```
/**
  * @file imagenES.h
  * @brief Fichero cabecera para la E/S de imágenes
  *
  * Permite la E/S de archivos de tipo PGM,PPM
  *
  */

#ifndef _IMAGEN_ES_H_
#define _IMAGEN_ES_H_

/**
  * @brief Tipo de imagen
  *
  * Declara una serie de constantes para representar los distintos tipos
  * de imágenes que se pueden manejar.
  *
  * @see LeerTipoImagen
  */
enum TipoImagen {IMG_DESCONOCIDO, IMG_PGM, IMG_PPM};

/**
  * @brief Devuelve el tipo de imagen del archivo
  *
  * @param nombre indica el archivo de disco que consultar
  * @return Devuelve el tipo de la imagen en el archivo
  *
  * @see TipoImagen
  */
TipoImagen LeerTipoImagen (const char *nombre);

/**
  * @brief Lee una imagen de tipo PPM
  *
  * @param nombre archivo a leer
  * @param filas Parámetro de salida con las filas de la imagen.
  * @param columnas Parámetro de salida con las columnas de la imagen.
  * @return puntero a una nueva zona de memoria que contiene @a filas x @a columnas x 3
  * bytes que corresponden a los colores de todos los píxeles en formato
  * RGB (desde la esquina superior izqda a la inferior drcha). En caso de que no
  * no se pueda leer, se devuelve cero. (0).
  * @post En caso de éxito, el puntero apunta a una zona de memoria reservada en
  * memoria dinámica. Será el usuario el responsable de liberarla.
  */
unsigned char *LeerImagenPPM (const char *nombre, int& fils, int& cols);

/**
  * @brief Escribe una imagen de tipo PPM
  *
  * @param nombre archivo a escribir
  * @param datos punteros a los @a f x @a c x 3 bytes que corresponden a los valores
  *    de los píxeles de la imagen en formato RGB.
  * @param f filas de la imagen
  * @param c columnas de la imagen
  * @return si ha tenido éxito en la escritura.
  */
bool EscribirImagenPPM (const char *nombre, const unsigned char *datos,
                        const int fils, const int cols);

/**
  * @brief Lee una imagen de tipo PGM
  *
  * @param nombre archivo a leer
  * @param filas Parámetro de salida con las filas de la imagen.
  * @param columnas Parámetro de salida con las columnas de la imagen.
  * @return puntero a una nueva zona de memoria que contiene @a filas x @a columnas
  * bytes que corresponden a los grises de todos los píxeles
  * (desde la esquina superior izqda a la inferior drcha). En caso de que no
  * no se pueda leer, se devuelve cero. (0).
  * @post En caso de éxito, el puntero apunta a una zona de memoria reservada en
  * memoria dinámica. Será el usuario el responsable de liberarla.
  */
```

```
unsigned char *LeerImagenPGM (const char *nombre, int& fils, int& cols);

/**
  * @brief Escribe una imagen de tipo PGM
  *
  * @param nombre archivo a escribir
  * @param datos punteros a los @a f x @a c bytes que corresponden a los valores
  *    de los píxeles de la imagen de grises.
  * @param f filas de la imagen
  * @param c columnas de la imagen
  * @return si ha tenido éxito en la escritura.
  */
bool EscribirImagenPGM (const char *nombre, const unsigned char *datos,
                        const int fils, const int cols);

#endif

/* Fin Fichero: imagenES.h */
```