Examen com 2

Ejercicio 1 Justificar todas las respuestas

A - Un sistema de archivos es lo mismo que un sistema operativo

[] Verdadero

[] Falso

B - Si ejecuto estos comandos en el siguiente orden:

mkdir -p dir1/dir2/dir3

cd dir1/dir2

cd ..

touch ABC

Elegir la rta correcta:

□se crea un archivo llamado ABC en dir3

□se crea un archivo llamado ABC en dir2

□se crea un archivo llamado ABC en dir1

□una mas que no me acuerdo

C - Los inodos guardan la información sobre el directorio donde se encuentra un archivo

[] Verdadero

[] Falso

D - drwxr-xr-- 2 aaa bbb blablabla mi\_file

Justificar verdadero o falso:

□Los usuarios del grupo aaa pueden leer

□mi\_file tiene soft links

□mi\_file es un directorio

□en octal, los permisos serian 765?

Ejercicio 2

A - Qué es el kernel o núcleo del sistema operativo? Describa funciones y características

B - Qué es un compilador? Que es un intérprete? Cuál es la diferencia entre leguaje compilado e interpretado?

C - Explique que es un lenguaje de bajo nivel

D - Qué es un hipervisor? De que tipos hay? Dar diferencias

EJERCICIO 1

**A) Un sistema de archivos es lo mismo que un sistema operativo**

No, un sistema de archivos no es lo mismo que un sistema operativo. Son conceptos diferentes pero relacionados en el ámbito de la informática.

Un sistema operativo es un software complejo que administra los recursos de hardware y proporciona servicios y funciones básicas para que otros programas o aplicaciones se ejecuten en un dispositivo de cómputo. Es responsable de administrar la memoria, los procesos, los dispositivos de entrada y salida, la seguridad, entre otras tareas.

Por otro lado, un sistema de archivos es una forma en la que los datos se organizan y se almacenan en un dispositivo de almacenamiento, como un disco duro o una memoria flash. Define la estructura de los archivos y directorios, así como los métodos para acceder, leer y escribir datos en el almacenamiento.

El sistema de archivos es una parte integral del sistema operativo, ya que el sistema operativo utiliza el sistema de archivos para almacenar y organizar los archivos del usuario, así como para realizar operaciones de lectura y escritura en ellos. Sin embargo, un sistema de archivos puede existir sin un sistema operativo, como en el caso de los sistemas de archivos utilizados en dispositivos de almacenamiento externos como memorias USB o tarjetas de memoria.

En resumen, mientras que el sistema operativo proporciona un entorno de ejecución y gestiona los recursos de hardware y software de un dispositivo, el sistema de archivos se encarga de la organización y el acceso a los datos almacenados en dicho dispositivo.

B) Respuesta: □se crea un archivo llamado ABC en dir1

C - Los inodos guardan la información sobre el directorio donde se encuentra un archivo

Sí, los inodos en un sistema de archivos UNIX o similar almacenan información sobre los archivos, incluyendo detalles sobre su ubicación en el directorio. Un inodo es una estructura de datos utilizada por el sistema de archivos para representar un archivo o directorio.

Cada archivo en un sistema de archivos tiene asociado un inodo único que contiene metadatos sobre el archivo, como su tamaño, permisos, propietario, timestamps y punteros a los bloques de datos que componen el archivo. Además, el inodo guarda información sobre el directorio que contiene el archivo, como el número de inodo del directorio padre y el nombre del archivo dentro del directorio.

Cuando se accede a un archivo, el sistema de archivos utiliza la información almacenada en el inodo para localizar los bloques de datos correspondientes y leer o escribir en ellos. En resumen, los inodos proporcionan una forma eficiente de mantener información sobre la estructura jerárquica de directorios y archivos en un sistema de archivos.

**D) drwxr-xr-- 2 aaa bbb blablabla mi\_file Justificar verdadero o falso:**

**□Los usuarios del grupo aaa pueden leer**

**□mi\_file tiene soft links**

**□mi\_file es un directorio**

**□en octal, los permisos serian 765?**

1. Los usuarios del grupo "aaa" pueden leer: Verdadero. La cadena de permisos "r-x" indica que los usuarios del grupo "aaa" tienen permisos de lectura en el directorio.
2. "mi\_file" tiene soft links: Falso. No hay información disponible en la cadena de permisos proporcionada que indique la presencia de soft links. La cadena "blablabla" parece ser un espacio reservado para información adicional o detalles que no se han proporcionado.
3. "mi\_file" es un directorio: Falso. La cadena "mi\_file" se muestra como un nombre de archivo en lugar de un nombre de directorio. Además, en la notación estándar de Linux/Unix, los directorios se indican con la letra "d" como el primer carácter en la cadena de permisos. En este caso, el primer carácter es "2", lo que sugiere que podría ser una indicación de alguna otra cosa, como el número de enlaces duros.
4. En octal, los permisos serían 765: Falso. La cadena de permisos "drwxr-xr--" no se puede convertir directamente a la notación octal sin más información. La cadena completa de permisos tiene nueve caracteres en total, donde los primeros tres caracteres "drw" indican los permisos del propietario del archivo/directorio, los siguientes tres caracteres "xr-" indican los permisos del grupo y los últimos tres caracteres "xr--" indican los permisos para otros usuarios.

Para convertir correctamente a la notación octal, asignamos un valor numérico a cada permiso. "r" representa el valor 4, "w" representa el valor 2 y "x" representa el valor 1. Por lo tanto, la cadena "drwxr-xr--" se traduciría en la notación octal como 751 (4 + 2 + 1 = 7 para el propietario, 4 + 1 = 5 para el grupo y 4 para otros usuarios).

EJERCICIO 2

**A) Qué es el kernel o núcleo del sistema operativo? Describa funciones y características**

El kernel o núcleo de un sistema operativo es la parte central y fundamental que actúa como intermediario entre el hardware y el software de una computadora. Es responsable de administrar los recursos del sistema y proporcionar servicios básicos para que los programas y los usuarios puedan interactuar con el hardware de manera eficiente y segura.

Las funciones y características del kernel pueden variar dependiendo del sistema operativo, pero en general, realizan las siguientes tareas:

1. Administración de memoria: El kernel asigna y administra la memoria disponible en el sistema. Controla el acceso a la memoria por parte de los programas y se encarga de liberar la memoria cuando ya no es necesaria.
2. Gestión del sistema de archivos: Proporciona una interfaz para acceder y manipular los archivos y directorios del sistema. El kernel administra las operaciones de lectura, escritura y eliminación de archivos, así como la organización y estructura de los sistemas de archivos.
3. Administración del procesador: Asigna y administra el tiempo de procesador disponible para los distintos programas en ejecución. Controla la planificación de los procesos, permitiendo que múltiples programas se ejecuten simultáneamente en el sistema.
4. Control de dispositivos: El kernel gestiona la comunicación y el control de los dispositivos de hardware, como impresoras, discos duros, teclados, entre otros. Proporciona una interfaz estandarizada para que los programas puedan interactuar con los dispositivos de manera uniforme.
5. Gestión de interrupciones: Maneja las interrupciones generadas por el hardware y los dispositivos. Cuando ocurre una interrupción, el kernel detiene temporalmente la ejecución de un programa para atender la solicitud o evento que generó la interrupción.
6. Seguridad y protección: El kernel es responsable de garantizar la seguridad y protección del sistema operativo y los datos almacenados en él. Controla el acceso a los recursos del sistema, implementa mecanismos de autenticación y controla los permisos de los usuarios y programas.
7. Comunicación entre procesos: Permite la comunicación y sincronización entre procesos o programas en ejecución. Proporciona mecanismos para el intercambio de datos y la coordinación entre los diferentes procesos, como colas, señales o memoria compartida.
8. Control de errores: El kernel gestiona los errores y excepciones que pueden ocurrir en el sistema operativo y los programas. Proporciona mecanismos para detectar y manejar errores, evitando que un error en una aplicación afecte la estabilidad de todo el sistema.

En resumen, el kernel del sistema operativo es el componente esencial que permite la interacción entre el hardware y el software de una computadora. Proporciona servicios básicos, administra los recursos del sistema y garantiza su estabilidad, seguridad y eficiencia.

**B) Qué es un compilador? Que es un intérprete? Cuál es la diferencia entre leguaje compilado e interpretado?**

Un compilador es un programa que traduce el código fuente de un programa escrito en un lenguaje de programación de alto nivel a un código objeto o código máquina específico de una arquitectura de computadora en particular. El compilador realiza un análisis completo del código fuente y lo traduce a un formato ejecutable antes de que se ejecute el programa. En resumen, el compilador toma el código fuente completo y lo traduce a instrucciones de bajo nivel que la computadora puede entender directamente.

Por otro lado, un intérprete es un programa que lee y ejecuta instrucciones escritas en un lenguaje de programación de alto nivel directamente, sin la necesidad de traducir el código completo de antemano. En lugar de traducir todo el código de una vez, el intérprete lee y ejecuta las instrucciones línea por línea o en bloques más pequeños, traduciéndolas y ejecutándolas en tiempo real.

La principal diferencia entre un lenguaje compilado y un lenguaje interpretado radica en cómo se procesa el código fuente. En un lenguaje compilado, el código fuente se traduce completamente a un código objeto antes de ejecutarse, lo que generalmente resulta en una mayor eficiencia y rendimiento durante la ejecución del programa. Una vez que el código ha sido compilado, el programa puede ejecutarse directamente y no requiere la intervención del compilador nuevamente, a menos que se realicen cambios en el código fuente y se requiera una nueva compilación.

Por otro lado, en un lenguaje interpretado, el código fuente se traduce y se ejecuta en tiempo real. Cada vez que el programa se ejecuta, el intérprete lee y traduce el código línea por línea o en bloques más pequeños, lo que puede llevar a una menor eficiencia y rendimiento en comparación con un lenguaje compilado. Sin embargo, los lenguajes interpretados suelen ofrecer una mayor flexibilidad y facilidad para depurar y modificar el código en tiempo de ejecución, ya que no es necesario volver a compilar todo el programa.

En resumen, la diferencia fundamental entre un lenguaje compilado y un lenguaje interpretado radica en cómo se traduce y ejecuta el código fuente. Un compilador traduce todo el código de antemano y luego se ejecuta el programa compilado, mientras que un intérprete traduce y ejecuta el código línea por línea o en bloques más pequeños en tiempo real.

**C - Explique que es un lenguaje de bajo nivel**

Un lenguaje de bajo nivel es un tipo de lenguaje de programación que está más cerca del lenguaje de máquina o lenguaje ensamblador, en contraste con los lenguajes de alto nivel que están más orientados al ser humano. Los lenguajes de bajo nivel brindan un nivel de abstracción muy bajo, lo que significa que se corresponden de manera más directa con las instrucciones y operaciones que una computadora puede ejecutar.

En un lenguaje de bajo nivel, las instrucciones se expresan en un formato que es comprensible para el hardware de la computadora. Estos lenguajes suelen tener una correspondencia uno a uno con las instrucciones en lenguaje de máquina, que es el conjunto de instrucciones que una CPU (Unidad Central de Procesamiento) puede ejecutar directamente. Las instrucciones en lenguajes de bajo nivel suelen ser simples y están estrechamente relacionadas con las operaciones de bajo nivel que realiza el hardware, como el movimiento de datos en la memoria, las operaciones aritméticas básicas y el control de flujo.

Algunos ejemplos de lenguajes de bajo nivel incluyen el lenguaje ensamblador y el lenguaje de máquina. Estos lenguajes brindan un mayor control sobre el hardware y permiten una programación más precisa y eficiente, pero también son más difíciles de aprender y utilizar en comparación con los lenguajes de alto nivel. Los lenguajes de bajo nivel son comúnmente utilizados en situaciones donde se requiere un control extremadamente fino del hardware, como el desarrollo de controladores de dispositivos, sistemas operativos y aplicaciones embebidas.

En resumen, un lenguaje de bajo nivel es un lenguaje de programación que se encuentra más cerca del lenguaje de máquina y permite un mayor control sobre el hardware de una computadora, pero a costa de una mayor complejidad y menos abstracción que los lenguajes de alto nivel.

**D - Qué es un hipervisor? De qué tipos hay? Dar diferencias**

Un hipervisor, también conocido como monitor de máquina virtual o VMM (Virtual Machine Monitor), es un software o firmware que permite la creación y ejecución de múltiples máquinas virtuales (VM) en un solo sistema físico. El hipervisor se sitúa entre el hardware subyacente y las máquinas virtuales, y se encarga de gestionar y asignar los recursos del sistema, como la CPU, la memoria, el almacenamiento y los dispositivos de entrada y salida, entre las diferentes máquinas virtuales.

Existen dos tipos principales de hipervisores:

1. Tipo 1 o hipervisor nativo: También conocido como "bare-metal", se instala directamente en el hardware subyacente del sistema físico. Actúa como una capa de abstracción entre el hardware y las máquinas virtuales. Los hipervisores tipo 1 tienen un rendimiento más eficiente y directo, ya que no dependen de un sistema operativo anfitrión. Ejemplos populares de hipervisores tipo 1 son VMware ESXi, Citrix XenServer y Microsoft Hyper-V.

Diferencias del hipervisor tipo 1:

* Se instala directamente en el hardware subyacente.
* No depende de un sistema operativo anfitrión.
* Ofrece un rendimiento más eficiente.
* Permite una mayor consolidación de recursos y escalabilidad.

1. Tipo 2 o hipervisor alojado: Se instala como una aplicación dentro de un sistema operativo anfitrión. El sistema operativo anfitrión debe estar instalado previamente en la máquina física y el hipervisor se ejecuta como un proceso dentro de él. Los hipervisores tipo 2 son más fáciles de instalar y configurar, pero pueden tener un rendimiento ligeramente inferior debido a la dependencia del sistema operativo anfitrión. Ejemplos de hipervisores tipo 2 incluyen VMware Workstation, Oracle VirtualBox y Microsoft Virtual PC.

Diferencias del hipervisor tipo 2:

* Se instala como una aplicación dentro de un sistema operativo anfitrión.
* Depende del sistema operativo anfitrión.
* Puede tener un rendimiento ligeramente inferior.
* Es más fácil de instalar y configurar.

En resumen, los hipervisores tipo 1 se instalan directamente en el hardware y no dependen de un sistema operativo anfitrión, mientras que los hipervisores tipo 2 se ejecutan como aplicaciones dentro de un sistema operativo anfitrión. Ambos tipos tienen sus ventajas y desventajas, y la elección depende de los requisitos y las necesidades específicas del entorno virtualizado.