# Bibliotecas

Las bibliotecas, en C, son conjuntos de funciones, tipos de datos, macros, etc, predefinidas, que pueden ser reutilizadas en múltiples programas. Facilitan la reutilización de código, mejoran la modularidad del programa y promueven prácticas de desarrollo eficientes.

1) Biblioteca estática:
   - Se enlazan directamente con el ejecutable del programa durante el tiempo de compilación.
   - Son copiadas en su totalidad al ejecutable final, lo que puede aumentar el tamaño del archivo ejecutable.
   - Requieren menos configuración durante la distribución y ejecución del programa.
2) Biblioteca dinámica (compartida):
   - Se enlazan con el ejecutable en tiempo de ejecución (lo realiza el sistema operativo a través de un cargador dinámico)
   - Son compartidas entre múltiples programas, lo que reduce el uso de memoria y el espacio en disco.
   - Permiten actualizaciones y mantenimiento centralizado de bibliotecas.

# Bibliotecas estáticas

Para trabajar con este tipo de biblioteca debemos hacer:

1) Crear los archivos objetos: se debe compilar cada uno de los archivos fuentes de la biblioteca, por ejemplo:

```
gcc -c util.c -o util.o
gcc -c operaciones.c -o operaciones.o
```

En este caso la biblioteca consta de dos archivos fuentes que generarán dos archivos objetos “util.o” y “operaciones.o”. Se utiliza la bandera ‘-c’ para compilar (no realiza el enlazado).

2) Crear la biblioteca estática: usaremos el comando **“ar”** para combinar los archivos objeto en una biblioteca estática:

```
ar rcs libmylib.a util.o operaciones.o
```

Por convención el inicio del nombre la biblioteca debe incluir “lib”.

- **r**: Inserta los archivos especificados en la biblioteca. Si la biblioteca no existe, se crea. Si ya existe, los archivos objeto se reemplazan.
- **c**: Crea la biblioteca si no existe, sin mostrar advertencias.
- **s**: Crea un índice en la biblioteca, lo que permite una búsqueda más rápida de los símbolos (funciones y variables).

3) Crear el ejecutable:

a) Opción 1:

I. Compilación de tu programa: compilaremos el archivo fuente principal (main.c), incluyendo la bandera -c para que no se realice el enlazado:

```
gcc -c main.c -o main.o
```

II. Enlazar con la biblioteca estática: enlazaremos todos los archivos incluyendo la biblioteca. Aquí tenemos dos opciones, en la primera usaremos la bandera **-L** para especificar el directorio de la biblioteca y **-lmylib** para indicar la biblioteca estática.

- `gcc -o miPrograma main.o -L/path/to/libs -lmylib`
- `gcc -o miPrograma main.o /path/to/libs/libmylib.a`

“-L/path/to/libs”: Especifica el directorio donde se encuentra la biblioteca estática libmylib.a.

b) Opción 2 – Compilación y enlazado en un solo paso:

- `gcc -o miPrograma main.c -L/path/to/libs -lmylib`
- **`gcc -o miPrograma main.c /path/to/libmylib.a`**

# Bibliotecas dinámica

Para trabajar con este tipo de biblioteca debemos hacer:

1) Crear los archivos objetos: se debe compilar cada uno de los archivos fuentes de la biblioteca, por ejemplo:

```
gcc -fPIC -c util.c -o util.o
gcc -fPIC -c operaciones.c -o operaciones.o
```

En este caso la biblioteca consta de dos archivos fuentes que generarán dos archivos objetos “util.o” y “operaciones.o”. Se utiliza la bandera ‘-c’ para compilar (no realiza el enlazado). La bandera ‘-fPIC’ se utiliza para generar código independiente de la posición, esto significa que el código compilado puede ser cargado en cualquier dirección de memoria sin necesidad de ser modificado.

2) Crear la biblioteca dinámica: usaremos **gcc** con la opción **-shared** para crear la biblioteca dinámica.

- `gcc -shared -o libmylib.so util.o operaciones.o`

Por convención el inicio del nombre la biblioteca debe incluir “lib” al inicio.

3) Crear el ejecutable:

a) Opción 1:

I. Compilación de tu programa: compilaremos el archivo fuente principal (main.c), incluyendo la bandera -c para que no se realice el enlazado:

```
gcc -c main.c -o main.o
```

II. Enlazar con la biblioteca estática: enlazaremos todos los archivos incluyendo la biblioteca. Usaremos la bandera **-L** para especificar el directorio de la biblioteca y **-lmylib** para indicar la biblioteca estática.

- `gcc -o miPrograma main.o -L/path/to/libs -lmylib`
- `gcc -o miPrograma main.o /path/to/libs/libmylib.so`

“-L/path/to/libs”: Especifica el directorio donde se encuentra la biblioteca estática libmylib.so.

b) Opción 2 – Compilación y enlazado en un solo paso:

- `gcc -o miPrograma main.c -L/path/to/libs -lmylib`
- **`gcc -o miPrograma main.c /path/to/libmylib.so`**

# Bibliotecas dinámica - consideraciones

Si ejecutamos un programa con una biblioteca dinámica, probablemente nos encontremos con el siguiente error: **error while loading shared libraries: libutil.so: cannot open shared object file: No such file or directory.** Este problema suele ocurrir cuando el sistema no puede ubicar la biblioteca compartida en los directorios estándar donde busca automáticamente al ejecutar un programa. Algunas soluciones serían:

1) Agregar la ruta de la biblioteca a la variable de entorno LD_LIBRARY_PATH
2) Copiar la biblioteca compartida a un directorio estándar, por ejemplo /usr/lib/, y luego actualizar el cache de bibliotecas compartidas con: `sudo ldconfig`. Este comando actualiza la cache de las bibliotecas compartidas y permite que el sistema operativo las encuentre correctamente al ejecutar programas.
3) Configurar el ejecutable para encontrar la biblioteca:

   Esto se hace agregando -Wl,-rpath=/path/to/libs al momento de compilar el programa:

   `gcc -o miPrograma main.o -L/path/to/libs -lmylib -Wl,-rpath=/path/to/libs`

   **-Wl**,: Esta parte del comando le indica al compilador GCC que pase opciones directamente al enlazador (ld, el enlazador de GNU).

   **-rpath**=/path/to/libs: Esta opción le dice al enlazador (ld) que establezca el camino de búsqueda (rpath) para las bibliotecas compartidas en el directorio especificado /path/to/libs.

**Nota:** con respecto al archivo cabecera (.h) de mi biblioteca es una buena práctica incluirlo dentro del directorio o sub directorio del proyecto.

# Algunas banderas del compilador

-Wall: Este flag indica al compilador GCC que muestre todas las advertencias posibles durante la compilación. Es una buena práctica usarlo para asegurarse de que el código esté lo más libre de errores posible y para mantener la calidad del código.

-L: Este flag se usa para especificar al compilador la ubicación de los directorios donde se encuentran las bibliotecas compartidas o estáticas que se van a enlazar con el programa durante el proceso de enlazado. Por ejemplo, si tienes una biblioteca llamada libmylib.a en /path/to/libs, usarías -L/path/to/libs para que el compilador pueda encontrarla.

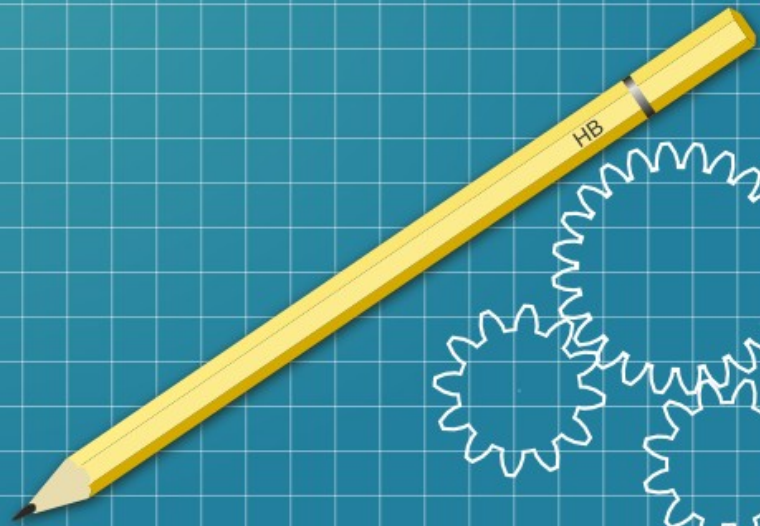
-l: Este flag se usa para especificar al compilador el nombre de la biblioteca que se va a enlazar con el programa durante el proceso de enlazado. Por ejemplo, si quieres enlazar con libmylib.a o libmylib.so, usarías -lmylib. El compilador automáticamente buscará libmylib.a o libmylib.so en las rutas especificadas con -L.

-I: Este flag se usa para especificar al compilador la ubicación de los directorios donde se encuentran los archivos de cabecera (.h) que se incluirán en el proceso de compilación. Por ejemplo, si tienes archivos de cabecera en /path/to/includes, usarías -I/path/to/includes para que el compilador pueda encontrarlos cuando uses #include en tu código.

# Actividad 2

1) Utilizando el módulo **“util”** creado la clase anterior, crear dos carpetas que contenga una libraría estática y otra librería dinámica llamada **libutil**.
2) Compilar en cada carpeta creando el ejecutable miPrograma, usando el **“main.c”** resulto anteriormente, con la librería correspondiente.
3) Utilizar el comando: `nm -a miPrograma` para ambos proyectos. Observar las direcciones de las funciones en ambas bibliotecas
4) Utilizar el comando: `ldd miPrograma` (para el programa con biblioteca dinámica)
5) Observar el tamaño que ocupan los dos archivos
6) Realizar comentarios para generar documentación por doxygen, sobre la biblioteca util.h
7) Emplear doxygen para generar la documentación correspondiente en formato html.





HB