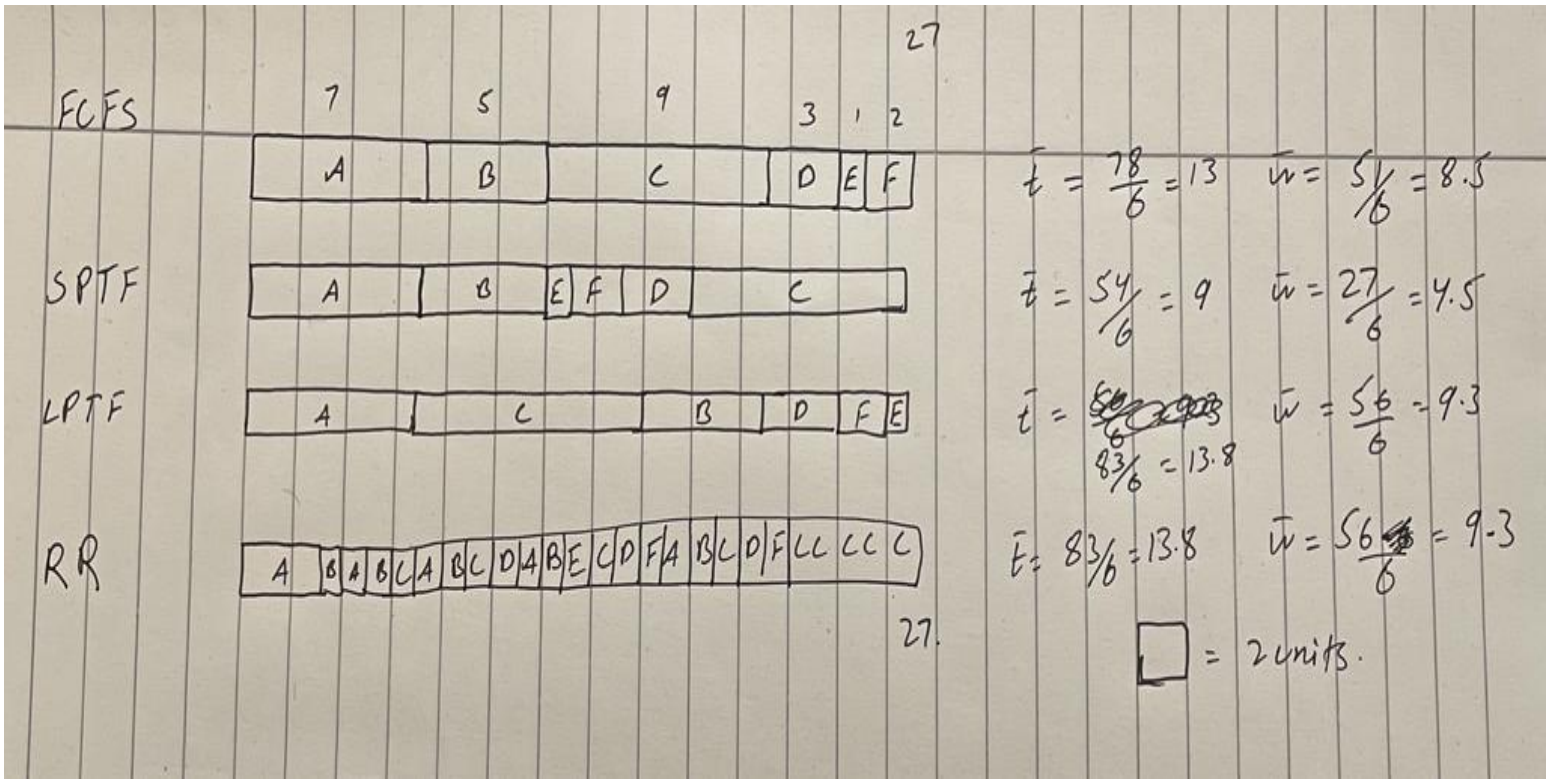


a)

| file | symbol | internal unlinkable symbol | reference of external symbol | weak linkable symbol | strong linkable symbol |
|---|---|---|---|---|---|
| a.c | x | | x | | |
| a.c | y | | | x | |
| a.c | f | x | | | |
| a.c | g | | | | x |
| a.c | h | | | | x |
| b.c | x | | | | x |
| b.c | y | x | | | |
| b.c | z | x | | | |
| b.c | f | | | | x |
| b.c | g | x | | | |

b) What will be printed to the standard output by the following `main()` function? Explain.

b.c: f()
b.c: g()
a.c: h()
a.c: g()
a.c: f()

c) What is name mangling and why do programming languages like C++ use name mangling? Why do I sometimes need to use `extern "C" {}` in C++ header files?

all C style header files (stdio.h, string.h, etc) have their declarations in the extern "C" block.
1. Since C++ supports function overloading, additional information has to be added to function names (called Name mangling) to avoid conflicts in binary code.
2. Function names may not be changed in C as it doesn't support function overloading.
To avoid linking problems, C++ supports the extern "C" block. C++ compiler makes sure that names inside the extern "C" block are not changed.