Entrega 1

Integrantes: Juan Ignacio Farizano, Natalia Mellino.

Ejercicio 1

1. **Abstracción:** los sockets nos proveen un mecanismo mediante el cual dos procesos pueden intercambiar información. El Sistema Operativo nos provee este mecanismo sin que nosotros tengamos la necesidad de saber cómo funciona a nivel de hardware, es decir, nos abstrae de la implementación.

2. **Abstracción:** el SO le provee al proceso una herramienta que le permite almacenar o acceder a la información en memoria sin tener que saber en qué parte de la memoria física se encuentra ese dato.

   Aislamiento: a cada proceso se le asigna un bloque de memoria distinto de los que se le asigna a los otros procesos. De esta forma, no pueden interferirse entre sí.

   Administración: se relaciona con la administración por lo que es la memoria virtual en sí: es una técnica de gestión de memoria que permite disponer de más memoria de la que realmente hay físicamente.

3. **Abstracción:** el SO no nos muestra detalles sobre cómo hace la asignación de memoria que nos da, simplemente nos provee el mecanismo y nosotros como programadores no tenemos que preocuparnos por cómo lo hace. **Administración:** la memoria es un recurso, por lo tanto el Sistema Operativo tiene la responsabilidad de administrarla y asignarla de forma correcta.

4. **Administración:** al tener señales que nos permiten notificar de un evento a un proceso, el SO puede interrumpir el flujo de ejecución normal para entregar la señal, a la cual el proceso responde mediante un 'signal handler' (si es que este está definido) o en caso contrario, seguirá el comportamiento por defecto.

5. **Aislamiento:** permite dar al usuario la idea de que él es el único que está trabajando en la PC con el mecanismo de gestión de usuarios.

   Administración: debe designar de forma correcta a qué cosas tienen acceso determinados usuarios. No todos los usuarios pueden estar autorizados para ver todo.

6. **Administración:** utilizamos los bloqueos para garantizar que el acceso a un recurso sea de forma segura cuando sea necesario, por ejemplo que en la escritura dos procesos no estén escribiendo al mismo tiempo.

7. **Abstracción:** permite abstraer al usuario de los detalles de cómo realmente están almacenados los archivos, brindando una interfaz más abstracta y simple que permite ver cómo está organizada la estructura de los archivos.

Ejercicio 2

- Desde el punto de vista de un desarrollador, es importante estudiar sistemas operativos ya que comprender su funcionamiento básico y saber diseñar algoritmos y procesos que se ajusten mejor al SO en el que se va a ejecutar el programa, puede resultar en un factor decisivo en el producto final.
- Como un administrador de sistemas, es tarea diaria enfrentarse a situaciones de bajo rendimiento, conflictos entre aplicaciones, demoras en la ejecución, etc. Para poder hacerle frente a estos problemas de manera óptima es muy importante entender lo que ocurre detrás de escena. Un área de interés son los sistemas de archivos, donde se plantean algunas de las siguientes cuestiones:
  - ¿Cómo comparar las virtudes y desventajas de tantos sistemas existentes?
  - ¿Por qué puede resultar conveniente mezclar distintos sistemas existentes en el mismo servidor?
  - ¿Cómo evitar la corrupción o pérdida de información?
  - ¿Cómo recuperar información de un disco dañado?

Ejercicio 3

Un algoritmo muy conocido y visto en Sistemas Operativos I es el algoritmo propuesto por Dijkstra para evitar el bloqueo mutuo. Este puede ser explicado usando la analogía de un banco, donde los clientes son los procesos y el dinero los recursos, mientras que el banquero es el sistema operativo (por este motivo el algoritmo es conocido como 'Algoritmo del Banquero'): cada cliente dispone de un crédito que corresponde al máximo de dinero que el banquero le prestará. El cliente irá solicitando fondos de ese crédito conforme los necesite. Los recursos del banquero son limitados, por lo tanto no puede darle a todos los clientes el máximo de crédito al mismo tiempo. Entonces, la solución es que el banquero gestione sus fondos de manera que todos los clientes puedan llegar al máximo en algún momento pero no al mismo tiempo. Se dice que mantiene un **estado seguro** si en el sistema existe una secuencia de asignaciones y liberaciones de recursos que permita a todos los clientes alcanzar en algún momento sus necesidades máximas de recursos, en este caso, el dinero. Este algoritmo se puede explicar muy fácilmente llevándolo a un caso de la vida real como hicimos recién, sin embargo, en la práctica no es muy aplicable ya que el algoritmo necesita conocer los recursos de cada proceso de antemano y la mayoría de las veces esto no es posible.

Ejercicio 4

1. Nachos corre como un único proceso Unix, donde en cambio sistemas operativos reales corren directamente sobre el hardware.
2. Nachos simula una máquina que aproxima la arquitectura MIPS.

3. La arquitectura MIPS sobre la que Nachos está basado es de tipo RISC. Contiene 40 **registros** numerados del 0 al 39 entre los cuales se incluye un puntero de pila, un registro doble para resultados de multiplicaciones, un contador de programas, entre otros. Contiene también una **memoria física** organizada, por defecto, en 31 páginas de 128 bytes. Además la máquina de Nachos soporta una **memoria virtual** a través de un TLB (Translation Lookaside Buffer) ó una única tabla de paginación lineal.