Archivos en C

- Código fuente (.c)
  - implementación
- Header o cabecera (.h)
  - declaración de variables y funciones (interfaz)
- Object (.o)
  - versión binaria del código (no ejecutable)
- Ejecutable
- Librería estática (.a)
- Librería dinámica (.so)

Archivos en C

- Código fuente (.c)
  - implementación

```c
//file main.c
#include<stdio.h>
int main(void)
{
    printf("Hello World \n");
}
```

```c
//file sum.c
int sum(int a, int b)
{
    return a+b;
}
```

Archivos en C

- Header o cabecera (.h)
  - declaración de variables o funciones (interfaz)

```
//file sum.h
int sum(int a, int b);
```

Archivos de C

- Para utilizar una función externa
  - Primero necesitamos declarar la función, mediante un include

```
    //file main.c
#include<stdio.h>
#include "sum.h"
int main(void)
{
    printf("Hello␣World␣\n");
    printf("sum:␣%d␣\n", sum(2,2));

}
```

Compliar en C

- Preproceso
  - Se hace antes de la compilación
  - Copia el contenido de <stdio.h> y de "sum.h" en la locación donde se encuentra el include

```
    //file main.c
int printf(const char *format, ...);
...
int sum(int a, int b);

int main(void)
{
    printf("Hello World \n");
    printf("sum: %d \n", sum(2,2));

}
```

Compliar en C

- A partir de ficheros de código fuente se generan ficheros object (binario)

```
>> gcc -c main.c sum.c
```

- Salida: main.o, sum.o

Compliar en C

- Linker
  - Los archivos object (.o) no son directamente ejecutables
  - Hay que añadir especificaciones para las librerías/funciones externas

```
>> gcc main.o sum.o -o myprog
>> ./myprog
```

- Salida: myprog

Recibir parámetros por línea de comandos (usar strcmp())

```
    //file main.c
#include<stdio.h>

int main(int argc, char *argv[])
{
    printf("\n-----------------");
    int i;
    //skip argv[0] (the program's name)
    for(i=1; i<argc; i++)
    {
            printf("\n␣%s",argv[i]);
    }
    printf("\n-----------------\n");

}
```

- Librerías estáticas
  - Encapsulan un conjunto de variables y funciones
  - Son cargadas en tiempo de compilación y copiadas en el ejecutable final
  - Se necesita un único fichero final lo cual simplifica su distribución
- Para crear una librería estática:

```
>> ar rcs libname.a file1.o file2.o file3.o
```

- Para utilizar una librería estática, a la hora de generar el ejecutable:

```
>> gcc file4.o file5.o -o myprog ./libname.a
```

- Librerías dinámicas
  - Encapsulan un conjunto de variables y funciones
  - No se incluyen dentro del ejecutable sino que son cargadas en memoria en tiempo de ejecución
  - Deben ser distribuidas junto con el ejecutable
  - Fácil mantención
- Para crear una librería dinámica:

```
>> gcc -c -fPIC file1.c file2.c file3.c
>> gcc -shared -o libname.so file1.o file2.o file3.o
```

- Para utilizar una librería dinámica, a la hora de generar el ejecutable:

```
>> gcc file4.o file5.o -o myprog ./libname.so
```

Bash scripts

- Ficheros con extensión .sh
- chmod +x script.sh

```
gcc -c main.c sum.c
gcc main.o sum.o -o myprog
./myprog
```

Archivos Makefile

- Un programa en C normalmente será compilado utilizando el programa make y un archivo Makefile
  - Simplifica el proceso de creación del ejecutable
  - Asegura que solo el código que ha sido modificado es recompilado

```
default: myprog
myprog: main.o sum.o
	gcc main.o sum.o -o myprog
main.o: main.c
	gcc -c main.c
sum.o: sum.c
	gcc -c sum.c
```