PRÁCTICO 2

Objetivos

- Comprender las estructuras y los componentes de un sistema operativo.
- Ejercitar el desarrollo de programas que usen llamadas al sistema.

Ejercicio 1 (básico)

Mencione los componentes principales de un sistema operativo.

Ejercicio 2 (básico)

Mencione las cinco actividades principales de un sistema operativo en lo que concierne a la gestión de procesos.

Ejercicio 3 (básico)

Mencione las tres actividades principales de un sistema operativo en lo que concierne a la gestión de la memoria.

Ejercicio 4 (básico)

Mencione las tres actividades principales de un sistema operativo en lo que concierne a la gestión del almacenamiento secundario.

Ejercicio 5 (básico)

Describa los 5 servicios principales que debe brindar un sistema operativo.

Ejercicio 6 (básico)

¿Para qué sirven las llamadas al sistema (*system calls*)?

Ejercicio 7 (básico)

¿Quiénes invocan los llamados al sistema?

Ejercicio 8 (medio)

Describa las 3 formas utilizadas para pasar parámetros en los llamados al sistema.

Ejercicio 9 (básico)

Describa las 5 clasificaciones básicas de los llamados al sistema.

Ejercicio 10 (en clase)

Utilizando llamadas al sistema, implementar un programa que lea un archivo, cuyo nombre recibe como parámetro, y lo imprima en el dispositivo *impresora*. Dicho dispositivo cuenta con un buffer de 512 bytes. Cuando la impresora tiene el buffer vacío, se lo indica al sistema operativo, el cual genera el evento *impresora_libre*.

El sistema operativo ofrece las siguientes llamadas:

- **Fin**
  Indica final exitoso de la ejecución.
- **Abort**
  Indica final anormal de la ejecución.
- **Abrir_archivo(nombre_archivo): Entero**

  Cerrar_archivo(nombre_archivo): Entero

  Abren y cierran un archivo dado su nombre. Devuelven –1 en caso de error.
- **Leer(nombre_archivo, buffer, n): Entero**

  Lee *n* caracteres de un archivo identificado por su nombre, almacenándolos en el array *buffer*, y devuelve la cantidad de bytes leídos, o –1 en caso de error. En caso de que se haya alcanzado el fin del archivo, devuelve 0.
- **Obtener_recurso(nombre_recurso)**
  Liberar_recurso(nombre_recurso)
  Obtienen y liberan un recurso (que puede ser un dispositivo, memoria, etc.) dado su nombre. Se bloquean hasta que el recurso sea obtenido o liberado.
- **Escribir_dispositivo(nombre_dispositivo, buffer, n): Entero**

  Escribe *n* bytes del buffer el dispositivo indicado por el nombre, y devuelve la cantidad de bytes que pudo escribir, o –1 en caso de error.
- **Esperar_evento(nombre_evento)**
  Bloquea al proceso hasta que suceda el evento esperado. Retornan el control inmediatamente si el evento ya ocurre en el momento de invocarlo.

Nota: El sistema es multiprogramado.

Ejercicio 11 (básico)

Comente 2 formas de estructurar un sistema operativo en la fase de diseño.

Ejercicio 12 (medio)

Comente ventajas y desventajas de los distintos enfoques en el diseño del sistema operativo.

Ejercicio 13 (medio)

Describa el diseño de un sistema operativo con el enfoque de micronúcleo (*microkernel*).