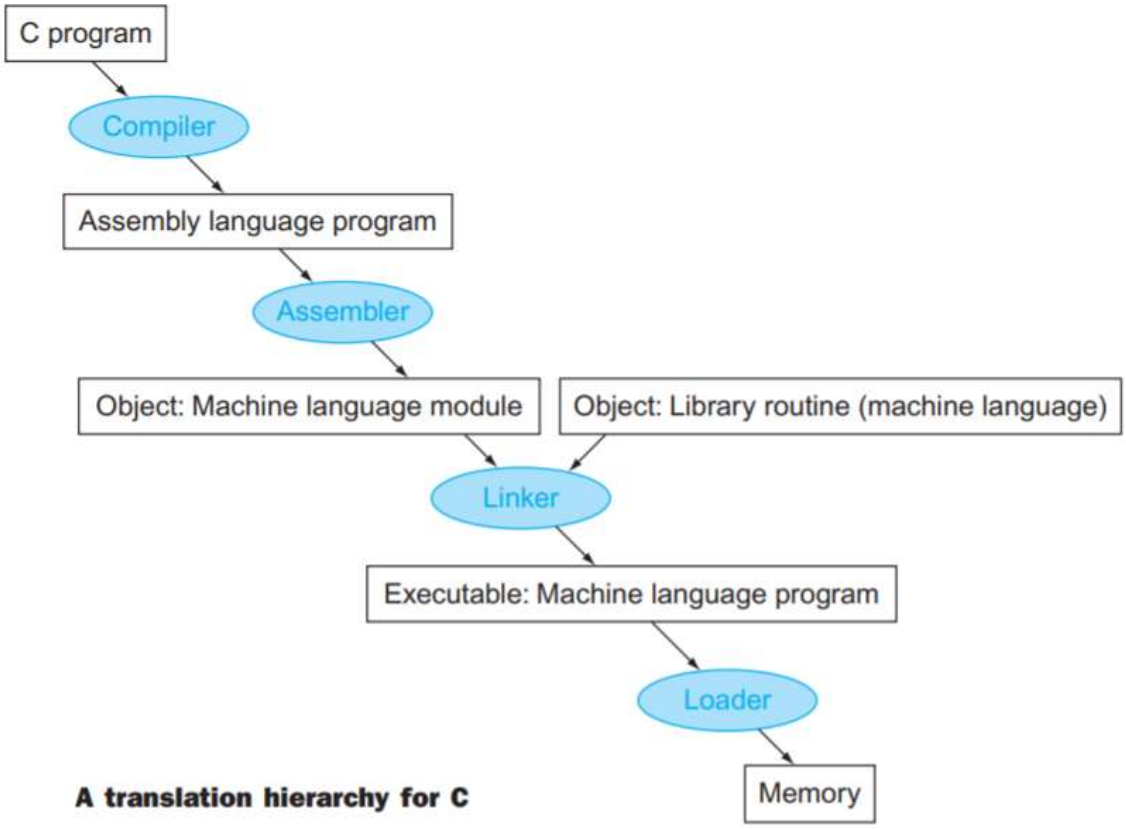
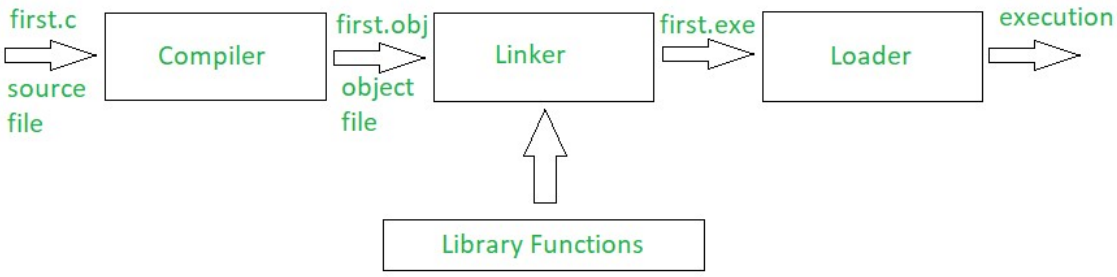
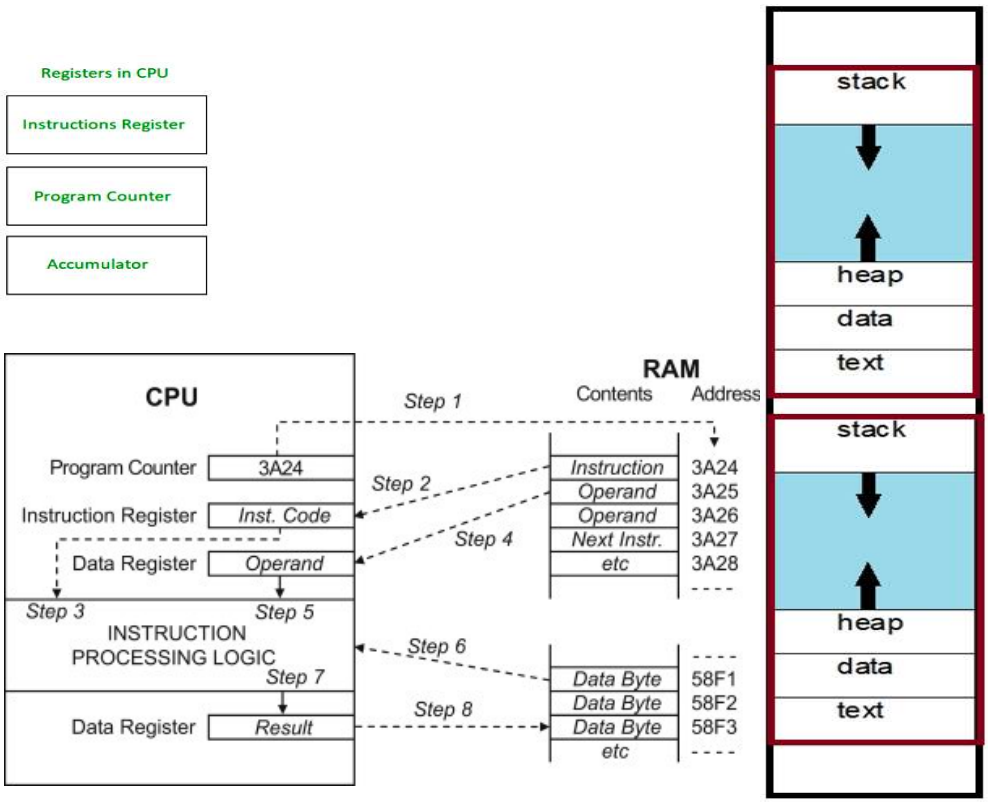


A translation hierarchy for C

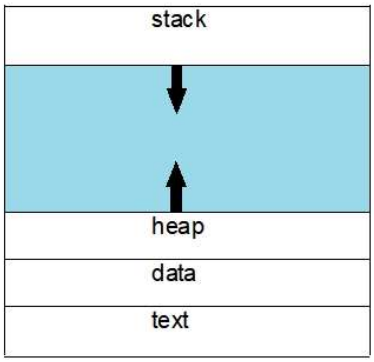




C = A + B (Add A, B Store C)

(Main, Primary)
메모리

Cycle Stage Fetch -> Decode -> Execution



Memory segment(Section) and how does a process look like.

TEXT

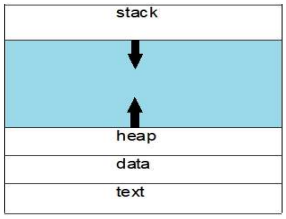
The program code or a code segment is known as Text Section.

Contains the **executable instructions** of a program.

Contains **constants**, **macros** and it is **read-only segment**.

to prevent accidentally modification of an instruction.

Another process can use this whenever it is required.



DATA

Data segment of memory contains **the global and static variables** that are initialized by the programmer prior to the execution of a program.

HEAP

To allocate **memory for variables whose size cannot be statically determined** by the compiler before program execution.

A requirement of **dynamic allocation of memory** which is done in heap segment. Managed via system calls to malloc, calloc, free, delete etc.

STACK

Stack contains **temporary data i.e. function parameters, return addresses, and local variables**.

On **x86** architecture it **grows downwards** to lower addresses. Some other architectures it may grow the **opposite direction**.

Stack grows **opposite direction** of heap for avoiding **overlapping problem**.

A **stack pointer** register keeps the tracks of the top of the stack. “pushed”, “pushed”

When a program is created

it is just pieces of Bytes stored in Hard Disk as **a passive entity**. Then the program starts

loading in memory and become **an active entity**, (double-clicked in windows, GUI) (entering executable file on the command line, CLI)

A program loaded into memory(Image) and executing(Process) is called a process. In simple, a process is a program in execution on processor.