Este archivo se centra en presentar algunas de las **funciones predefinidas** disponibles en lenguajes como C para trabajar con cadenas (strings). Estas funciones simplifican tareas comunes que, de otra manera, requerirían escribir lógica más compleja recorriendo las cadenas carácter a carácter.

Según la fuente, algunas de las funciones de strings más comunes son:

- **strcpy()**: Se utiliza para **copiar el contenido de una cadena en otra**. La copia es destructiva para la cadena de destino, lo que significa que los caracteres originales en el destino se reemplazan por los de la fuente, incluso si el destino era más largo. La cadena destino se pone como primer argumento y la cadena fuente como segundo. Es crucial asegurarse de que la cadena destino tenga suficiente espacio para alojar la cadena origen. El archivo muestra un ejemplo donde se copia "corta" a "mediana", y luego "larguisima" a "mediana", demostrando el reemplazo del contenido.
- **strcat()**: Se utiliza para **concatenar o agregar una cadena al final de otra**. La función agrega la cadena2 al final de la cadena1. Un ejemplo ilustra cómo concatenar " y " y "Don Jose" a la cadena inicial "Don Pepito".
- **strcmp()**: Se usa para **comparar una cadena con otra**. (Nota: Aunque se menciona, el archivo no proporciona detalles específicos sobre su funcionamiento o valor de retorno, solo su propósito).
- **strchr()**: Permite **localizar la primera instancia de un carácter** específico dentro de un string. (Nota: No se proporcionan más detalles).
- **strstr()**: Se utiliza para **localizar la primera ocurrencia de un string dentro de otro**. (Nota: No se proporcionan más detalles).
- **strlen()**: Esta función **devuelve el número de caracteres que tiene la cadena, sin contar el carácter nulo '\0'**. Un ejemplo muestra cómo obtener y mostrar la longitud de una cadena ingresada por el usuario.

Además de las funciones de manipulación de cadenas completas, el archivo también lista y describe funciones para trabajar con caracteres individuales, a menudo utilizadas para **clasificación o conversión**:

- **isalnum(caracter)**: Devuelve un valor "cierto" (cualquier entero distinto de cero) si el caracter es una letra o un dígito, y "falso" (el valor entero 0) en caso contrario.
- **isalpha(caracter)**: Devuelve "cierto" si el caracter es una letra, y "falso" en caso contrario.
- **isblank(caracter)**: Devuelve "cierto" si el caracter es un espacio en blanco o un tabulador.
- **isdigit(caracter)**: Devuelve "cierto" si el caracter es un dígito, y "falso" en caso contrario.
- **isspace(caracter)**: Devuelve "cierto" si el caracter es un espacio en blanco, un salto de línea, un retorno de carro, un tabulador, etc., y "falso" en caso contrario.
- **islower(caracter)**: Devuelve "cierto" si el caracter es una letra minúscula, y "falso" en caso contrario.
- **isupper(caracter)**: Devuelve "cierto" si el caracter es una letra mayúscula, y "falso" en caso contrario.
- **toupper(caracter)**: Devuelve la letra mayúscula asociada al caracter, si existe; si no, devuelve el mismo caracter.
- **tolower(caracter)**: Devuelve la letra minúscula asociada al caracter, si existe; si no, devuelve el mismo caracter.