistema octal utiliza 8 dígitos [0-7] y se utiliza en la programación y la electrónica.

0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17, 20

Sistema de numeración hexadecimal. El sistema hexadecimal utiliza 16 dígitos [0-9 y A-F] y se utiliza en la programación y en la representación de colores en las pantallas de los dispositivos electrónicos y la informática.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, 10

Sistemas de numeración binario

El sistema de numeración binario es un sistema numérico que utiliza solo dos dígitos, 0 y 1, para representar valores numéricos. Este sistema es fundamental en la informática y la electrónica digital, ya que los dispositivos electrónicos utilizan señales eléctricas que se pueden representar como bits, que son dígitos binarios.

Ej. $67 = 111101011_{(2)}$

Sistema decimal

El sistema de numeración decimal es el sistema numérico más comúnmente utilizado en el mundo. Este sistema utiliza diez dígitos numéricos, del 0 al 9, y cada posición en el número tiene un valor diez veces mayor que la posición a su derecha.

El sistema decimal se utiliza en la mayoría de las aplicaciones que implican números, desde cálculos financieros hasta mediciones y estadísticas. También es el sistema numérico utilizado por las personas en su vida cotidiana.

Ej. $87 = 87_{(10)}$

Sistema octal

El sistema de numeración octal es un sistema numérico que utiliza 8 dígitos [0-7] para representar valores numéricos. Cada dígito tiene un valor posicional, y el valor total de un número octal se calcula sumando los valores de cada dígito multiplicado por su posición.

En la programación, el sistema octal se utiliza para representar permisos de archivos y directorios en los sistemas operativos basados en Unix y Linux.

Ej. $60 = 74_{(8)}$

Sistema hexadecimal

El sistema de numeración hexadecimal es un sistema numérico que utiliza 16 dígitos, incluyendo los números del 0 al 9 y las letras de la A a la F para representar valores numéricos. Cada dígito tiene un valor posicional, y el valor total de un número hexadecimal se calcula sumando los valores de cada dígito multiplicado por su posición.

El sistema hexadecimal se utiliza para representar direcciones de memoria, valores de color en gráficos digitales, y en otras aplicaciones donde la representación compacta y la facilidad de manipulación y conversión a otros sistemas son importantes.

Ej. El color rojo en la informática es FF0000

Conversión de un sistema a otro

Decimal a binario

Para convertir un número decimal a binario, se puede seguir el siguiente proceso:

- Dividir el número decimal por 2.
- Anotar el resto de la división (que será 0 o 1) como el primer dígito del número binario.
- Dividir el resultado de la división anterior por 2, y anotar el resto como el segundo dígito del número binario.
- Continuar este proceso de división y anotación de restos hasta que el resultado sea 0.

$$\begin{array}{r}
 74 \quad | \quad 2 \\
 14 \quad | \quad 37 \quad | \quad 2 \\
 (0) \quad | \quad 17 \quad | \quad 18 \quad | \quad 2 \\
 \quad \quad | \quad (1) \quad | \quad (0) \quad | \quad 9 \quad | \quad 2 \\
 \quad \quad \quad \quad | \quad \quad \quad \quad | \quad \quad \quad \quad | \quad (1) \quad | \quad 4 \quad | \quad 2 \\
 \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad | \quad (0) \quad | \quad 2 \quad | \quad 2 \\
 \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad | \quad (0) \quad | \quad (1)
 \end{array}$$

$$74_{(10)} = 1001010_{(2)}$$

$$\begin{array}{r}
 93 \quad | \quad 2 \\
 13 \quad | \quad 46 \quad | \quad 2 \\
 (1) \quad | \quad (0) \quad | \quad 23 \quad | \quad 2 \\
 \quad \quad | \quad \quad \quad \quad | \quad (1) \quad | \quad 11 \quad | \quad 2 \\
 \quad \quad \quad \quad | \quad \quad \quad \quad | \quad \quad \quad \quad | \quad (1) \quad | \quad 5 \quad | \quad 2 \\
 \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad | \quad (1) \quad | \quad 2 \quad | \quad 2 \\
 \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad | \quad (0) \quad | \quad (1)
 \end{array}$$

$$93_{(10)} = 1011101_{(2)}$$

Decimal a octal

Para convertir un número decimal a octal, se puede seguir el siguiente proceso:

- Dividir el número decimal por 8.
- Anotar el resto de la división como el último dígito del número octal.
- Dividir el resultado de la división anterior por 8, y anotar el resto como el siguiente dígito del número octal.
- Continuar este proceso de división y anotación de restos hasta que el resultado sea 0.

$$\begin{array}{r}
 74 \quad | \quad 8 \\
 (2) \quad | \quad 9 \quad | \quad 8 \\
 \quad \quad | \quad (1) \quad | \quad (1)
 \end{array}$$

$$74_{(10)} = 112_{(8)}$$

$$\begin{array}{r}
 93 \quad | \quad 8 \\
 13 \quad | \quad 11 \quad | \quad 8 \\
 (5) \quad | \quad (3) \quad | \quad (1)
 \end{array}$$

$$93_{(10)} = 135_{(8)}$$

Decimal a hexadecimal

Para convertir un número decimal a hexadecimal, podemos seguir estos pasos:

- Dividamos el número decimal entre 16 y obtengamos el cociente y el residuo.
- El residuo representa el dígito hexadecimal correspondiente.
- Si el residuo es mayor a 9, reemplacémosle por la letra correspondiente en hexadecimal [A, B, C, D, E, F].

- Repetamos los pasos 1-3 con el cociente obtenido en el paso anterior hasta que el cociente sea 0.
- Los dígitos hexadecimales obtenidos en los residuos forman el número hexadecimal.

$$\begin{array}{r} 74 \\ \underline{16} \\ 4 \end{array}$$

10
A

$$74_{(10)} = 4A_{(16)}$$

$$\begin{array}{r} 93 \\ \underline{16} \\ 5 \end{array}$$

13
D

$$93_{(10)} = 5D_{(16)}$$

Binario a decimal

Para convertir un número binario a decimal, sigamos estos pasos:

- Escribamos el número binario.
- Asignemos una posición a cada dígito binario, comenzando desde la derecha y asignando valores consecutivos a cada posición ($2^0, 2^1, 2^2, \dots$).
- Multipliquemos cada dígito binario por el valor correspondiente a su posición y sumemos los resultados.
- El resultado obtenido es el número decimal equivalente al número binario.

1	1	1	0	1	0	0	0	0
2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
254	128	64	32	16	8	4	2	1
1x254	1x128	1x64	0x32	1x16	0x8	0x4	0x2	0x1
254	128	64	0	16	0	0	0	0
254+128+64+16=464								

Entonces $11101000_2 = 464_{10}$

Hexadecimal a decimal

- Para convertir un número hexadecimal a decimal, sigamos estos pasos:
- Identifiquemos cada dígito hexadecimal.

- Asignemos un valor decimal a cada dígito hexadecimal, utilizando la siguiente correspondencia:
 - a. Dígitos del 0 al 9: su valor decimal es el mismo que su valor hexadecimal.
 - b. Dígitos A-F: A=10, B=11, C=12, D=13, E=14, F=15.
- Multipliquemos cada dígito hexadecimal por la potencia correspondiente de 16, comenzando desde la derecha y asignando valores consecutivos a cada posición (16^0 , 16^1 , 16^2 , ...).
- Sumemos los resultados de todas las multiplicaciones.
- El resultado obtenido es el número decimal equivalente al número hexadecimal.

Ejemplo: Conviertamos 1D0 hexadecimal en un número decimal:

1	D	0
1	13	0
16^2	16^1	16^0
1×256	16	1
$1 \times 256 = 256$	$13 \times 16 = 208$	$0 \times 1 = 0$
$254 + 128 + 64 + 16 = 464$		



Valoremos lo aprendido

Respondamos las siguientes incógnitas en nuestro cuaderno:

- ¿Cómo funciona nuestra PC?
- Desarrollemos las aplicaciones que tiene el sistema binario en la vida cotidiana (mínimo 5).



Unidad temática N° 2. Lógica matemática



Analicemos y marquemos si es verdadero o falso

Proposiciones	V	F
Los sistemas operativos como Windows, macOS y Linux son necesarios para el funcionamiento de una computadora.		
Todos los programas de software están completamente libres de errores.		
La programación es un lenguaje utilizado para crear aplicaciones y software.		
Todos los sitios web en Internet están almacenados en un solo servidor.		
Los virus informáticos pueden infectar y dañar los archivos y sistemas de una computadora.		
La tecnología de inteligencia artificial todavía no tiene aplicaciones prácticas en el mundo real.		
El lenguaje de programación Python es ampliamente utilizado en el campo de la ciencia de datos.		
Todas las contraseñas seguras deben contener al menos ocho caracteres.		
Los algoritmos de compresión(winRAR) pueden reducir el tamaño de los archivos sin perder información.		
La programación es una habilidad que solo puede ser dominada por expertos en matemáticas.		



Lógica matemática

La lógica matemática es una rama de la matemática que se ocupa del estudio de los principios y métodos utilizados en el razonamiento y la argumentación válida. Se basa en reglas y estructuras formales para analizar y evaluar la validez de los argumentos y proposiciones.

En la lógica matemática, se utilizan símbolos y operadores para representar conceptos lógicos y se establecen reglas precisas para manipular y combinar estos símbolos.

Proposiciones

Son declaraciones que pueden ser verdaderas o falsas. Se representan mediante letras o símbolos y pueden ser combinadas mediante operadores lógicos.

Aquí hay algunos ejemplos de proposiciones:

- El sol es una estrella.
- $2 + 2 = 4$.
- Todos los seres humanos necesitan respirar.
- Los gatos son animales domésticos.
- Si llueve, entonces me llevaré un paraguas.
- Hoy es martes.
- 10 es mayor que 5.
- Los triángulos tienen tres lados.
- El agua hierve a 100 grados Celsius.
- 7 es un número primo.

Estas son solo algunas proposiciones simples. También se pueden combinar proposiciones para formar proposiciones más complejas, utilizando operadores lógicos. Por ejemplo:

- Si hace sol y hace calor, entonces iré a la playa.
- Si estudio, entonces aprobaré el examen.
- Todos los perros ladran o mueven la cola.
- No es cierto que hoy sea lunes.

Operadores lógicos. Son símbolos que se utilizan para combinar proposiciones y formar nuevas proposiciones. Los operadores lógicos más comunes son:

Negación (\sim): representa la negación de una proposición.

Conjunción (\wedge): representa la operación lógica "y" entre dos proposiciones.

Disyunción (\vee): representa la operación lógica "o" entre dos proposiciones.

Implicación (\rightarrow): representa la relación lógica de implicación entre dos proposiciones.

Bicondicional (\leftrightarrow): representa la relación lógica de doble implicación entre dos proposiciones.

Tabla de verdad

En lógica matemática, una tabla de verdad es una representación sistemática de los valores de verdad de una o más proposiciones en todas las posibles combinaciones de sus valores de verdad. Estas tablas son una herramienta fundamental para analizar y evaluar la validez de los argumentos lógicos, así como para determinar si una proposición o una combinación de proposiciones es una tautología [siempre verdadera], una contradicción [siempre falsa] o una contingencia [verdadera en algunos casos y falsa en otros].

En una tabla de verdad, se enumeran todas las combinaciones posibles de los valores de verdad de las proposiciones involucradas. Cada columna representa una proposición diferente, y cada fila representa una combinación específica de valores de verdad para las proposiciones. A continuación, se calcula el valor de verdad resultante de la proposición o combinación de proposiciones en cada fila.

Para construir una tabla de verdad, primero necesitamos determinar la cantidad de proposiciones involucradas.

$$2^{(\text{número de proposiciones})} = \text{números de filas}$$

Ejemplo:

$2^1=2$	
p	
V	
F	

$2^2=4$	
p	q
V	V
V	F
F	V
F	F

$2^3=4$		
p	q	r
V	V	V
V	V	F
V	F	V
V	F	F
F	V	V
F	V	F
F	F	V
F	F	F

Tabla de negación (\sim): representa la negación de una proposición

p	$\sim q$
V	F
F	V

Tabla de conjunción (\wedge): representa la operación lógica "y" entre dos proposiciones

p	q	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

Tabla de disyunción (\vee): representa la operación lógica "o" entre dos proposiciones

p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

Tabla de implicación (\rightarrow): representa la relación lógica de implicación entre dos proposiciones

p	q	$p \rightarrow q$
V	V	V
V	F	F
F	V	V
F	F	V

Tabla de bicondicional (\leftrightarrow): representa la relación lógica de doble implicación entre dos proposiciones

p	q	$p \leftrightarrow q$
V	V	V
V	F	F
F	V	F
F	F	V



Ejercitemos

Ejercicio 1

Construyamos la tabla de verdad para la proposición $(p \vee q) \wedge (\sim p)$.

p	q	$\sim q$	$p \leftrightarrow q$	$\sim p \wedge q$	$(p \leftrightarrow q) \vee (\sim p \wedge q)$
V	V	F	V	V	V
V	F	F	F	F	F
F	V	V	F	V	V
F	F	V	V	F	V

Ejercicio 2

Consideremos la proposición $(p \leftrightarrow q) \vee (\sim p \wedge q)$. Creemos la tabla de verdad correspondiente.

p	q	$\sim p$	$p \leftrightarrow q$	$\sim p \wedge q$	$(p \leftrightarrow q) \vee (\sim p \wedge q)$
V	V	F	V	V	V
V	F	F	F	F	F
F	V	V	F	V	V
F	F	V	V	F	V

Ejercicio 3

Construyamos la tabla de verdad para la proposición $\sim(p \vee q) \leftrightarrow (\sim p \wedge \sim q)$.

p	q	$p \vee q$	$\sim(p \vee q)$	$\sim p \wedge \sim q$	$\sim(p \vee q) \leftrightarrow (\sim p \wedge \sim q)$
V	V	V	F	F	V
V	F	V	F	F	F
F	V	V	F	F	V
F	F	F	V	V	V

Ejercicio 4

Dada la proposición $(p \wedge q) \vee (\sim p \wedge r)$, completamos la tabla de verdad considerando las proposiciones p, q y r.

p	q	r	$p \wedge q$	$\sim p \wedge r$	$(p \wedge q) \vee (\sim p \wedge r)$
V	V	V	V	V	V
V	V	F	V	F	V
V	F	V	F	V	V
V	F	F	F	F	F
F	V	V	F	V	V
F	V	F	F	F	F



Valoremos lo aprendido

Respondamos las siguientes incógnitas en nuestro cuaderno:

- ¿Qué entendemos por lógica matemática?
- Valoremos, analicemos y redactemos cinco falacias que suceden en nuestra sociedad, viendo las noticias.



Resolvamos las siguientes tablas de verdad

- Construyamos la tabla de verdad para la proposición $p \wedge q$.
- Dada la proposición $p \vee (q \wedge r)$, completemos la tabla de verdad para las proposiciones p , q y r , así como para la proposición completa.
- Consideremos la proposición $\sim(p \vee q) \wedge (\sim p \vee r)$. Crea la tabla de verdad correspondiente.
- Construyamos la tabla de verdad para la proposición $p \rightarrow (q \vee \sim r)$.
- Dada la proposición $p \wedge (\sim q \vee r)$, completemos la tabla de verdad considerando las proposiciones p , q y r .
- Construyamos la tabla de verdad para la proposición $\sim(p \leftrightarrow q)$.
- Dada la proposición $(p \wedge q) \rightarrow r$, completemos la tabla de verdad para las proposiciones p , q y r , así como para la proposición completa.
- Consideremos la proposición $(p \vee \sim q) \wedge (\sim p \vee r)$. Creemos la tabla de verdad correspondiente.
- Construyamos la tabla de verdad para la proposición $p \leftrightarrow (q \vee r)$.



Unidad temática N° 3. Progresiones, sucesiones y series numéricas



Demos una lectura y compartamos en clases las series más famosas y más usadas

Serie de Fibonacci: La serie de Fibonacci tiene aplicaciones en diversas áreas, como la teoría de números, la biología, la música y el arte. Se encuentra en la naturaleza en patrones de crecimiento de plantas, pétalos de flores y estructuras de conchas marinas. También se utiliza en algoritmos de búsqueda y optimización, y en la generación de secuencias numéricas.

Serie de armónicas. Se utilizan en matemáticas y física para analizar ondas y oscilaciones, así como en la teoría de números. También tienen aplicaciones en procesamiento de señales y análisis de espectro.

Serie de potencias. Las series de potencias son fundamentales en el cálculo y se utilizan para aproximar funciones complicadas mediante una suma infinita de términos. Son utilizadas en la expansión de funciones en series de Taylor, que permiten aproximar funciones en puntos cercanos a un valor dado.

Serie de Taylor. La serie de Taylor se utiliza para aproximar funciones complejas mediante una serie de potencias. Se utiliza en cálculo numérico, análisis numérico y diversas ramas de la física y la ingeniería para aproximar funciones y resolver ecuaciones diferenciales.

Serie de Fourier. La serie de Fourier es ampliamente utilizada en matemáticas, física e ingeniería, ya que proporciona una forma de analizar y representar funciones periódicas de manera más sencilla y conveniente. Al descomponer una función periódica en una serie de senos y cosenos, es posible estudiar sus propiedades, como la amplitud, la frecuencia y la fase de cada componente.



Progresiones, sucesiones y series numéricas

Una sucesión (o progresión) es una secuencia de números en la que cada término se obtiene siguiendo un patrón o una regla predefinida. En una progresión, los términos sucesivos están relacionados entre sí a través de una diferencia o una razón común, mientras que una serie es la suma de los términos de una sucesión.

Sucesiones o progresiones

Una sucesión numérica se representa comúnmente como la_n , donde "n" es un índice o número natural que indica la posición del término en la secuencia y la_n es el término correspondiente. Por ejemplo, una sucesión simple podría ser {1, 2, 3, 4, 5}, donde cada término se obtiene sumando 1 al término anterior.

Las sucesiones pueden ser definidas de diferentes maneras, ya sea de forma explícita, donde se proporciona una fórmula para calcular cada término, o de forma recursiva, donde cada término se calcula a partir de uno o más términos anteriores.

Ejemplos

- Sucesión de los primeros n números naturales

{1, 2, 3, 4, 5, ...}

Fórmula: $a_n = n$, donde "n" es el índice o número natural.

● Sucesión de los primeros n números pares

{2, 4, 6, 8, 10, ...}

Fórmula: $a_n = 2n$, donde "n" es el índice o número natural.

● Sucesión de los primeros n términos de la serie geométrica

{1, 2, 4, 8, 16, ...}

Fórmula: $a_n = 2^{(n-1)}$, donde "n" es el índice o número natural.

● Sucesión de los términos de la serie de Fibonacci

{0, 1, 1, 2, 3, 5, 8, 13, ...}

Fórmula: $a_n = a_{(n-1)} + a_{(n-2)}$, donde "n" es el índice o número natural, y $a_0 = 0$ y $a_1 = 1$ son los primeros dos términos.

● Sucesión de los primeros n términos de una progresión aritmética

{a, a + d, a + 2d, a + 3d, ...}

Fórmula: $a_n = a + (n-1)d$, donde "n" es el índice o número natural, "a" es el primer término y "d" es la diferencia común.

Clasificación

Las progresiones se pueden clasificar de diversas formas según sus características y propiedades. Aquí hay algunas clasificaciones comunes de las progresiones:

Clasificación por diferencia o razón

● **Progresiones aritméticas (PA).** En una progresión aritmética, la diferencia entre dos términos consecutivos es constante. Por ejemplo: {2, 5, 8, 11, 14}.

● **Progresiones geométricas (PG).** En una progresión geométrica, la razón entre dos términos consecutivos es constante. Por ejemplo: {2, 6, 18, 54}.

Clasificación por crecimiento o decrecimiento

- **Sucesiones crecientes.** Las sucesiones crecientes son aquellas en las que cada término es mayor que el término anterior. Por ejemplo, {1, 2, 3, 4, 5, ...} es una sucesión creciente de números naturales. En esta sucesión, cada término es mayor que el término anterior.
- **Sucesiones decrecientes.** Las sucesiones decrecientes son aquellas en las que cada término es menor que el término anterior. Por ejemplo, {5, 4, 3, 2, 1, ...} es una sucesión decreciente de números naturales. En esta sucesión, cada término es menor que el término anterior.

Clasificación por finitud o infinitud

- **Sucesiones finitas.** Las sucesiones finitas son aquellas que tienen un número específico de términos. Por ejemplo, {1, 2, 3, 4, 5} es una sucesión finita con cinco términos. Estas sucesiones tienen un final definido y no continúan indefinidamente.
- **Sucesiones infinitas.** Las sucesiones infinitas son aquellas que continúan indefinidamente, sin tener un límite en el número de términos. Por ejemplo, {1, 2, 3, 4, ...} es una sucesión infinita de números naturales que no tiene un final definido.

Progresión aritmética y geométrica

Progresión aritmética (PA)

Una progresión aritmética (PA) es una secuencia de números en la que la diferencia entre dos términos consecutivos es constante. Cada término de una progresión aritmética se obtiene sumando la diferencia común (denotada como "d") al término anterior.

La fórmula general para la n-ésimo término (a_n) de una progresión aritmética es:

$$a_n = a_1 + (n - 1)d$$

Donde:

" a_n " Es el n-ésimo término de la progresión.

" a_1 " Es el primer término de la progresión.

"d" Es la diferencia común entre términos consecutivos.

"n" Es el índice o posición del término en la progresión.

Ejemplos

{2, 5, 8, 11, 14, ...}

En esta progresión aritmética, el primer término a_1 es 2 y la diferencia común $[d]$ es 3.

{10, 7, 4, 1, -2, ...}

En esta progresión aritmética, el primer término a_1 es 10 y la diferencia común $[d]$ es -3.

{-4, -1, 2, 5, 8, ...}

En esta progresión aritmética, el primer término a_1 es -4 y la diferencia común $[d]$ es 3.

Suma de una progresión aritmética

La suma de una progresión aritmética finita se puede calcular utilizando la fórmula de la suma de los términos de una progresión aritmética. La fórmula general para la suma de los primeros n términos de una progresión aritmética es:

$$s_n = ((a_1 + a_n)n) / 2$$

Donde:

" s_n " es la suma de los primeros n -términos de la progresión.

" n " es el número de términos que deseas sumar.

" a_1 " es el primer término de la progresión.

" a_n " es el n -ésimo término de la progresión.

Para calcular la suma de una progresión aritmética, sigue estos pasos:

Determina el valor del primer término " a_1 ", el número de términos n y la diferencia común d entre los términos consecutivos de la progresión aritmética.

Calcula el valor del n -ésimo término a_n utilizando la fórmula general de la progresión aritmética:

$$a_n = a_1 + (n - 1)d$$

Sustituye los valores de a_1 , a_n y n en la fórmula de la suma de los términos de la progresión aritmética:

$$s_n = ((a_1 + a_n)n) / 2$$

Realiza los cálculos necesarios para obtener el valor de la suma de los términos de la progresión aritmética.

Ejemplo

Consideremos la progresión aritmética {2, 5, 8, 11, 14}. Queremos calcular la suma de los primeros 4 términos de esta progresión.

$a_1=2$ (primer término)

$n=4$ (número de términos)

Para encontrar a_n , utilizamos la fórmula de la progresión aritmética:

$$a_n = a_1 + (n - 1)d$$

$$a_4 = 2 + (4 - 1)3$$

$$a_4 = 2 + 3*3$$

$$a_4 = 2+9$$

$$a_4 = 11$$

Sustituyendo los valores en la fórmula de la suma de los términos:

$$S_n = ((2+11)4)/2$$

$$S_n = ((13)4)/2$$

$$S_n = 26$$

Por lo tanto, la suma de los primeros 4 términos de la progresión aritmética {2, 5, 8, 11, 14} es igual a 26.

Progresión geométrica (PG)

Una progresión geométrica (PG) es una secuencia de números en la que cada término se obtiene multiplicando el término anterior por una razón constante. La razón común se denota como "r". Cada término de una progresión geométrica se obtiene aplicando la razón común al término anterior.

La fórmula general para el n-ésimo término [a_n] de una progresión geométrica es:

$$a_n = a_1 * r^{(n-1)}$$

Donde:

" a_n " es el n-ésimo término de la progresión.

" a_1 " es el primer término de la progresión.

" r " es la razón común entre términos consecutivos.

" n " es el índice o posición del término en la progresión.

Ejemplos

{2, 4, 8, 16, 32, ...}

En esta progresión geométrica, el primer término a_1 es 2 y la razón común r es 2.

{5, -10, 20, -40, 80, ...}

En esta progresión geométrica, el primer término a_1 es 5 y la razón común r es -2.

{1/2, 1/4, 1/8, 1/16, ...}

En esta progresión geométrica, el primer término a_1 es 1/2 y la razón común r es 1/2.

Suma de una progresión geométrica

La suma de una progresión geométrica finita se puede calcular utilizando la fórmula de la suma de los términos de una progresión geométrica. La fórmula general para la suma de los primeros n términos de una progresión geométrica es:

$$S_n = (a_1 * (1 - r^n)) / (1 - r)$$

Donde:

" S_n " es la suma de los primeros n -términos de la progresión.

" a_1 " es el primer término de la progresión.

" r " es la razón común entre términos consecutivos.

" n " es el número de términos que deseas sumar.

Para calcular la suma de una progresión geométrica, sigamos estos pasos:

Determinemos el valor del primer término a_1 , la razón común r y el número de términos n que deseas sumar en la progresión geométrica.

Utilicemos la fórmula de la suma de los términos de la progresión geométrica: $S_n = (a_1 * (1 - r^n)) / (1 - r)$

Sustituyamos los valores de a_1 , r y n en la fórmula de la suma de los términos de la progresión geométrica.

Realicemos los cálculos necesarios para obtener el valor de la suma de los términos de la progresión geométrica.

Ejemplo

Consideremos la progresión geométrica {2, 4, 8, 16, 32}. Queremos calcular la suma de los primeros 4 términos de esta progresión.

$a_1 = 2$ [primer término]

$r = 4/2 = 2$ [razón común]

$n = 4$ [número de términos]

Sustituyendo los valores en la fórmula de la suma de los términos:

$$S_4 = (2 * (1 - 2^4)) / (1 - 2)$$

$$S_4 = 2 * (1 - 16) / (1 - 2)$$

$$S_4 = 2 * (-15) / (-1)$$

$$S_4 = 30$$

Por lo tanto, la suma de los primeros 4 términos de la progresión geométrica {2, 4, 8, 16, 32} es igual a 30.



Valoremos lo aprendido

Respondamos las siguientes incógnitas en nuestro cuaderno:

- ¿Cuál es el valor de la proporción Áurea?
- Mencionemos otros nombres de la proporción áurea.



Resolvamos las siguientes progresiones aritméticas

- Calculemos la suma de los primeros 8 términos de la progresión aritmética $\{3, 7, 11, 15, \dots\}$.
- Encontramos la suma de los primeros 10 términos de la progresión aritmética $\{2, 6, 10, 14, \dots\}$.
- Determinemos la suma de los primeros 12 términos de la progresión aritmética $\{-4, -1, 2, 5, \dots\}$.
- Calculemos la suma de los primeros 15 términos de la progresión aritmética $\{1, 4, 7, 10, \dots\}$.
- Encontramos la suma de los primeros 20 términos de la progresión aritmética $\{12, 9, 6, 3, \dots\}$.
- Determinemos la suma de los primeros 6 términos de la progresión aritmética $\{5, 8, 11, 14, \dots\}$.
- Calculemos la suma de los primeros 7 términos de la progresión aritmética $\{1, -3, -7, -11, \dots\}$.
- Encontramos la suma de los primeros 9 términos de la progresión aritmética $\{2, -1, -4, -7, \dots\}$.
- Determinemos la suma de los primeros 10 términos de la progresión aritmética $\{10, 7, 4, 1, \dots\}$.
- Calculemos la suma de los primeros 5 términos de la progresión aritmética $\{6, 3, 0, -3, \dots\}$.



Unidad temática N° 4. Álgebra booleana



Demos una lectura y compartamos en clases las series más famosas y más usadas

- ¿Si les digo, ve a comprar galleta y refresco, qué debemos traer?
- ¿Si les digo, ve a comprar galleta o refresco, que debemos traer?
- ¿Si les digo, ve a comprar galleta o refresco pero no galleta oreo, que debemos traer?

Álgebra booleana

El álgebra booleana es un sistema matemático que se basa en un conjunto de reglas y operaciones lógicas que se aplican a variables booleanas, que solo pueden tomar dos valores: verdadero [1] o falso [0]. Fue desarrollada por el matemático y lógico George Boole en el siglo XIX y es ampliamente utilizada en el diseño y análisis de circuitos digitales, programación, sistemas de control y lógica computacional

Leyes lógicas

Idempotencia	$p \vee p = p$	$p \wedge p = p$
Asociativas	$(p \vee q) \vee r = p \vee (q \vee r)$	$(p \wedge q) \wedge r = p \wedge (q \wedge r)$
Conmutativas	$p \vee q = q \vee p$	$p \wedge q = q \wedge p$
Distributivas	$p \vee (q \wedge r) = (p \vee q) \wedge (p \vee r)$	$p \wedge (q \vee r) = (p \wedge q) \vee (p \wedge r)$
Identidad	$p \vee F = p$	$p \wedge V = p$
Negación (Complemento)	$p \vee \sim p = V \quad \sim(\sim p) = p$	$p \wedge \sim p = F$ $\sim V = F \quad \sim F = V$
De Morgan	$\sim(p \vee q) = \sim p \wedge \sim q$	$\sim(p \wedge q) = \sim p \vee \sim q$
Condicional	$p \rightarrow q = \sim p \vee q$	$\sim(p \rightarrow q) = p \wedge \sim q$
Bicondicional	$p \leftrightarrow q = (p \rightarrow q) \wedge (q \rightarrow p)$	$p \leftrightarrow q = (p \wedge q) \vee (\sim p \wedge \sim q)$
Absorción	$p \wedge (p \vee q) = p \quad p \wedge (\sim p \vee q) = p \wedge q \quad p \vee V = V$ $p \vee (p \wedge q) \equiv p \quad p \vee (\sim p \wedge q) \equiv p \vee q \quad p \wedge F = F$	

Ejemplo 1

Simplifiquemos la expresión $(p \vee q) \wedge (p \vee \sim q) \wedge (p \vee r)$.

Solución

Utilizando la ley de distribución y las leyes de absorción, podemos simplificar la expresión de la siguiente manera:

$$= [(p \vee q) \wedge (p \vee \sim q)] \wedge (p \vee r) \text{ [ley de asociatividad]}$$

$$= p \wedge (q \vee \sim q) \wedge (p \vee r) \text{ [ley de absorción]}$$

$$= p \wedge V \wedge (p \vee r) \text{ [ley de negación]}$$

$$= p \wedge (p \vee r) \text{ [ley de identidad]}$$

$$= p \text{ [ley de absorción]}$$

Por lo tanto, la expresión se simplifica a p .

Ejemplo 2

Simplifiquemos la expresión $(p \wedge q) \vee (p \rightarrow q)$.

Solución

Utilizaremos las leyes lógicas para simplificar la expresión

$$(p \wedge q) \vee (p \rightarrow q)$$

Podemos simplificar esto utilizando la ley de implicación y distribución

$$= (p \wedge q) \vee (\sim p \vee q) \text{ [ley de implicación]}$$

$$= (p \wedge q) \vee (q \vee \sim p) \text{ [ley conmutativa de la disyunción]}$$

$$= (q \vee (p \wedge q)) \vee \sim p \text{ [ley asociativa de la disyunción]}$$

$$= [(p \wedge q) \vee q] \vee \sim p \text{ [ley idempotente]}$$

$$= [q \vee \sim p] \vee \sim p \text{ [ley de absorción]}$$

$$= q \vee (\sim p \vee \sim p) \text{ [ley asociativa de la disyunción]}$$

$$= q \vee \sim p \text{ [ley idempotente]}$$

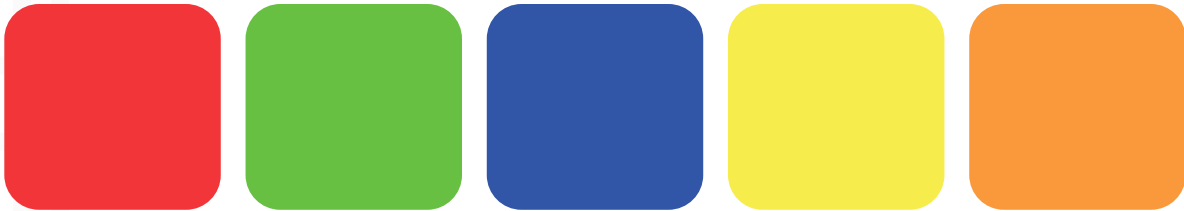
Por lo tanto, la expresión se simplifica a $q \vee \sim p$.



Analizamos el siguiente problema de lógica

"El enigma de los colores"

Tenemos una fila de cinco tarjetas, cada una de un color diferente: rojo, verde, azul, amarillo y naranja. Solo tenemos tres pistas para ordenar las tarjetas correctamente:



- El color azul no está en el extremo izquierdo.
- El color rojo está entre el verde y el amarillo.
- El color naranja está al lado del color verde.

¿Podríamos determinar el orden correcto de las tarjetas?



Resolvamos los siguientes ejercicios de conjuntos

- Simplifiquemos la expresión $\sim(p \wedge q) \vee (p \wedge r) \vee (q \wedge r)$.
- Simplifiquemos la expresión $(p \vee q) \wedge (p \wedge \sim q)$.
- Simplifiquemos la expresión $(p \wedge (q \vee r)) \vee (p \wedge (q \wedge r))$.
- Simplifiquemos la expresión $(p \wedge q) \vee (p \wedge r) \vee (q \wedge r)$.



Unidad temática N° 5. Teoría de conjuntos



Leamos y opinemos

El álgebra relacional es un lenguaje formal utilizado para manipular y consultar conjuntos de datos en bases de datos relacionales. El uso de conjuntos en álgebra relacional es fundamental para realizar operaciones como la unión, la intersección y la diferencia entre conjuntos.

Supongamos que tenemos dos tablas en una base de datos: "Empleados" y "Departamentos". La tabla "Empleados" contiene información sobre los empleados de una empresa, mientras que la tabla "Departamentos" contiene información sobre los diferentes departamentos de la empresa.

Tabla "Empleados"

ID	Nombre	Departamento
1	Juan	Ventas
2	María	Marketing
3	Pedro	Ventas
4	Ana	Recursos Humanos
5	Carlos	Ventas
6	Juana	Marketing

Tabla "Departamentos"

ID	Nombre
1	Ventas
2	Marketing
3	Finanzas
4	Recursos Humanos

- ¿Cuántos departamentos en la empresa hay?
- ¿Cuántas personas trabajan en ventas ?
- ¿Cuáles departamentos no tienen empleados?
- ¿Cuáles el departamento con más empleados?



Teoría de conjuntos

La teoría de conjuntos es una rama de las matemáticas que se centra en el estudio de los conjuntos, que son colecciones bien definidas de elementos. Fue desarrollada por el matemático Georg Cantor en el siglo XIX y ha sido fundamental en el desarrollo de diversas ramas de las matemáticas y la lógica.

Elementos y pertenencia. Un conjunto está compuesto por elementos individuales. Si un elemento pertenece a un conjunto, se denota utilizando el símbolo " \in ". Por ejemplo, si A es un conjunto y " a " es un elemento, se escribe " $a \in A$ " para indicar que " a " pertenece al conjunto A .

Conjunto vacío. Es un conjunto que no contiene ningún elemento y se representa por el símbolo " \emptyset " o " $\{\}$ ". Es importante destacar que el conjunto vacío es un conjunto válido y existe en la teoría de conjuntos.

Igualdad de conjuntos. Dos conjuntos son iguales si tienen exactamente los mismos elementos. Se denota por el símbolo " $=$ ".

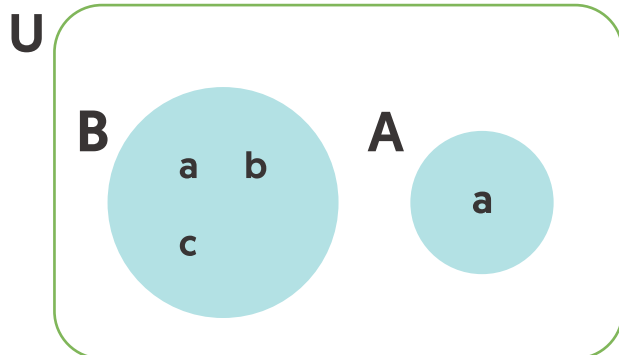
Conjunto universal. Es el conjunto que contiene a todos los elementos considerados en un contexto dado. Se denomina por " U ".

Determinación de conjuntos

La determinación de conjuntos se refiere al proceso de identificar y describir los elementos que forman parte de un conjunto específico. Esto implica determinar qué elementos cumplen con las propiedades o criterios establecidos para formar parte del conjunto en cuestión.

Existen diferentes formas de determinar conjuntos, dependiendo de cómo se especifiquen los elementos o las características que deben cumplir. Algunos métodos comunes incluyen:

Diagramas de Venn. Los diagramas de Venn son representaciones gráficas que se utilizan para determinar conjuntos y visualizar las relaciones entre ellos. En un diagrama de Venn, los conjuntos se representan como regiones cerradas y se superponen para mostrar las intersecciones y las diferencias entre los conjuntos.



Conjuntos por extensión: En este método, los elementos del conjunto se enumeran directamente dentro de llaves $\{ \}$. Por ejemplo, si queremos determinar el conjunto de los números pares menores que 10, podemos escribirlo como $A = \{2, 4, 6, 8\}$.

Conjuntos por comprensión: En este método, se utilizan expresiones lógicas y símbolos matemáticos para describir los elementos del conjunto. Por ejemplo, si queremos determinar el conjunto de los números enteros positivos, podemos escribirlo como $C = \{x \mid x \in \mathbb{Z}, x > 0\}$, donde \mathbb{Z} representa el conjunto de los números enteros.

Subconjuntos o inclusión de conjuntos

Un conjunto A es un subconjunto de otro conjunto B si todos los elementos de A también pertenecen a B. Se denota por " \subset ". Si A es un subconjunto propio de B, se denota por " \subsetneq ". Por ejemplo, si $A = \{1, 2\}$ y $B = \{1, 2, 3\}$, entonces $A \subset B$.

Propiedades de inclusión

Propiedad reflexiva

Todo conjunto es subconjunto de sí mismo. Es decir, para cualquier conjunto A, se cumple que $A \subset A$.

Propiedad de transitividad

Si un conjunto A es subconjunto de un conjunto B, y a su vez B es subconjunto de un conjunto C, entonces A es subconjunto de C. En otras palabras, si $A \subset B$ y $B \subset C$, entonces $A \subset C$.

Propiedad antisimétrica

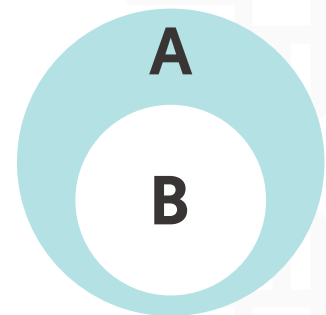
La propiedad antisimétrica es una propiedad que puede aplicarse a las relaciones de inclusión entre conjuntos donde $A \subset B$ entonces $B \not\subset A$.

Propiedad de subconjunto vacío

El conjunto vacío $\{\emptyset\}$ es subconjunto de cualquier conjunto. Es decir, para cualquier conjunto A, se cumple que $\emptyset \subset A$.

Propiedad de la igualdad de conjuntos

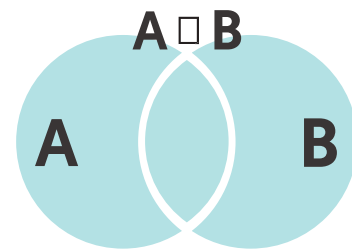
Dos conjuntos son iguales si y sólo si son mutuamente subconjuntos. Es decir, si $A \subset B$ y $B \subset A$, entonces $A = B$.



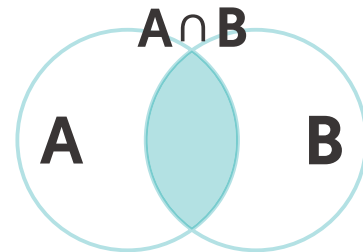
Operaciones con conjuntos

Las operaciones con conjuntos son acciones que se pueden realizar para combinar, comparar o manipular conjuntos. Las operaciones más comunes son la unión, la intersección, la diferencia y el complemento. A continuación, te explico cada una de estas operaciones:

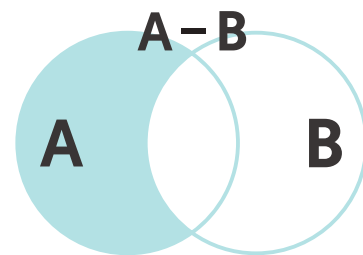
Unión. La unión de dos conjuntos A y B, denotada por $A \cup B$, es un nuevo conjunto que contiene todos los elementos que pertenecen a A o a B (o a ambos conjuntos). Es decir, la unión de A y B es el conjunto que contiene todos los elementos de A y todos los elementos de B sin repetir ninguno.



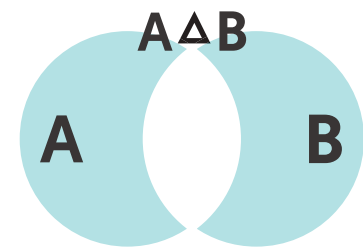
Intersección. La intersección de dos conjuntos A y B, denotada por $A \cap B$, es un nuevo conjunto que contiene todos los elementos que pertenecen tanto a A como a B. Es decir, la intersección de A y B es el conjunto que contiene sólo los elementos que son comunes a ambos conjuntos.



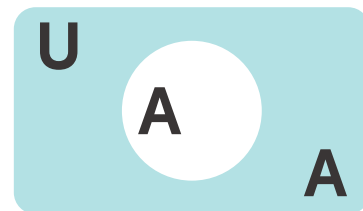
Diferencia. La diferencia de dos conjuntos A y B, denotada por $A - B$ o $A \setminus B$, es un nuevo conjunto que contiene todos los elementos que pertenecen a A pero no a B. En otras palabras, la diferencia de A y B es el conjunto que contiene los elementos que están en A pero no en B.



Diferencia simétrica. La diferencia simétrica nos permite encontrar los elementos que están en uno u otro conjunto, pero no en ambos. Se denota por el símbolo Δ [delta]. La diferencia simétrica entre dos conjuntos A y B se define de la siguiente manera: $A \Delta B = (A - B) \cup (B - A)$



Complemento. El complemento de un conjunto A con respecto a un conjunto universal U, denotado por A' , es un nuevo conjunto que contiene todos los elementos que pertenecen a U pero no a A. Es decir, el complemento de A es el conjunto que contiene todos los elementos que no están en A.



Valoremos lo aprendido

Respondamos las siguientes incógnitas en nuestro cuaderno:

- ¿Cómo funciona ChatGPT? La revolución de la Inteligencia Artificial

- Realicemos un glosario de términos informáticos que son nuevos o desconocidos (mínimo 10 palabras), de seguro iremos aprendiendo en los siguientes módulos.



Resolvamos los siguientes ejercicios de conjuntos

- Dados los conjuntos $A = \{1, 2, 3\}$ y $B = \{2, 3, 4\}$, encontremos $A \cup B$ (unión).
- Dados los conjuntos $A = \{1, 2, 3\}$ y $B = \{2, 3, 4\}$, encontremos $A \cap B$ (intersección).
- Dados los conjuntos $A = \{1, 2, 3\}$ y $B = \{2, 3, 4\}$, encontremos $A - B$ (diferencia).
- Dados los conjuntos $A = \{1, 2, 3\}$ y $B = \{2, 3, 4\}$, encontremos $B - A$ (diferencia).
- Dados los conjuntos $A = \{1, 2, 3\}$ y $B = \{2, 3, 4\}$, encontremos $A \Delta B$ (diferencia simétrica).
- Dados los conjuntos $A = \{1, 2, 3\}$ y $B = \{2, 3, 4\}$, encontremos el complemento de A con respecto a U (donde U es el conjunto universal).
- Dados los conjuntos $A = \{\text{rojo, azul, verde}\}$ y $B = \{\text{verde, amarillo, naranja}\}$, encontremos $A \cap B$ (intersección).
- Dados los conjuntos $A = \{\text{manzana, plátano, pera}\}$ y $B = \{\text{manzana, uva, piña}\}$, encontremos $A \cup B$ (unión).
- Dados los conjuntos $A = \{1, 2, 3, 4\}$ y $B = \{3, 4, 5, 6\}$, encontremos $A \cap B$ (intersección).
- Dados los conjuntos $A = \{a, b, c\}$ y $B = \{b, c, d\}$, encontremos $A - B$ (diferencia).
- Dados los conjuntos $A = \{1, 2, 3\}$ y $B = \{2, 3, 4\}$, encontremos $B - A$ (diferencia).
- Dados los conjuntos $A = \{\text{rojo, azul, verde}\}$ y $B = \{\text{verde, amarillo, naranja}\}$, encontremos el complemento de A con respecto a U .

- Dados los conjuntos $A = \{\text{perro, gato, pájaro}\}$ y $B = \{\text{pájaro, pez, serpiente}\}$, encontremos $A \cup B$.
- Dados los conjuntos $A = \{1, 2, 3, 4\}$ y $B = \{3, 4, 5, 6\}$, encontremos $B \cup A$.
- Dados los conjuntos $A = \{a, b, c\}$ y $B = \{b, c, d\}$, encontremos $A \cap B$.
- Dados los conjuntos $A = \{\text{manzana, plátano, pera}\}$ y $B = \{\text{manzana, uva, piña}\}$, encontremos $B - A$.
- Dados los conjuntos $A = \{\text{rojo, azul, verde}\}$ y $B = \{\text{verde, amarillo, naranja}\}$, encontremos $A \Delta B$.
- Dados los conjuntos $A = \{1, 2, 3, 4\}$ y $B = \{3, 4, 5, 6\}$, encontremos $B \Delta A$.
- Dados los conjuntos $A = \{\text{perro, gato, pájaro}\}$ y $B = \{\text{pájaro, pez, serpiente}\}$, encontremos $B - A$.
- Dados los conjuntos $A = \{\text{manzana, plátano, pera}\}$ y $B = \{\text{manzana, uva, piña}\}$, encontremos $A \Delta B$.
- Dados los conjuntos $A = \{1, 2, 3\}$, $B = \{2, 3, 4\}$ y $C = \{3, 4, 5\}$, encontremos $A \cap B \cap C$ (intersección).
- Dados los conjuntos $A = \{\text{rojo, azul}\}$, $B = \{\text{azul, verde}\}$ y $C = \{\text{rojo, verde}\}$, encontremos $A \cup B \cup C$ (unión).
- Dados los conjuntos $A = \{1, 2, 3\}$, $B = \{2, 3, 4\}$ y $C = \{3, 4, 5\}$, encontremos $A - B \cup C$ (diferencia y unión).
- Dados los conjuntos $A = \{a, b, c\}$, $B = \{b, c, d\}$ y $C = \{c, d, e\}$, encontremos $A \Delta B \Delta C$ (diferencia simétrica).
- Dados los conjuntos $A = \{\text{manzana, pera}\}$, $B = \{\text{pera, uva}\}$ y $C = \{\text{manzana, uva}\}$, encontremos el complemento de A con respecto a $B \cap C$.

Problemas de conjuntos

En una encuesta a un grupo de estudiantes, se les preguntó si preferían matemáticas (M), ciencias (C) o literatura (L). Los resultados mostraron que 25 estudiantes prefieren matemáticas, 18 prefieren ciencias, 12 prefieren literatura, 10 prefieren matemáticas y ciencias, 8 prefieren ciencias y literatura, y 6 prefieren matemáticas y literatura, 3 prefieren las tres materias. De un total de 40 estudiantes, ¿cuántos estudiantes no prefieren ninguna materia?

Módulo III

Programación I



Objetivo holístico del módulo

Fortalecemos habilidades y destrezas de programación, de la misma manera promovemos temáticas de prevención de toda forma de violencia a través de la sensibilización a la población, conocemos los distintos lenguajes de programación y aplicabilidad.



Unidad temática N° 1. Diseño de algoritmos



Respondamos en nuestros cuadernos las siguientes preguntas

- ¿Qué es un algoritmo y para qué se utiliza en programación?
- Mencionemos tres características importantes de los algoritmos.
- Expliquemos la importancia de la precisión en los algoritmos.



Diseño de algoritmos

El diseño de algoritmos se trata de crear una serie organizada de pasos que, cuando se siguen en orden, permiten resolver un problema específico o lograr un objetivo deseado. Los algoritmos son como recetas detalladas que cualquier persona puede seguir para obtener un resultado predecible.

Características de los algoritmos

Secuencia de pasos. Un algoritmo consiste en una serie de pasos u operaciones que se ejecutan de manera secuencial para resolver un problema.

Entrada de datos. Un algoritmo actúa sobre unos datos de entrada para generar unos resultados.

Precisión. Cada paso del algoritmo debe estar perfectamente definido y no dar lugar a ambigüedades.

Finitud. Los algoritmos deben terminar después de un número finito de pasos. No pueden ser infinitos.

Resultado. La ejecución de un algoritmo debe producir algún resultado observable y determinista.

Generalidad. Los algoritmos deben ser genéricos y no depender de valores concretos para poder aplicarse a distintos conjuntos de datos.

Independencia. El algoritmo no debe hacer suposiciones sobre los datos de entrada más allá de las especificadas.

Eficiencia. Un algoritmo eficiente produce el resultado deseado optimizando recursos como tiempo y espacio de memoria.



Representación de algoritmos

Existen diferentes formas de representar algoritmos en programación:

Diagramas de flujo. Utilizan símbolos estandarizados para representar el flujo del algoritmo. Son muy intuitivos y fáciles de entender.

Pseudocódigo. Describe los pasos del algoritmo en texto plano utilizando una sintaxis simple pero no un lenguaje formal de programación.

Diagrama N/S. Muestra la arquitectura y relaciones entre módulos mediante cajas y flechas. Útil para algoritmos complejos.




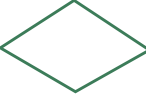





Símbolo	Función	Símbolo	Función
Terminal 	Indicar el inicio y fin del diagrama.	Teclado. 	Introducir datos manualmente por el teclado.
Entrada/salida 	Entrada o salida simple de información.	Decisión. 	Indica operaciones lógicas o de comparación y tienen dos salidas dependiendo del resultado.
Proceso 	Realizar cualquier operación o cálculo con la información.		
Salida a Impresora 	Salida de información a la impresora.	Conectores. 	Une dos partes del diagrama a la misma o diferente página.
Salida a pantalla 	Mostrar información de salida a la pantalla.	Flechas de flujo 	Indica la dirección del flujo de la información.

Diagrama de chapin. Representa el flujo del algoritmo mediante flechas que indican secuencias, bucles, bifurcaciones.

Descripción narrativa. Explica el algoritmo paso a paso en texto corrido y de manera coloquial.

Representación formal. Usa notación matemática rigurosa para especificar el algoritmo. Por ejemplo, notación "Gran O".

Lenguaje de programación. Implementa directamente el algoritmo en un lenguaje como C, Python, Java, etc.

Pseudolenguaje. Mezcla codificación de alto nivel con descripción narrativa de los pasos.

La representación elegida dependerá de la audiencia y la complejidad del algoritmo. Lo más común durante el diseño es usar diagramas de flujo o pseudocódigo antes de codificar.

Diagrama de flujo

El diagrama de flujo o flujograma es una representación gráfica de todos los pasos que consta un proceso. Este esquema visual muestra la sucesión cronológica de las operaciones que componen un sistema. En este sentido, todos los pasos y acciones del proceso se encuentran interrelacionadas para conducir a un resultado específico.

A grandes rasgos, un diagrama de flujo consta de figuras geométricas que identifican distintos tipos de pasos y acciones. Por ejemplo, una elipse alargada suele marcar el inicio o final del proceso. Otras figuras comunes son el rectángulo, que ilustra una acción; el rombo, que representa una decisión; y las flechas, que determinan la dirección del flujo.



El diagrama de flujo sirve, principalmente, para ilustrar algoritmos en informática o procesos industriales. Este tipo de diagrama está ligado a la tecnología, pues ayuda a mostrar qué función cumple cada máquina, o los pasos que se realizan en una rutina programada.

Otros ámbitos en los que se usan los diagramas de flujo son en la economía y psicología. También es útil para visualizar y hacer seguimiento de proyectos, de apoyo para la toma de decisiones o para identificar cuellos de botella en un proceso.

Tipos de diagrama de flujo

De acuerdo con su disposición en el plano y su función informativa, existen cuatro tipos de diagramas de flujo: vertical, horizontal, panorámico y arquitectónico.

Diagrama de flujo vertical

El diagrama de flujo vertical describe una secuencia de acciones en dirección descendente. Puede haber ramificaciones laterales, pero la mayoría del flujo debe presentar los pasos de forma que todo el proceso se despliegue hacia abajo.

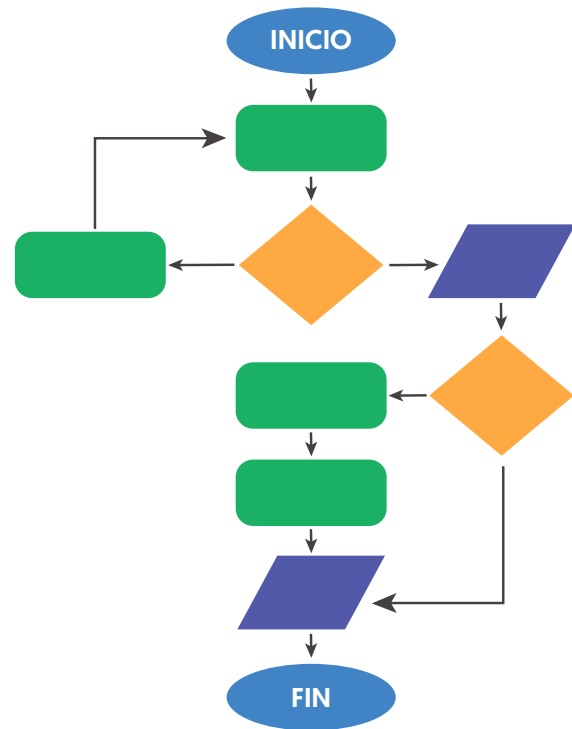


Diagrama de flujo horizontal

El diagrama de flujo horizontal muestra una secuencia de operaciones que comienza en el lado izquierdo del gráfico y termina en el lado derecho. Similar al diagrama de flujo vertical, el horizontal también puede tener ramificaciones en dirección ascendente o descendente.

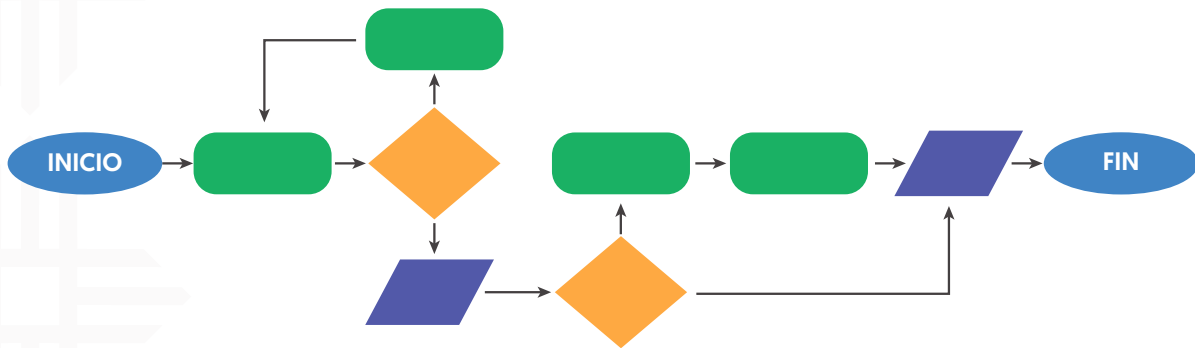


Diagrama de flujo panorámico

Es un tipo de diagrama de flujo que contiene, en un solo plano, todas las secuencias de acciones que consta en un proceso. Aquí hablamos de un nivel de complejidad mayor, combinando flujos en sentido vertical y horizontal. También se pueden representar acciones simultáneas en paralelo u otros procesos relacionados.

El diagrama de flujo panorámico es usado en todo proceso en el que haya un gran número de pasos, decisiones, subprocesos y entradas/salidas de datos.

- **Inicio o fin:** el óvalo o elipse se emplea para indicar el inicio y final del diagrama.
- **Paso, etapa o actividad:** el rectángulo posee información relativa al tipo de actividad que se realiza.
- **Decisión:** el rombo formula una pregunta o decisión a tomar. Usualmente, de esta figura surgen dos líneas de flujo, una cuando la respuesta es "Sí", y otra si la respuesta es "NO".
- **Entrada o salida de datos:** el rectángulo en cursiva se utiliza para mostrar cuándo se introducen o envían datos en el proceso.
- **Terminal:** el rectángulo redondeado tiene una función similar al de Inicio/Fin, pero sirve para ilustrar el inicio o final de un subproceso.
- **Flecha:** se utiliza para indicar el sentido de las acciones y la conexión entre los pasos.
- **Comentarios y anotaciones:** la línea [sin flecha] y rectángulo abierto sirven para anotar comentario relacionado con un paso del diagrama.
- **Proceso predefinido:** el doble rectángulo sirve para incluir un proceso o subproceso ya definido en otra parte, ya sea dentro o fuera del diagrama.
- **Preparación:** el hexágono alargado ilustra un paso de preparación o inicialización de otro paso posterior.
- **Operación manual:** es un símbolo que indica una operación que debe activarse manualmente.
- **Entrada manual:** es un símbolo que indica una entrada de datos que se introduce de forma manual.
- **Documento:** este símbolo se usa cuando se imprime o guarda un documento con información del proceso. En ocasiones, se utiliza tres de estos símbolos de forma intercalada para mostrar que se guardan varios documentos.
- **Archivo temporal:** el triángulo indica que se trata de un archivo que se guarda temporalmente.
- **Archivo permanente:** el triángulo inverso indica que se trata de un archivo que permanece durante el resto del proceso.
- **Retraso:** el símbolo que mezcla el rectángulo con una elipse ilustra que, antes de realizar el siguiente paso, hay un retraso o demora en el proceso.

- **Conector on-page:** el círculo funciona como conector entre dos puntos del diagrama de flujo. Este símbolo se utiliza como mínimo en parejas, con letras que identifican la conexión entre los puntos. Es útil para diagramas de flujo complejos, pues puedes ahorrar flechas.
- **Conector off-page:** es un símbolo con una función similar al anterior, pero para conectar con otros diagramas de flujo.

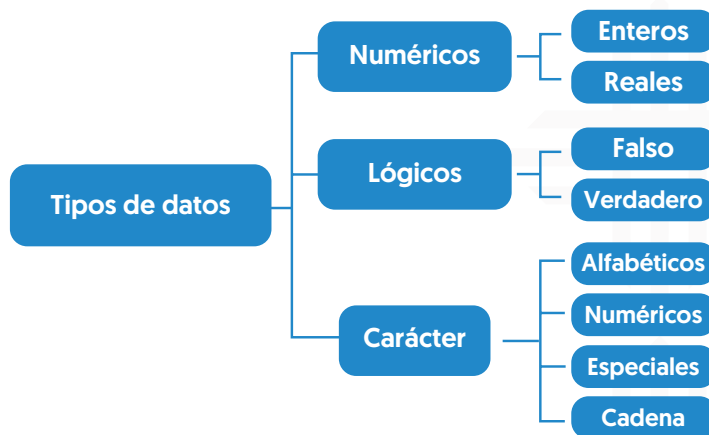
Además de estos símbolos, existen bastantes más que se emplean en diferentes ámbitos, pero no son tan comunes.

Datos

Los datos en programación son una representación de la información que maneja y procesa un programa. Son uno de los componentes fundamentales en el desarrollo de software.

Algunas características de los datos en programación:

- Representan valores concretos que pueden ser numéricos, texto, imágenes, audio, etc. Por ejemplo, un programa puede manejar datos como el nombre de una persona, su edad, su dirección, etc.
- Se almacenan en estructuras de datos que pueden ser variables, arreglos, registros, bases de datos, entre otros.
- Tienen diferentes tipos o formatos como números enteros, decimales, booleanos, caracteres, cadenas de texto, etc. El tipo de dato determina cómo se almacena y procesa.



Constante, variable y expresiones

Las constantes, variables y expresiones son conceptos fundamentales en programación:

Constante: es un valor que no cambia durante la ejecución de un programa. Se suelen escribir en mayúsculas por convención. Ejemplos: $PI = 3.1416$, $GRAVEDAD = 9.8$.

Variable: espacio de memoria con un nombre que contiene un valor que puede cambiar durante la ejecución de un programa. Se declaran con un tipo de dato. Ejemplos: $edad = 25$, $nombre = "Juan"$.

Expresión: es una combinación de constantes, variables, operadores y funciones que se evalúa para producir un único valor. Ejemplos: $\text{área} = \text{PI} * \text{radio}^2$, $\text{total} = \text{subtotal} + \text{impuesto}$.

Algunas características:

- Las constantes solo se pueden leer, las variables se pueden leer y modificar.
- Las variables deben declararse e inicializarse antes de usarse.
- Las expresiones siguen reglas y precedencia de operadores al evaluarse.
- Las expresiones siempre producen un resultado o valor al evaluarse.
- Las variables y expresiones facilitan la representación de valores que cambian durante el programa. Expresiones: Combinaciones de constantes, variables y operadores que producen un nuevo valor. Por ejemplo, " $x + 3$ " es una expresión que suma el valor de " x " con 3.

Expresiones	Variables	Constantes	Operaciones
$x+2$	x	1,2	Suma
$p-8$	p	1,8	Resta
$5y$	y	5	Multiplicación
$7/a$	a	1,7	División
$4b+1$	b	4,1	Multipl. y Suma
x^3	x	1,3	Multiplicación

Operadores aritméticos

Los operadores aritméticos son símbolos que se utilizan para realizar operaciones matemáticas y numéricas en la programación. Los operadores aritméticos más comunes son:

- Operador de adición (+):** se utiliza para sumar valores. Ej.: $2 + 3 = 5$
- Operador de sustracción (-):** se utiliza para restar valores. Ej.: $5 - 2 = 3$
- Operador de multiplicación (*):** se utiliza para multiplicar valores. Ej.: $3 * 4 = 12$
- Operador de división (/):** se utiliza para dividir valores. Ej.: $8 / 2 = 4$

- **Operador de módulo (%):** se utiliza para obtener el residuo o resto de la división. Ej.: $8 \% 3 = 2$
- **Operador de incremento (++):** se utiliza para incrementar un valor en 1. Ej.: $x++$
- **Operador de decremento (--):** se utiliza para disminuir un valor en 1. Ej.: $x--$

Estos operadores permiten realizar cálculos matemáticos en la programación como sumas, restas, multiplicaciones, divisiones, residuos, incrementos y decrementos de valores. Se pueden combinar y anidar operadores aritméticos para expresiones más complejas según las reglas de precedencia y asociatividad.

Operadores Aritméticos			
Operador	Descripción	Ejemplo	Resultado
+	Suma	$c=3+5$	$c=8$
-	Resta	$c=4-2$	$c=2$
-	Negación	$c=-7$	$c=-7$
*	Multiplicación	$c=3*6$	$c=18$
**	Potenciación	$c=2**3$	$c=8$
/	División	$c=7.5/2$	$c=3.75$
//	División entera	$c=7.5//2$	$c=3.0$
%	Módulo	$c=8\%3$	$c=2$

Operadores relacionales y lógicos

Los operadores relacionales y lógicos son símbolos que se utilizan para comparar valores y expresar condiciones lógicas en la programación:

Operadores relacionales

- **Operador de igualdad (==):** compara si dos valores son iguales. Ej.: $5 == 5$
- **Operador de desigualdad (!=):** compara si dos valores son diferentes. Ej.: $5 != 4$
- **Operador mayor que (>):** compara si un valor es mayor que otro. Ej.: $5 > 4$
- **Operador menor que (<):** compara si un valor es menor que otro. Ej.: $4 < 5$
- **Operador mayor o igual (>=):** compara si un valor es mayor o igual a otro.
- **Operador menor o igual (<=):** compara si un valor es menor o igual a otro.

Operadores lógicos

- **AND (&&):** evalúa si dos condiciones son verdaderas. Ej.: $[5 > 4] \&\& [3 < 6]$

OR (||): evalúa si al menos una condición es verdadera. Ej.: $[5 > 4] || [3 > 6]$

NOT (!): niega o invierte el valor lógico de una condición. Ej.: $![5 == 4]$

Operador	Significado
&&	AND
	OR
!	NOT

Estos operadores permiten evaluar comparaciones y expresar condiciones lógicas que derivan en valores booleanos (verdadero o falso). Son esenciales en la programación para controlar el flujo de ejecución según ciertas condiciones y tomar decisiones. Se usan mucho en estructuras de control como if-else y loops o bucles.

Operador	Significado	Operador	Significado
<	Menor a	<=	Menor o igual a
>	Mayor a	>=	Mayor o igual a
==	Igual a	!=	Diferente a



Valoremos lo aprendido

Explicamos con nuestras propias palabras qué es un algoritmo y por qué es importante en la programación.

- Escribamos un ejemplo de un programa mínimo en un lenguaje de programación de nuestra elección.
- Describamos una situación en la que usaríamos una estructura condicional "if-else" en un programa.
- Enumeremos al menos tres fases del proceso de programación y expliquemos brevemente qué sucede en cada una.



Unidad temática N° 2. Lenguaje de programación



Respondamos en nuestros cuadernos las siguientes preguntas

- ¿Qué es un lenguaje de programación y por qué se utiliza?
- Mencionemos las fases principales en el proceso de programación.
- Describamos el proceso de compilación en la programación.



Lenguaje de programación

Un lenguaje de programación es un conjunto de reglas y símbolos que se utilizan para escribir instrucciones que una computadora puede entender y ejecutar. Los programadores usan de programación para crear software y controlar el comportamiento de las computadoras.

Fases de la programación

La programación consta de varias fases que los programadores siguen para crear un software funcional. Estas fases hijo:

Análisis. Comprender el problema y los requisitos del software.

Diseño. Planear cómo se estructurará y funcionará el software.

Codificación. Escribir el código del programa utilizando un lenguaje de programación.

Compilación (o interpretación). Convertir el código fuente en un formato que la computadora pueda entender y ejecutar.

Prueba. Verificar que el software funciona según lo esperado.

Depuración. Identificar y corregir errores [bugs] en el código.

Implementación. Poner el software en uso en el entorno real.

Mantenimiento. Realizar mejoras y correcciones en el software a lo largo del tiempo.

Descripción de los menús de la herramienta de programación

Las herramientas de programación, como los entornos de desarrollo integrados (IDE), tienen interfaces gráficas que facilitan la escritura, compilación y depuración del código. Los menús principales de una herramienta de programación suelen incluir:

Estructura general de un programa

Un programa de computadora generalmente sigue una estructura específica. Aunque puede variar según el lenguaje de programación, suele incluir elementos como:

Declaraciones. Introducción de variables y constantes.

Instrucciones. Acciones que la computadora debe llevar a cabo.

Estructuras de control. Instrucciones condicionales [como "if"] y bucles [como "for" o "while"] para controlar el flujo del programa.

Funciones o métodos. Bloques de código que realizan tareas específicas y pueden ser reutilizados.

Codificación y ejecución de un programa mínimo

Un programa mínimo es un ejemplo simple de código que suele imprimir un mensaje en pantalla. Por ejemplo, en el lenguaje Python, un programa mínimo sería:

Python

```
print ("¡Hola, mundo!")
```

Proceso de compilación

El proceso de compilación es cuando el código fuente escrito por el programador se traduce a un lenguaje que la computadora puede entender y ejecutar. El resultado de este proceso es un archivo ejecutable o binario que puede ser ejecutado directamente.

Almacenamiento y recuperación de código fuente

El código fuente es el texto legible por humanos que los programadores escriben para crear software. Se almacena en archivos con extensiones como .py (Python), .java (Java) o .cpp (C++). Los sistemas de control de versiones, como Git, son herramientas que permiten a los programadores almacenar, gestionar y recuperar diferentes versiones del código fuente a lo largo del tiempo





Unidad temática N° 3. Estructuras básicas de programación



Respondamos en nuestros cuadernos las siguientes preguntas

- Expliquemos la diferencia entre estructuras secuenciales, condicionales y repetitivas.
- Describamos cómo funciona una estructura condicional "if-else".
- ¿Qué tipos de condiciones pueden utilizarse en las estructuras condicionales?



Estructuras básicas de programación

Las estructuras básicas de programación son patrones fundamentales que los programadores utilizan para organizar el flujo de ejecución de un programa. Estas estructuras permiten controlar el orden en que las instrucciones se ejecutan y cómo responden a diferentes situaciones.

Estructuras secuenciales

Las estructuras secuenciales son una serie de instrucciones que se ejecutan en orden, una después de la otra. Cada instrucción comienza después de que la anterior haya finalizado.

Ejemplo: Python

- `a = 5`
- `b = 3`
- `c = a + b`
- `print(c)`

Estructuras condicionales

Las estructuras condicionales permiten que un programa tome decisiones basadas en ciertas condiciones. Si una condición es verdadera, se ejecuta un conjunto de instrucciones; si es falsa, se puede ejecutar otro conjunto de instrucciones o simplemente omitir el bloque.

Tipos de condiciones

If. Se utiliza para ejecutar un bloque de código si una condición es verdadera.

If-else. Se utiliza para ejecutar un bloque de código si la condición es verdadera y otro bloque si la condición es falsa.

If-elif-else. Se utiliza para manejar múltiples condiciones. El bloque correspondiente al primer caso verdadero se ejecuta, y si no se cumple ninguna condición, se ejecuta el bloque "else".

Ejemplo (if-else): Python

```
edad = 18
if edad >= 18:
    print ("Eres mayor de edad")
else:
    print("Eres menor de edad")
```

Estructuras repetitivas

Las estructuras repetitivas permiten que un bloque de código se repita varias veces. Esto es útil para realizar tareas similares de manera eficiente.

while: se ejecuta mientras una condición sea verdadera.

For: se utiliza para iterar sobre una secuencia (como una lista, rango numérico, etc.).

Ejemplo (while): Python

```
contador = 0
while contador < 5:
    print("Contador:", contador)
```


● contador += 1

Ejemplo (for): Python

● for i in range(5):

● print ("Valor:", i)

Estas estructuras son fundamentales en la programación y te permitirán crear programas más complejos y funcionales. Recuerda que puedes combinar estas estructuras según sea necesario para lograr tus objetivos de programación



Valoremos lo aprendido

Respondamos las siguientes incógnitas en nuestro cuaderno:

- ¿Sabes qué es PYTHON y por qué es un lenguaje de programación tan importante?
- Valoremos, analicemos y redactemos cinco falacias que suceden en nuestra sociedad, viendo las noticias.

Preguntas de verdadero o falso

- Un algoritmo debe tener un número infinito de pasos para ser efectivo. [Verdadero / Falso]
- Los operadores lógicos se utilizan para realizar operaciones matemáticas. [Verdadero / Falso]
- El proceso de compilación convierte el código fuente en un archivo ejecutable. [Verdadero / Falso]
- La estructura de repetición "for" se utiliza para ejecutar un bloque de código mientras una condición sea verdadera. [Verdadero / Falso]

Módulo IV

Hardware de computadoras I



Objetivo holístico del módulo

Desarrollamos conocimientos, habilidades, destreza y técnicas sobre hardware y software de computadoras, con capacidad de dilucidar problemas y resolverlos en área laboral y en sus estudios teniendo los conocimientos necesarios para reconocer las partes y los componentes de una computadora, instalación de software y aplicaciones, seguridad y mantenimiento correcto del equipo, además del manejo del equipo y de Microsoft Office y sus herramientas de diseño gráfico.



Unidad temática N° 1. Evolución y generación de computadoras



Investiguemos a fondo

- Máquina electrónica capaz de almacenar información y tratarla automáticamente mediante operaciones matemáticas y lógicas controladas por programas informáticos, llamada computadora.
- Escribamos en nuestro cuaderno las partes físicas externas de una computadora de escritorio



Evolución y generación de computadoras

¿Para qué sirve la tarjeta gráfica de un sistema informático?

Una parte esencial de un sistema informático es la tarjeta gráfica, también conocida como unidad de procesamiento de gráficos [GPU]. Su función principal es procesar y mostrar gráficos, imágenes y vídeos. La GPU está diseñada específicamente para realizar el paralelismo intensivo requerido para aplicaciones de gráficos y visualización, como juegos, edición de video y diseño 3D.

¿Qué está relacionado con el overclocking con el hardware de la computadora?

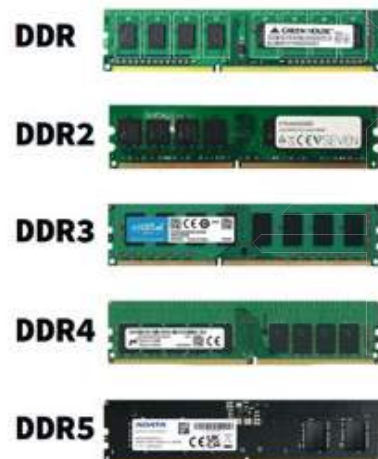
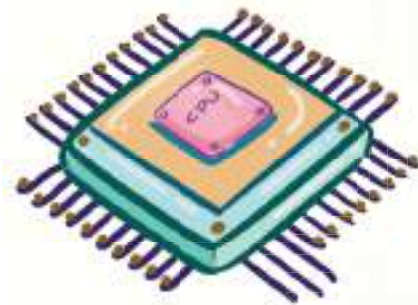
El overclocking es la práctica de aumentar la velocidad de un componente de hardware, como un procesador o una tarjeta gráfica, más allá de las especificaciones especificadas por el fabricante. Esto es para mejorar el rendimiento en términos de velocidad del procesador. Sin embargo, el overclocking puede aumentar las temperaturas y el consumo de energía, lo que puede provocar problemas de estabilidad o acortar la vida útil de los componentes si no se realiza correctamente.



Conozcamos más sobre la historia de la computadora, su estructura y servicios básicos de los sistemas y aplicaciones.

El hardware informático son todos los componentes físicos y tangibles que componen un sistema informático. Se incluyen en esta categoría los aparatos electrónicos, circuitos, cables, extras y cualquier otra cosa que pueda manipularse o tocarse físicamente. El hardware funciona con el software, es decir, programas y aplicaciones que se ejecutan en la computadora. El hardware de la computadora se divide en diferentes categorías según su función y conexión al sistema. Éstos son algunos de los principales tipos de hardware:

- Unidad central de procesamiento [CPU]. El "cerebro" de la computadora se llama procesador. Es responsable de ejecutar instrucciones de software y realizar cálculos aritméticos y lógicos. Un procesador tiene uno o más núcleos que procesan datos al mismo tiempo.
- RAM. La memoria de acceso aleatorio [RAM]. Es el almacenamiento temporal de datos e instrucciones que el procesador necesita para realizar operaciones. La RAM es volátil, lo que significa que su contenido se borra cuando se apaga la computadora.
- Disco duro o unidad de estado sólido [SSD]. Estos dispositivos de almacenamiento son responsables de almacenar datos de forma permanente en su computadora. Las unidades



de estado sólido utilizan memoria flash más rápida y no tienen partes móviles a diferencia de los discos duros, que utilizan discos magnéticos para almacenar datos.

- Placa base (placa base). Es el componente principal que conecta todos los demás componentes de la computadora. Los circuitos y conectores necesarios para conectar los demás componentes están presentes en la placa base.
- Tarjeta gráfica (GPU). Responsable de procesar y crear la imagen que se muestra en la pantalla. Las tarjetas gráficas también se utilizan en aplicaciones con uso intensivo de gráficos, como videojuegos y diseño gráfico.
- Dispositivos periféricos de entrada y salida. Estos son dispositivos que se conectan a su computadora para interactuar con ella. Ejemplos de dispositivos cotidianos incluyen teclados, ratones, escáneres, impresoras y parlantes.



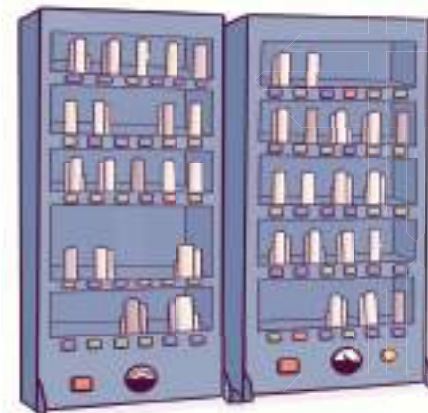
A continuación, se enumeran solo algunos de los tipos más comunes de hardware informático. La selección y configuración adecuadas son esenciales para satisfacer sus necesidades porque cada pieza de hardware es crucial para el funcionamiento de su computadora.



Evolución de las computadoras

Actualmente existen hasta cinco generaciones de ordenadores, pero algunos van más allá y mencionan frecuentemente la octava generación, que es un futuro lejano y ni siquiera ha sido creada. El desarrollo informático en este punto normalmente se clasifica de la siguiente manera:

Primera generación. Las computadoras de primera generación fueron los primeros dispositivos electrónicos de computación desarrollados en las décadas de 1940 y 1950. Eran enormes, ocupaban salas enteras y se basaban en válvulas de vacío para realizar cálculos. Estas computadoras tenían una capacidad de procesamiento limitada y requerían un mantenimiento constante. Aunque eran mucho más lentas y menos potentes en comparación con las computadoras modernas, sentaron las bases para el desarrollo de la tecnología informática.



Segunda generación. Las computadoras de segunda generación fueron desarrolladas en la década de 1950 y principios de la década de 1960. A diferencia de las computadoras de primera generación, estas utilizaron transistores en lugar de válvulas de vacío, lo que permitió un tamaño más pequeño, mayor confiabilidad y un rendimiento mejorado. Estas computadoras también introdujeron el uso de circuitos integrados, lo que llevó a una mayor capacidad de procesamiento y la reducción de costos. Aunque aún eran considerablemente grandes y costosas, las computadoras de segunda generación sentaron las bases para el desarrollo de tecnologías más avanzadas en el campo de la informática.



Las computadoras de tercera generación. Desarrolladas a fines de la década de 1960 y durante la década de 1970, introdujeron importantes avances tecnológicos en comparación con las generaciones anteriores. Estas computadoras utilizaban circuitos integrados de mayor densidad, lo que permitía un mayor rendimiento y una reducción significativa en el tamaño y el costo de los sistemas informáticos. Además, se introdujo la tecnología de microprocesadores, lo que llevó a un aumento aún mayor en la capacidad de procesamiento y la creación de computadoras más potentes y eficientes. Las computadoras de tercera generación también fueron las primeras en utilizar sistemas operativos y lenguajes de programación de alto nivel, lo que facilitó la programación y el uso de las computadoras.



Cuarta generación. Las computadoras de cuarta generación, desarrolladas a partir de la década de 1980, representaron un gran avance en la tecnología informática. Estas computadoras se caracterizaron por la introducción de los microprocesadores de alta velocidad, lo que permitió un aumento significativo en la capacidad de procesamiento y el rendimiento de las computadoras. Además, se implementaron avances en la tecnología de memoria, lo que permitió un acceso más rápido y eficiente a los datos. Las computadoras de cuarta generación también se beneficiarán de mejoras en la capacidad de almacenamiento, con la introducción de discos duros y dispositivos de almacenamiento óptico. Estas computadoras también fueron más pequeñas y más económicas en comparación con las generaciones anteriores.



Quinta generación. Las computadoras de quinta generación, desarrolladas a partir de la década de 1980, se caracterizaron por importantes avances en la inteligencia artificial y el procesamiento paralelo. Estas computadoras fueron capaces de realizar tareas más complejas, como el reconocimiento de voz y el procesamiento de lenguaje natural. Además, se enfocaron en el desarrollo de sistemas expertos y el uso de la lógica difusa para realizar inferencias y tomar decisiones más sofisticadas. Las computadoras de quinta generación también se beneficiarán de mejoras en la capacidad de almacenamiento y la velocidad de procesamiento.



Las computadoras de sexta generación. Aún se encuentran en desarrollo, están centradas en la integración de tecnologías como la inteligencia artificial, la realidad virtual y aumentada, y la computación cuántica. Estas computadoras se esperan que sean capaces de procesar grandes cantidades de datos de manera más eficiente y rápida, y se espera que sean utilizadas en aplicaciones avanzadas como el reconocimiento facial, la traducción automática y la conducción autónoma.



La séptima generación de computadoras. Aún no ha sido completamente definida, pero se espera que se enfoque en el desarrollo de tecnologías como la computación neuromórfica y la computación en la nube. Estas computadoras podrían ser capaces de simular el funcionamiento del cerebro humano y realizar tareas complejas de manera más eficiente y precisa.

La octava generación de computadoras. Es una proyección futura y aún no ha sido completamente definida. Sin embargo, se espera que se centren en avances tecnológicos aún más avanzados, como la computación cuántica a gran escala y la integración de la inteligencia artificial en todos los aspectos de la vida cotidiana. Estas computadoras podrían ser capaces de resolver problemas aún más complejos y realizar tareas que actualmente están fuera del alcance de las computadoras convencionales.

Es importante tener en cuenta que estas generaciones de computadoras están en constante evolución y desarrollo, y es posible que su definición y características cambien a medida que avanza la tecnología.

Generación de computadoras

Una generación de computadoras se refiere a un período en el que se lanza al mercado un conjunto de dispositivos con tecnologías, características y capacidades similares, aunque puedan ser fabricados por diferentes empresas.



**Primera
Generación**
1951-1958



**Cuarta
Generación**
1978-1988



**Segunda
Generación**
1958-1964



**Quinta
Generación**
1983-Actual



**Tercera
Generación**
1965-1971



**Sexta
Generación**
Actual



Unidad temática N° 2. Partes de una computadora

Procesadores

Un procesador es un componente fundamental en un sistema informático que se encarga de ejecutar las instrucciones y realizar los cálculos necesarios para el funcionamiento de una computadora. También se le conoce como unidad central de procesamiento (CPU, por sus siglas en inglés). El procesador es responsable de realizar operaciones aritméticas y lógicas, acceder y manipular datos en la memoria, controlar el flujo de datos y ejecutar programas.

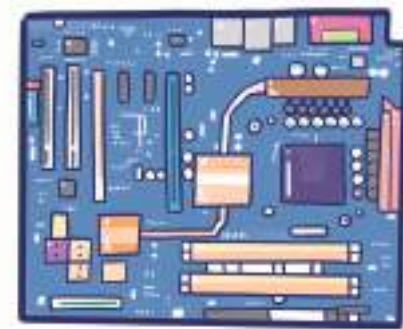


Tarjeta madre

Una tarjeta madre, también conocida como placa base, es un componente esencial en una computadora que se encarga de conectar y proporcionar energía a todos los demás componentes. Es el componente principal que permite la comunicación entre el procesador, la memoria, los dispositivos de almacenamiento y otros periféricos.

La tarjeta madre contiene varios conectores y puertos para permitir la conexión de diferentes componentes, como el procesador, la memoria RAM, las tarjetas de expansión (como tarjetas gráficas y tarjetas de sonido), los discos duros, los puertos USB y otros dispositivos de entrada y salida.

Además de proporcionar conexiones físicas, la tarjeta madre también contiene circuitos y chips que controlan y facilitan la comunicación entre los diferentes componentes. También puede incluir características adicionales, como controladores de red, controladores de audio y puertos de expansión.



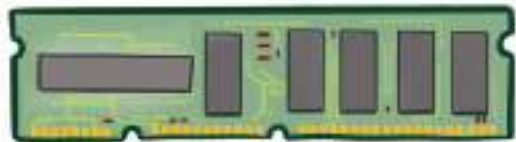
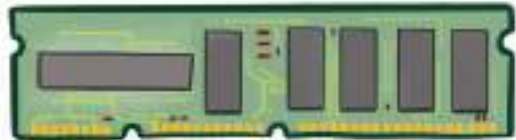
Tarjeta de video

Una tarjeta de video, también conocida como GPU [Unidad de Procesamiento Gráfico], es un componente esencial en una computadora que se encarga de procesar y renderizar imágenes y videos en una pantalla. Está diseñada para acelerar tareas relacionadas con gráficos y juegos, aliviando la carga del procesador principal. Las tarjetas de video varían en rendimiento y capacidades, desde modelos básicos para tareas de oficina hasta GPUs de alto rendimiento para juegos y diseño gráfico.

Se conectan a la placa madre y su rendimiento se mide en términos de velocidad de reloj, memoria VRAM y unidades de procesamiento.

Memoria RAM

La memoria RAM, también conocida como Memoria de Acceso Aleatorio, es un elemento fundamental en una computadora que tiene la función de almacenar temporalmente los datos y las instrucciones necesarias para que el procesador pueda llevar a cabo sus operaciones. La RAM cumple el papel de ser una especie de "memoria de trabajo" de la computadora, en la cual los datos pueden ser accedidos y modificados de forma rápida y eficiente.



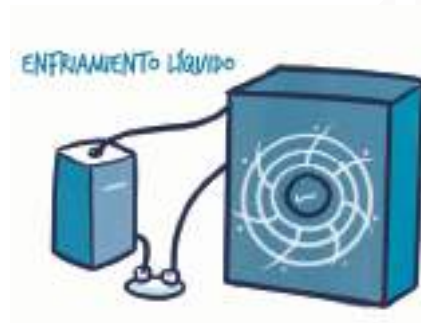
Discos duros

El disco duro, también conocido como unidad de almacenamiento o HDD (Hard Disk Drive, por sus siglas en inglés), es un componente esencial en una computadora que se encarga de almacenar y recuperar datos de forma persistente. El disco duro utiliza un sistema de grabación magnética para guardar la información en discos o platos giratorios recubiertos de material magnético.



Refrigeración

La refrigeración es un proceso fundamental que se utiliza para disipar el calor generado por los componentes electrónicos de una computadora u otro dispositivo. Este proceso se lleva a cabo mediante el uso de sistemas de ventilación o enfriamiento, como ventiladores, disipadores de calor y líquidos refrigerantes, con el fin de mantener una temperatura adecuada y evitar el sobrecalentamiento.



de los componentes. La refrigeración es esencial para el correcto funcionamiento y la prolongación de la vida útil de los dispositivos electrónicos.

Case

El "case" [caja o carcasa] es un componente físico en una computadora que proporciona un alojamiento y protección para los demás componentes internos. También se le conoce como chasis. El estuche está diseñado para contener y organizar los componentes de la computadora, como la placa madre, la fuente de alimentación, el disco duro y otros dispositivos de almacenamiento, así como las tarjetas de expansión y otros periféricos.



Conexiones

Las conexiones son enlaces o vínculos establecidos entre diferentes dispositivos electrónicos o componentes para permitir la transmisión de datos, información o señales entre ellos. Estas conexiones pueden ser físicas, como cables o cables de red, o inalámbricas, como conexiones Wi-Fi o Bluetooth. También pueden incluir puertos o interfaces específicas, como puertos USB, HDMI o Ethernet, que permiten la conexión y comunicación entre dispositivos. En resumen, las conexiones son los medios utilizados para establecer la comunicación y la transferencia de datos entre dispositivos electrónicos.



Puerto VGA. El puerto VGA es una conexión física utilizada para transmitir señales de vídeo analógicas desde una fuente, como una computadora, un dispositivo de visualización, como un monitor o un proyector. Esta conexión utiliza un conector de 15 pines y se utiliza en computadoras más antiguas. El puerto VGA permite una transmisión de vídeo de alta calidad, aunque está siendo reemplazado gradualmente por conexiones digitales, como HDMI y DisplayPort, que ofrecen una mayor calidad de imagen.



Puerto DVI. El puerto DVI es una conexión física utilizada para transmitir señales de vídeo digital desde una fuente, como una computadora o reproductor de DVD, a un dispositivo de visualización, como un monitor o un televisor. Este puerto utiliza un conector de 24 pines y es ampliamente utilizado en

dispositivos de alta definición. El puerto DVI permite una transmisión de vídeo de alta calidad y es compatible con diferentes resoluciones y frecuencias de actualización. Sin embargo, está siendo reemplazado gradualmente por conexiones más modernas, como HDMI y DisplayPort, que ofrecen una mayor capacidad de transmisión de datos y soporte para audio.

Conector con HDMI. El conector HDMI es una conexión física utilizada para transmitir señales de alta definición de video y audio desde una fuente, como un reproductor de Blu-ray o una consola de videojuegos, un dispositivo de visualización, como un televisor o un proyector. Este conector utiliza un cable que combina tanto la señal de video como la señal de audio en un solo cable, lo que simplifica la conexión y evita la necesidad de cables separados para el video y el audio. El conector HDMI es ampliamente utilizado debido a su capacidad para transmitir señales de alta calidad y soportar resoluciones de vídeo de hasta 4K. También es compatible con formatos de audio de alta calidad, como Dolby Digital y DTS.



Conector DisplayPort. El conector DisplayPort es una conexión física utilizada para transmitir señales de vídeo y audio de alta definición desde una fuente, como una computadora o una tarjeta gráfica, a un dispositivo de visualización, como un monitor o una televisión. Este conector ofrece una alta capacidad de transmisión de datos y es capaz de soportar resoluciones de vídeo de hasta 8K. Además, el conector DisplayPort permite la transmisión de audio de alta calidad y es compatible con tecnologías adicionales, como la transmisión de múltiples pantallas y la sincronización adaptativa. Es una conexión cada vez más popular en dispositivos de última generación debido a su versatilidad y capacidad de ofrecer una experiencia visual y auditiva inmersiva.



Accesorios

Instalación de sistemas operativos

En esta unidad conocerás acerca de la instalación de los sistemas, sus características, definición: La instalación de un sistema operativo es simplemente colocar en un lugar del disco duro del equipo para que funcione correctamente y así realice la función que le corresponde. De esta manera permitir la ejecución de las operaciones necesarias para el funcionamiento del equipo de cómputo.

Configuración del BIOS

El sistema básico de entrada/salida (BIOS) es un componente crítico del hardware de una computadora responsable de inicializar y configurar los componentes de hardware durante el proceso de arranque. Configurar correctamente los ajustes del BIOS es esencial para la estabilidad y el rendimiento del sistema.

Para acceder al BIOS, normalmente presione una tecla designada (por ejemplo, F2, Del o Esc) durante el inicio del sistema. Dentro del BIOS, encontrará varias configuraciones, que incluyen:

- Orden de inicio: puede especificar el orden en el que su computadora busca dispositivos de inicio. Asegúrese de que su unidad de arranque principal esté configurada correctamente (por ejemplo, SSD o HDD).
- Fecha y hora: establezca la fecha y hora del sistema para garantizar marcas de tiempo precisas.
- Seguridad: configure contraseñas del BIOS o habilite opciones de inicio seguro para mayor seguridad.
- Configuración de la CPU: ajuste la configuración relacionada con la CPU, como la velocidad del reloj y las opciones de virtualización.
- Configuración de memoria: configure los ajustes de RAM, como la velocidad y los tiempos.
- Configuración de almacenamiento: verifique los modos de unidad (AHCI, RAID) y habilite/deshabilite los dispositivos.
- Periféricos integrados: personalice la configuración de los componentes integrados, como controladores USB, audio y adaptadores de red.
- Administración de energía: controle las funciones de ahorro de energía, como la configuración ACPI y los modos de suspensión.
- Monitoreo de hardware: verifique las temperaturas, las velocidades del ventilador y las lecturas de voltaje para verificar el estado del sistema.
- Opciones avanzadas: algunas placas base ofrecen configuraciones avanzadas para overclocking o ajustes de hardware específicos.

Instalación del sistema operativo

La instalación del sistema operativo es un proceso esencial para poner en funcionamiento una

computadora. Comienza con la inserción de un medio de instalación, como un DVD o una unidad USB con los archivos del sistema operativo. Al arrancar la computadora desde este medio, se inicia el proceso de instalación.

Durante la instalación, se deben tomar decisiones importantes, como seleccionar el idioma y la región, proporcionar una clave de producto en el caso de sistemas operativos como Windows, y elegir la unidad o partición donde se instalará el sistema operativo. También es posible formatear la unidad si es necesario.

Una vez completadas estas opciones, el proceso de instalación copia archivos del sistema operativo a la computadora y configura los componentes básicos. Después de la instalación, la computadora se reinicia y se realizan configuraciones adicionales, como la creación de cuentas de usuario y la configuración de preferencias.

Configuración e instalación de controladores

La configuración e instalación de controladores son pasos cruciales para asegurar que los dispositivos de hardware de una computadora funcionen correctamente con el sistema operativo. Aquí hay un resumen:

Configuración de controladores

La configuración de controladores generalmente comienza con la instalación del sistema operativo. Este proceso incluye una selección básica de controladores genéricos para permitir el funcionamiento esencial de los componentes de hardware, como el teclado y el mouse.

Instalación de controladores

Después de instalar el sistema operativo, se requiere la instalación de controladores específicos para cada dispositivo. Esto se puede hacer de varias maneras:

Uso de CD/DVD de Controladores: Algunos fabricantes proporcionan discos con controladores específicos que se pueden utilizar para la instalación. Simplemente inserta el disco y sigue las instrucciones.

Descarga desde el Sitio Web del Fabricante: Es común descargar los controladores más recientes directamente desde el sitio web del fabricante del dispositivo. Esto asegura que estés utilizando la versión más actualizada.

Uso de Administrador de Dispositivos (Windows): En Windows, puedes usar el Administrador de Dispositivos para buscar automáticamente e instalar controladores actualizados para los dispositivos reconocidos.

Paquetes de Controladores [Linux]: En sistemas basados en Linux, los paquetes de controladores se pueden instalar desde los repositorios del sistema o descargarse manualmente.

Instalación de software de aplicación

La instalación de software de aplicación es un proceso fundamental para agregar nuevas funcionalidades y programas a una computadora. Aquí hay un resumen de este proceso:

1. Descarga o adquisición del software. El primer paso consiste en obtener el software de aplicación. Esto puede implicar descargarlo desde internet o instalarlo desde un medio físico como un DVD. Es importante asegurarse de que el software sea seguro y legítimo, especialmente cuando se descarga de fuentes en línea.
2. Iniciar el asistente de instalación. Una vez que se tiene el archivo de instalación del software, se inicia el proceso de instalación haciendo doble clic en él o ejecutándolo desde el medio físico. Esto suele abrir un asistente de instalación que guía al usuario a través del proceso.
3. Configuración de opciones. Durante la instalación, se pueden configurar opciones específicas, como el directorio de instalación, la creación de accesos directos en el escritorio o el menú de inicio, y la selección de idioma.
4. Proceso de instalación. El asistente de instalación copia los archivos necesarios del software en el sistema y realiza configuraciones específicas. Esto puede tomar unos minutos.
5. Finalización de la instalación: Una vez que la instalación está completa, se muestra un mensaje de confirmación. Algunos programas pueden requerir reiniciar la computadora para que los cambios surtan efecto.
6. Registro y activación. En algunos casos, se puede requerir registrar o activar el software utilizando una clave de licencia. Esto garantiza que el software se utilice de acuerdo con los términos de la licencia.
7. Uso del software. Después de la instalación, el software de aplicación está listo para su uso. Se puede acceder a él desde el menú de inicio, el escritorio o la ubicación donde se instaló.

Instalación de antivirus

La instalación de un antivirus es un paso crítico para proteger una computadora contra amenazas de malware y virus. Aquí hay un resumen de este proceso:

1. Descarga del antivirus: Comienza por obtener el antivirus deseado. Puedes descargarlo desde el sitio web oficial del proveedor de antivirus o utilizar una fuente de confianza.

2. Ejecución del archivo de instalación. Una vez descargado, ejecuta el archivo de instalación del antivirus haciendo doble clic en él.
3. Asistente de instalación. Se abrirá un asistente de instalación que te guiará a través del proceso. A menudo, tendrás que aceptar los términos y condiciones del software.
4. Configuración de opciones. Durante la instalación, podrás configurar opciones como el directorio de instalación y las características específicas que deseas incluir, como análisis en tiempo real o programado.
5. Actualización de definiciones. Después de la instalación inicial, el antivirus suele buscar actualizaciones de definiciones de virus y malware para asegurarse de que esté al día en la detección de amenazas.
6. Escaneo inicial. Algunos antivirus realizan un escaneo inicial del sistema para identificar y eliminar cualquier amenaza preexistente.
7. Activación o registro. Es posible que debas activar o registrar el antivirus utilizando una clave de licencia proporcionada por el proveedor. Esto garantiza que el software esté legalmente autorizado.
8. Configuración adicional. Puedes personalizar la configuración del antivirus según tus preferencias, como las áreas específicas que deseas escanear o las notificaciones que deseas recibir.
9. Protección en tiempo real. Una vez instalado y configurado, el antivirus protegerá tu computadora en tiempo real, escaneando archivos y actividades en busca de amenazas.
10. Actualizaciones regulares. Mantén el antivirus actualizado descargando las últimas definiciones y actualizaciones de software para garantizar una protección efectiva.

¿Qué es el IoT o internet de las cosas?

El término Internet de las Cosas, o IoT por sus siglas en inglés [Internet of Things], se refiere a la capacidad de interconectar y transmitir datos entre objetos cotidianos y la internet.

Internet de las cosas IoT

El término IoT apareció a finales de los noventa y fue definido por Kevin Ashton en el contexto de la gestión de la cadena de suministro [el proceso de obtención de materiales, la transformación de estos en productos y la distribución de los productos terminados a los consumidores]. El IoT es un paradigma que brinda la base y el modelo para resolver la problemática de tener interconectadas todas las “cosas” que nos rodean.

Paulatinamente conocerás otras aplicaciones que pueden ayudar a entender mejor esta unidad de estudio para ello se recomienda utilizar Arduino, fácil manejo, accesible y útil.



Instalar un Sistema Operativo desde CERO



Sistemas Ciberfísicos CPS en la visión industria 4.0

En la visión de la industria 4.0, los Sistemas Ciberfísicos [CPS] se refieren a mecanismos o sistemas físicos que son controlados o monitoreados por algoritmos basados en computación y están estrechamente integrados en Internet. Estos sistemas son componentes clave de la industria 4.0 y están transformando la forma en que los seres humanos interactúan con el entorno físico al integrarse con el mundo cibernético. El objetivo principal de la implementación de los sistemas CPS es mejorar la calidad de los productos, ya sea dentro o fuera de la Internet de las Cosas [IoT].



Internet de las cosas y sistemas ciberfísicos

El Internet de las Cosas [IoT] y los Sistemas Ciberfísicos [CPS] son dos conceptos interrelacionados que transforman la forma en que interactuamos con el mundo digital y físico. Aquí tienes un resumen de ambos:

Internet de las Cosas [IoT]: El IoT se refiere a la interconexión de dispositivos físicos mediante internet. Estos dispositivos pueden ser cualquier cosa, desde electrodomésticos y sensores industriales hasta relojes inteligentes y automóviles. La clave es que pueden recopilar, transmitir y recibir datos, lo que permite el monitoreo y control remoto. Los ejemplos incluyen termostatos inteligentes que ajustan la temperatura automáticamente y sensores agrícolas que optimizan el riego según las condiciones climáticas.

Sistemas Ciberfísicos (CPS): Los CPS son sistemas integrados en los que componentes físicos y virtuales interactúan de manera estrecha y coordinada. Estos sistemas combinan hardware y software para controlar y gestionar procesos del mundo real. Ejemplos incluyen vehículos autónomos que utilizan sensores para navegar y máquinas de producción automatizadas que ajustan la producción en tiempo real según la demanda. Los CPS mejoran la eficiencia y la capacidad de respuesta de los sistemas físicos.

Ecosistema de dispositivos

Un ecosistema de dispositivos se refiere a un conjunto diverso de dispositivos electrónicos interconectados que trabajan juntos para proporcionar una experiencia integral y funcional. Aquí tienes un resumen de este concepto:

Interconexión. En un ecosistema de dispositivos, los componentes pueden comunicarse y colaborar entre sí a través de redes, como Wi-Fi o Bluetooth. Esto permite compartir datos y controlar los dispositivos de manera remota.

Diversidad. Los dispositivos en un ecosistema pueden variar ampliamente en términos de funciones y propósitos. Pueden incluir teléfonos inteligentes, tabletas, computadoras, electrodomésticos inteligentes, relojes inteligentes y más.

Colaboración. La colaboración es fundamental en un ecosistema de dispositivos. Por ejemplo, un teléfono inteligente puede conectarse a una impresora y enviar documentos para imprimir, o un termostato inteligente puede comunicarse con un sistema de iluminación para ajustar automáticamente la temperatura y la iluminación según las preferencias del usuario.

Sincronización y control. Los dispositivos pueden sincronizarse para funcionar en conjunto de manera armoniosa. Un usuario puede controlar varios dispositivos desde una aplicación centralizada o a través de comandos de voz.

Experiencia del usuario. Un ecosistema de dispositivos busca proporcionar una experiencia de usuario fluida y conveniente al permitir que los dispositivos trabajen juntos de manera eficiente y simplificando la vida cotidiana.

Plataforma IOT y su arquitectura

Una plataforma IoT es una infraestructura tecnológica que permite la interconexión de dispositivos, la recopilación de datos, el procesamiento y el análisis de información, y la gestión de aplicaciones IoT. Estas plataformas facilitan la creación de soluciones IoT eficientes y escalables.

Arquitectura de una plataforma IoT

La arquitectura de una plataforma IoT consta de varios componentes clave:

- **Dispositivos IoT.** Son sensores, actuadores o dispositivos conectados que recopilan datos del mundo físico, como temperaturas, humedad o información de ubicación.
- **Conectividad.** La plataforma IoT proporciona la infraestructura para conectar los dispositivos a través de protocolos de comunicación, como Wi-Fi, Bluetooth, LoRa, o redes celulares.
- **Procesamiento de datos.** Los datos recopilados se envían a una capa de procesamiento que puede incluir la normalización de datos, filtrado y agregación para prepararlos para su análisis.
- **Almacenamiento de datos.** Los datos se almacenan en bases de datos que pueden ser SQL o NoSQL, lo que permite un acceso rápido y eficiente.
- **Plataforma de aplicación.** Esta capa es donde se desarrollan y ejecutan las aplicaciones IoT. Puede incluir lógica de negocio, reglas y algoritmos para tomar decisiones basadas en datos.
- **Seguridad.** La seguridad es fundamental en una plataforma IoT para proteger los datos y dispositivos. Esto incluye autenticación, autorización, cifrado y monitoreo de amenazas.
- **Interfaz de usuario.** Los usuarios interactúan con la plataforma a través de interfaces de usuario, aplicaciones móviles o tableros de control web para supervisar y controlar dispositivos y datos.
- **Integración con servicios externos.** Las plataformas IoT a menudo se integran con otros servicios, como servicios en la nube, análisis avanzados o sistemas de gestión empresarial.
- **Gestión de dispositivos.** La plataforma debe permitir la gestión de dispositivos IoT a lo largo de su ciclo de vida, incluida la configuración, actualización de firmware y detección de fallos.

Digital Twins

Un "Digital Twin" o "Gemelo Digital" es una representación virtual en tiempo real de un objeto físico, proceso o sistema. Aquí tienes un resumen de este concepto:

- **Definición.** Un Digital Twin es una réplica digital precisa de un objeto físico o entidad, que incluye su geometría, características, estado y comportamiento.
- **Interconexión.** Los Digital Twins están conectados de manera bidireccional con su contraparte física, lo que significa que pueden recibir datos en tiempo real desde sensores y otros dispositivos conectados al objeto físico.

- **Modelado y simulación.** Los Digital Twins utilizan modelos matemáticos y algoritmos para simular el comportamiento y rendimiento del objeto físico en diferentes situaciones y condiciones.
- **Aplicaciones.** Se utilizan en una variedad de industrias, como manufactura, energía, salud y transporte, para mejorar la toma de decisiones, predecir problemas y optimizar el rendimiento.
- **Mantenimiento predictivo.** Los Digital Twins permiten el monitoreo constante de activos físicos, lo que facilita la detección temprana de problemas y la planificación de mantenimiento predictivo.
- **Optimización de diseño.** En la fase de diseño, los Digital Twins ayudan a probar y refinar modelos antes de construir objetos físicos, lo que ahorra tiempo y recursos.
- **Mejora de procesos.** En la industria, se utilizan para optimizar procesos y operaciones, mejorando la eficiencia y la productividad.
- **Evolución continua.** Los Digital Twins evolucionan con el tiempo a medida que se recopilan más datos y se mejoran los modelos, lo que permite un rendimiento cada vez más preciso.

Indoor & Outdoor Geolocation (Real Time Geospatial)

La geolocalización en interiores y exteriores, también conocida como georreferenciación en tiempo real, se refiere a la capacidad de rastrear y determinar la ubicación de objetos o dispositivos en entornos tanto cerrados como al aire libre utilizando tecnología geoespacial. Aquí tienes un resumen de este concepto:

Geolocalización en interiores

La geolocalización en interiores se utiliza para rastrear la ubicación de objetos o personas dentro de edificios, centros comerciales, hospitales y otros espacios cerrados donde el GPS tradicional puede no ser preciso. Se basa en tecnologías como el Wi-Fi, Bluetooth, RFID y sensores de proximidad para determinar la posición en tiempo real.

Geolocalización en exteriores

La geolocalización en exteriores utiliza el sistema de posicionamiento global (GPS) y otras tecnologías de satélite para rastrear la ubicación de dispositivos o personas en áreas al aire libre. Es ampliamente utilizado en aplicaciones de navegación, mapas y seguimiento de vehículos.

Aplicaciones

Estas tecnologías tienen una amplia gama de aplicaciones, desde ayudar a los usuarios a encontrar su camino en centros comerciales hasta permitir el seguimiento de flotas de vehículos, mejorar la seguridad en entornos industriales y brindar información geoespacial en tiempo real para aplicaciones de mapeo y cartografía.

Precisión y cobertura

La geolocalización en exteriores tiende a ser más precisa y tener una cobertura más amplia debido a la disponibilidad de señales de satélite, mientras que la geolocalización en interiores puede variar en precisión según la tecnología utilizada y la densidad de puntos de acceso.

Desafíos de privacidad

Sistema de seguridad inteligente

Un Sistema de Seguridad Inteligente es una solución tecnológica avanzada diseñada para proteger propiedades, personas y activos utilizando tecnologías de vanguardia y análisis de datos en tiempo real. Aquí tienes un resumen de este concepto:

Características principales

- **Sensores avanzados.** Utiliza sensores como cámaras de vigilancia, detectores de movimiento, sensores de humo y otros dispositivos para detectar amenazas y actividades sospechosas.
- **Conectividad.** Está interconectado a través de redes de comunicación, lo que permite el monitoreo y el control desde dispositivos móviles y otros sistemas.
- **Análisis de datos.** Emplea análisis de datos y algoritmos avanzados para identificar patrones de comportamiento y alertar sobre eventos potencialmente peligrosos.
- **Automatización.** Puede automatizar respuestas, como activar alarmas, cerrar puertas o notificar a las autoridades, sin intervención humana.
- **Integración.** Puede integrarse con otros sistemas, como sistemas de acceso, iluminación y control de climatización, para una seguridad y comodidad mejoradas.

Aplicaciones

- **Seguridad residencial.** Protege hogares contra robos, intrusiones y situaciones de emergencia como incendios.

- **Seguridad empresarial.** Salvaguarda oficinas, almacenes y empresas mediante la supervisión constante y la respuesta inmediata a amenazas.
- **Seguridad vehicular.** Incluye sistemas de rastreo y seguridad en vehículos para prevenir robos y garantizar la seguridad de los ocupantes.
- **Seguridad industrial.** Monitoriza instalaciones industriales y refuerza la seguridad en entornos peligrosos.
- **Seguridad en ciudades inteligentes.** Ayuda a mantener la seguridad pública, supervisando áreas urbanas y proporcionando información a las fuerzas del orden.

Beneficios

- **Prevención y detección temprana.** Detecta amenazas antes de que se conviertan en situaciones de emergencia, lo que permite una respuesta más rápida.
- **Mejora de la eficiencia.** Automatiza tareas de seguridad, lo que reduce la necesidad de supervisión constante.
- **Mayor conciencia situacional.** Proporciona información en tiempo real sobre la seguridad, lo que facilita la toma de decisiones informadas.
- **Disuasión.** La presencia visible de sistemas de seguridad inteligente puede disuadir a posibles delincuentes.

Seguridad en las Plataformas IOT e IIOT

La seguridad en las plataformas IoT [Internet de las Cosas] e IIoT [Internet Industrial de las Cosas] es esencial para proteger dispositivos conectados, datos y sistemas críticos. Aquí tienes un resumen de este tema:

Seguridad en IoT

1. **Identidad y autenticación.** Garantizar que solo dispositivos y usuarios autorizados tengan acceso a la plataforma mediante autenticación sólida y gestión de claves.
2. **Cifrado de datos.** Proteger la información transmitida y almacenada utilizando cifrado robusto para evitar la interceptación y el acceso no autorizado.
3. **Actualizaciones de firmware.** Mantener los dispositivos IoT actualizados con parches de seguridad y actualizaciones de firmware para corregir vulnerabilidades conocidas.

4. **Gestión de acceso.** Limitar los privilegios de acceso y aplicar el principio de "menos privilegios" para reducir el riesgo de ataques.
5. **Monitoreo y detección.** Implementar sistemas de monitoreo continuo y detección de amenazas para identificar actividades anómalas y posibles intrusiones.

Seguridad en IloT (Industrial IoT)

1. **Aislamiento de redes.** Segmentar las redes industriales de las redes empresariales para evitar que las amenazas se propaguen desde una parte de la red a otra.
2. **Protocolos seguros.** Utilizar protocolos de comunicación seguros y estándares industriales para proteger la integridad de los datos y los sistemas.
3. **Resiliencia.** Planificar la continuidad del negocio y la recuperación ante desastres para garantizar la disponibilidad y la resistencia de los sistemas IloT.
4. **Seguridad física.** Proteger físicamente los dispositivos y sistemas IloT de acceso no autorizado o daño.
5. **Cumplimiento normativo.** Cumplir con regulaciones específicas de la industria y las normativas de seguridad cibernética.

Desafíos Comunes

- **Escalabilidad.** La expansión de la infraestructura IoT puede dificultar la gestión y la seguridad de un gran número de dispositivos.
- **Interoperabilidad.** La diversidad de dispositivos y protocolos puede complicar la implementación de medidas de seguridad coherentes.
- **Actualización de dispositivos antiguos.** Mantener dispositivos IoT antiguos actualizados puede ser un desafío, lo que puede dejarlos vulnerables a amenazas.
- **Privacidad de datos.** La recopilación y el manejo de datos personales en aplicaciones IoT plantean preocupaciones de privacidad que deben ser abordadas.

Wearables at Work

Los "wearables" en el entorno laboral, o "Wearables at Work", se refieren a la utilización de dispositivos tecnológicos que se llevan puestos, como relojes inteligentes, dispositivos de seguimiento de actividad física y gafas inteligentes, en el contexto de un lugar de trabajo.

Funcionalidad en el Trabajo

1. **Monitoreo de salud y bienestar.** Los wearables pueden rastrear la salud y el bienestar de los empleados, como la frecuencia cardíaca, el sueño y la actividad física, promoviendo un estilo de vida más saludable.
2. **Seguridad en el trabajo.** Algunos wearables pueden alertar sobre situaciones de peligro o ayudar en la ubicación de trabajadores en entornos potencialmente riesgosos, como la construcción.
3. **Productividad.** Los dispositivos como los relojes inteligentes pueden proporcionar notificaciones y recordatorios, ayudando a los trabajadores a administrar su tiempo y tareas de manera más eficiente.
4. **Formación y capacitación.** Las gafas inteligentes pueden ofrecer instrucciones en tiempo real o mostrar información relevante para la capacitación de los empleados.

Beneficios

- **Mejora de la productividad.** Los wearables pueden aumentar la eficiencia y la productividad al proporcionar acceso rápido a datos y notificaciones.
- **Seguridad mejorada.** En entornos peligrosos, los dispositivos wearables pueden mejorar la seguridad al proporcionar alertas y geolocalización.
- **Bienestar del empleado.** El seguimiento de la salud y el bienestar puede promover una fuerza laboral más saludable y reducir las bajas por enfermedad.
- **Capacitación eficiente.** Facilitan la capacitación y la resolución de problemas al proporcionar información en tiempo real.

Desafíos

- **Privacidad y seguridad de datos.** La recopilación de datos personales y de salud plantea preocupaciones de privacidad y seguridad que deben abordarse.
- **Adopción y aceptación.** No todos los trabajadores pueden sentirse cómodos usando wearables en el trabajo, lo que puede generar resistencia.
- **Costo.** La inversión en dispositivos y su mantenimiento pueden ser un desafío para las empresas.

Implementación de una API para interactuar con una plataforma

La implementación de una API (Interfaz de Programación de Aplicaciones) para interactuar con una plataforma es un proceso clave que permite que aplicaciones y sistemas externos se comuniquen y accedan a funcionalidades y datos de la plataforma. Aquí tienes un resumen de este proceso:

- **Definición de objetivo.** El primer paso es definir claramente los objetivos de la API, determinando qué funcionalidades o datos de la plataforma estarán disponibles para los usuarios externos.
- **Diseño de la API.** Se diseña la estructura de la API, incluyendo los endpoints, métodos de solicitud (GET, POST, PUT, DELETE), y formatos de datos (generalmente JSON o XML).
- **Desarrollo de la API.** Los desarrolladores crean la API, escribiendo el código necesario para que la plataforma responda adecuadamente a las solicitudes externas. Esto implica la creación de controladores, rutas y validaciones de datos.
- **Autenticación y autorización.** Se implementan mecanismos de autenticación y autorización para garantizar que solo los usuarios autorizados puedan acceder a la API y sus recursos.
- **Documentación.** Se crea documentación detallada que describe cómo usar la API, incluyendo ejemplos de solicitudes y respuestas, para que los desarrolladores externos puedan integrarla fácilmente.
- **Pruebas y depuración.** Se realizan pruebas exhaustivas para asegurarse de que la API funcione correctamente y que los errores se manejen adecuadamente.
- **Implementación en la plataforma.** La API se implementa en la plataforma, generalmente en un servidor web o en la nube, y se asegura su disponibilidad y escalabilidad.
- **Monitoreo y mantenimiento.** Se establece un sistema de monitoreo continuo para supervisar el rendimiento y la seguridad de la API. Además, se realizan actualizaciones y mantenimiento según sea necesario.
- **Lanzamiento y promoción.** La API se lanza para su uso público o privado, y se promociona entre la comunidad de desarrolladores y usuarios interesados.
- **Soporte técnico.** Se proporciona soporte técnico para ayudar a los usuarios externos con preguntas o problemas relacionados con la API.



Valoremos lo aprendido

Analicemos y reflexionemos sobre la seguridad, la responsabilidad en el uso y manejo de las computadoras, el cuidado al medio ambiente y la Madre Tierra

- ¿Cuáles son los componentes de una computadora?
- ¿Qué es software?
- ¿Qué es la memoria RAM en un ordenador?
- ¿Qué es seguridad en informática?
- ¿Cuál fue la primera computadora electrónica y cuándo se construyó?
- ¿Qué proporciona la fuente de alimentación?

Bibliografía

- Alonso Jiménez, José A. [2010], “Temas de lógica informática”. Universidad de Sevilla. Sevilla.
- Àlvarez Canal, Jordi y Vilaplana Pastó, Josep [2010], “Introducción a la algorítmica”. UOC. Cataluña.
- Beynon-Davies, Paul [2018], “Sistemas de información: Introducción a la informática en las organizaciones”. Reverté. Madrid.
- Cerquera Rojas, Yamil Armando [2002], “Algorítmica para programación”. Autoedición. Ecuador.
- Hurtado Alegría, Julio Ariel et al [2014], “Temas de Teoría de la Computación”. LATIN. México.
- Navarro, Gonzalo [2016], “Teoría de la computación: lenguajes formales, computabilidad y complejidad”. Universidad de Chile. Santiago.
- Ortega Arjona, Jorge L. [2008], “Breves Notas sobre Teoría de la Computación”. UNAM. México D. F.

WEBGRAFÍA

- <https://www.youtube.com/watch?v=RVGIXfC4Xeg>
- Passei Directo. [Dakota del Norte]. Sistemas Operativos - Estructura. Recuperado de <https://www.passeidireto.com/arquivo/124336607/sistemas-operativos-estructura>
- Quindel. [sf]. Quindel. Recuperado de <https://quindel.com/es-es>
- TecnoBits. [sf]. Cómo funciona el sistema operativo Windows. Recuperado de <https://tecnobits.com/como-funciona-el-sistema-operativo-windows/>



ESTADO PLURINACIONAL DE
BOLIVIA

MINISTERIO
DE EDUCACIÓN

VICEMINISTERIO DE
EDUCACIÓN ALTERNATIVA Y
ESPECIAL



minedu.gob.bo



[@minedubol](https://www.facebook.com/minedubol)



[minedu_bol](https://www.youtube.com/minedu_bol)

Av. Arce No. 2147 - Teléfonos: (591 -2) 2442144 - 2681200
La Paz - Bolivia