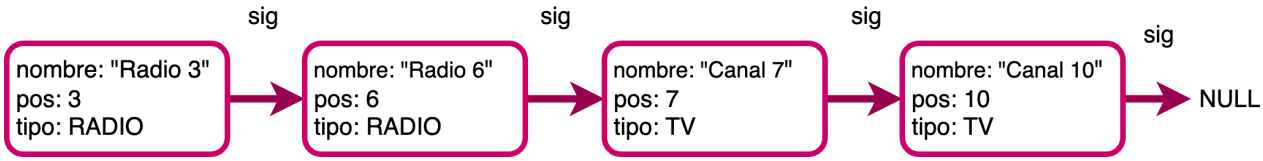
En este ejercicio vamos a desarrollar un módulo simplificado para la gestión de la lista de canales favoritos de un televisor. Para ello vamos a utilizar las siguientes definiciones de tipos, que se encuentran en el fichero canales.h.

```
#define MAX_NOMBRE 25

enum Tipo {TV, RADIO};

struct T_Canal{
    char nombre[MAX_NOMBRE];
    int pos;            //posicion que ocupa el canal en la lista
    enum Tipo tipo;     //canal de radio o television
    struct T_Canal*  sig;   //referencia al siguiente canal
};
```

El tipo enumerado `enum Tipo` nos permite especificar si el canal es un cana de radio o de televisión. El tipo registro `struct T_Canal` permite almacenar la información necesaria de un canal que consiste en el nombre, la posición que ocupa el canal en la lista de favoritos y el tipo de canal.
La lista de canales favoritos se va a implementar utilizando una **lista enlazada de nodos `struct T_Canal` que estará ordenada de manera creciente por el campo `pos` de los canales** en la lista. La lista de canales favoritos puede estar vacía (en ese caso no habrá ningún canal) o puede contener varios canales cuyas posiciones no tienen porqué ser consecutivas (pero si diferentes). En la siguiente imagen se muestra un ejemplo de lista de canales con 4 canales. Observa que el campo siguiente de cada canal tiene que apuntar al siguiente canal, excepto en el caso del último canal cuyo campo siguiente apunta a `NULL`.



Entre las operaciones que ofrece este módulo está la inserción de un canal en la lista en una posición dada. La inserción debe generar una que sigue estándo ordenada de forma creciente por el campo pos. Si se quiere insertar un canal en una posición que ya está ocupada en la lista, el nuevo canal se inserta en la posición indicada. El antiguo canal de esa posición (y todos los canales consecutivos con posiciones mayores) se desplazarán una posición. Por ejemplo, si en la lista anterior se quiere insertar en la posición 6 el canal de TV “Canal 6”, el canal “Radio 6” pasará a la posicón 7 y el canal “Canal 7” pasará a la posición 8, tal y como se muestra en la siguiente figura:

Para la realización del ejercicio se proporciona el fichero de cabecera (canales.h) y un fichero para probar la implementación (principal.c) Además, al final de este documento tienes la salida esperada por consola y un anexo con las funciones para lectura/escritura en ficheros y manipulación de cadenas de texto.

Se pide implementar en el fichero canales.c las siguientes funciones:

```
/**
 * @brief Crea un nuevo canal.
 *
 * Esta función se encarga de inicializar la lista de canales.
 *
 * @param canales Doble puntero a una estructura T_Canal donde se almacenarán los
canales en un futuro.
 */
void crear(struct T_Canal** canales);


/**
 * @brief Destruye la lista de canales.
 *
 * Esta función libera la memoria asignada para cada canal, dejando el @p canales
apuntando a NULL.
 *
 * @param canales Dirección de memoria del puntero a la cabeza de la lista.
 */
void destruir(struct T_Canal** canales);


/**
 * @brief Muestra la información de los canales.
 *
 * Esta función toma un puntero a una estructura T_Canal y muestra la información
 * contenida en los canales.
 *
 * @param canales Puntero a una estructura T_Canal que contiene la información de los
canales.
 */
void mostrar(struct T_Canal*  canales);


/**
 * @brief Inserta un nuevo canal en la lista de canales.
 *
 * Esta función inserta un nuevo canal en la posición especificada dentro de la lista
de canales. Si en la lista de canales ya existe un canal en dicha
posicion, el nuevo se insertar en la posición indicada y el canal antiguo (y los
consecutivos en posiciones mayores) se desplazaran una posición.
 *
 * @param canales Dirección de memoria del puntero a la cabeza de la lista.
 * @param nombreCanal Nombre del canal a insertar.
 * @param posCanal Posición en la que se debe insertar el nuevo canal.
 * @param tipoCanal Tipo del canal a insertar.
 */
void insertar(struct T_Canal* *canales, char *nombreCanal, int posCanal, enum Tipo
tipoCanal);
```

```c
/**
 * @brief Elimina un canal de la lista de canales.
 *
 * Esta función busca un canal por su nombre en la lista de canales proporcionada
 * y lo elimina si es encontrado.
 *
 * @param canales Dirección de memoria del puntero a la cabeza de la lista.
 * @param nombreCanal Nombre del canal a eliminar.
 * @return int Retorna 0 si el canal fue eliminado exitosamente, o un valor negativo si
ocurrió un error.
 */
int eliminar(struct T_Canal** canales, char *nombreCanal);


/**
 * @brief Extrae un canal de radio de la lista de canales de origen.
 *
 * @param origen Dirección de memoria dónde está el puntero a la cabeza de la lista.
 * @return Dirección de memoria de la lista de solo canales de radio extraída. Debes
mover, no pedir nueva memoria para maxima nota.
 */
struct T_Canal* extraerRadio(struct T_Canal**origen);
```