Laboratorio #5

* Structs es un tipo de data que permite combinar diferentes tipos de datos. Son utilizados para tener registros.
  + struct [structure tag] {

definición de miembro;   
} [variables de estructura];

* Las directivas son utilizadas para hacer source programs fáciles de cambiar y compilar en varios ambientes de ejecución.
* & es un puntero a una dirección de memoria y provee dicha dirección  
  \* Hace un request al valor al que apunta la dirección de memoria.
* APT y dpkg son dos instaladores de paquetes utilizados en Ubuntu y sistemas basados en Debian. APT utiliza dpkg en el backend, puede descargar de manera remota, mientras que dpkg únicamente de manera local.
* Sched.h declara funciones para manipular y manejar fácilmente la calendarización de ejecución de procesos.
* Las definiciones definen las implementaciones de todas las políticas de calendarización.
* Un task hace referencia a una unidad de ejecución que puede compartir recursos con otros tasks.
* task\_struct es un descriptor de proceso. Contiene todo lo que el kernel necesitaría saber sobre un proceso. En Windows, EPROCESS es un equivalente a task\_struct.
* contiene los parámetros requeridos para la implementación de cada política de scheduling.
* rt\_policy decide si una política de scheduling pertenece a una clase en tiempo real o no. Es un macro utilizado para generar código muy optimizado que no sería posible hacer en una instrucción de Branch/jump.
* EDF calendariza las tareas dependiendo del orden en el que deben completarse.
* EDF toma precedencia por la prioridad de deadline. Luego RT sobre CFS, porque RT toma precedencia sobre el default, el cual es CFS.
* casio\_task tiene el nodo del árbol red-black. Luego el deadline absoluto. Donde se encuentra el head de la lista, por ultimo un puntero de una estructura de task.
* Casio\_rq contiene una linked list que apunta hacia una task p
* Atomict es una variable int con un set de operaciones que garantiza ser atómica sin necesidad de un bloqueo explícito
  + RMW lee una ubicación de memoria y escribe un nuevo valor simultáneamente.
  + MMIO es un método de reconocimiento I/O entre CPU y los periféricos conectados a la máquina.
* .next indica la dirección del siguiente task calendarizado.
* Funcionamiento
  + Se hace la llamada al tipo de calendarización
  + Se toma el proceso y se ingresa a la lista junto con sus valores.
  + Se crea el task con los valores encesarios para la política de calendarización
  + Se agrega el task al scheduler
  + Se revisa lo que se debe hacer con el task ya sea ejecutarla, agregarla o removerla del árbol
  + Por último, se ordenan y ejecutan las task con prioridad del deadline absoluto.
* Un task preemptea a otro que este en ejecución cuando su deadline absoluto es menor al actual.
* Diferencias:
  + After empieza con prioridad 6.
  + El orden de los tasks es distinto.
  + El tiempo de ejecución de los tasks es menor.