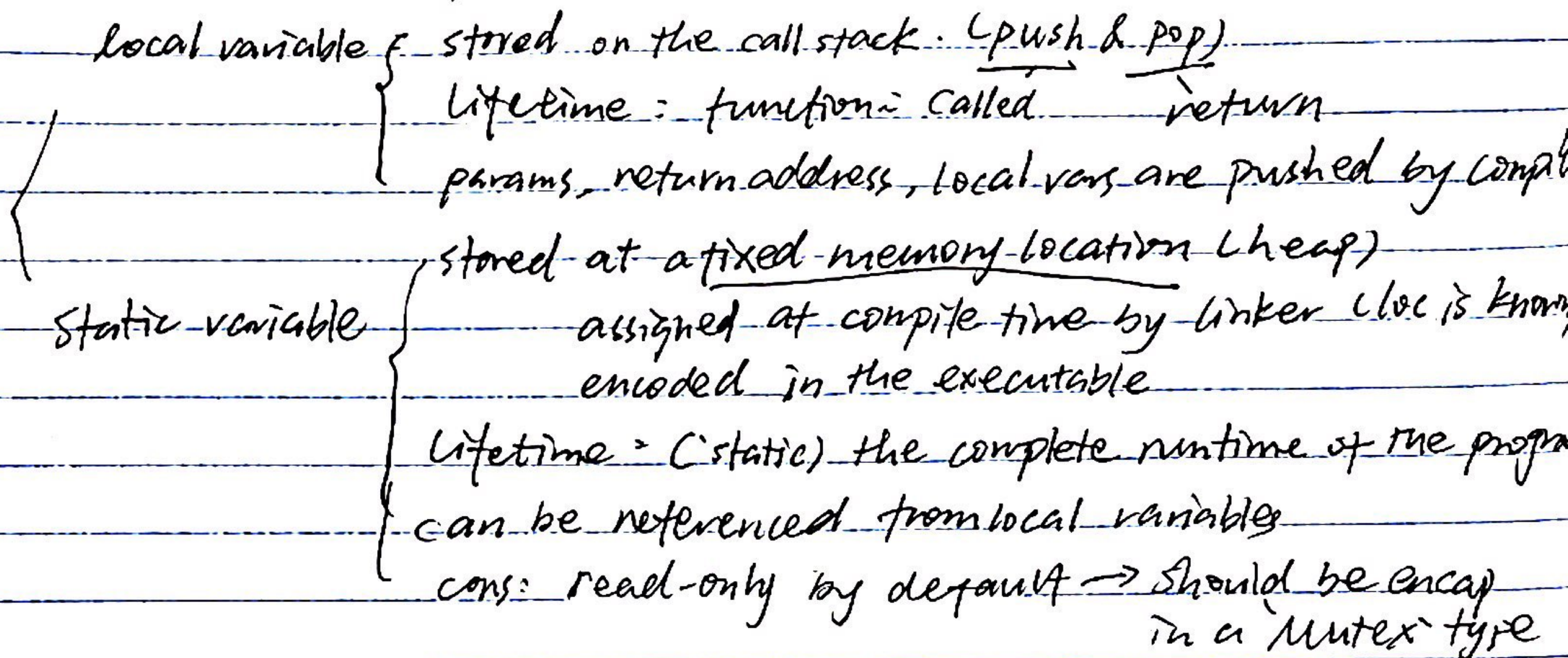


<Heap Allocation>

1. Static vs. Local Variables.



2. Dynamic Memory = Heap

(a single mutex at any point)

- heap supports dynamic memory allocation ⇒ avoid data race at runtime through [allocate] & [deallocate].

Common errors:

- ① memory leak: forget to deallocate (excessive mem consumption)
- ② use-after-free: can be exploited to exe arbitrary code
- ③ double-free: can lead to use-after-free.

↓

Garbage Collection: the program is regularly paused & scanned for (Java, Python, ...) unused heap variables ⇒ auto deallocate

↓ (assign an abstract lifetime to each reference)

Rust: Ownership: - check the correctness of dynamic memory operations at compile time

alloc: Box::new(...)

dealloc: Drop trait

(goes out of range) ⇒ no performance overhead.

⇒ no garbage collection at runtime & fine-grained control

Resource acquisition is initialization (RAII)

Rust ensures memory safety & thread

safety