10

Exam

Exercise – The heap

- Comeback of 3rd year bonus exercise
  - In Rust no_std we can not use allocators
  - Except if we create one
    - https://bd103.github.io/blog/2023-06-27-global-allocators/
    - As usual it goes with a trait
    - Your goal, implementing a global allocator, and design it in no_std so that you can use it in your kernel next time

Exercise – The heap

- The goal here is to implement a slab allocator
  - Two things there
    - First find what a slab allocator is and how it works
    - Then implement it in rust

Exercise – The heap

- Little reminder from last year

Dynamic allocator – what are they

- Implementation of the data structure type “pool”. Pre allocates resources and keep a pool of core resources that are frequently used to manage it directly
- Self managed by the program, in order to improve resource utilization and ensure the program has a fixed number of resources
- Memory pool +/= dynamic allocators

Testing your rust code

- To test your program you can use test_runner
- See one of the best blog (although on OS so you're lacking things like VGA print) https://os.phil-opp.com/testing/

Exercise – global alloc

- It’s usually a good thing to keep crates no_std compatible
  - https://www.lurklurk.org/effective-rust/no-std.html
  - https://gist.github.com/tdelabro/b2d1f2a0f94ceba72b718b92f9a7ad7b
  - https://siliconislandblog.wordpress.com/2022/04/24/writing-a-no_std-compatible-crate-in-rust/
  - https://blog.dbrgn.ch/2019/12/24/testing-for-no-std-compatibility/
  - https://github.com/hobofan/cargo-nono

Coding with features - bonus

- So one thing I would like is to keep the global alloc behind a feature
  - https://web.mit.edu/rust-lang_v1.25/arch/amd64_ubuntu1404/share/doc/rust/html/book/first-edition/conditional-compilation.html
  - https://betterprogramming.pub/compile-time-feature-flags-in-rust-why-how-when-129aada7d1b3?gi=dafd57e2f7c0

exam

- Git repo, add me as contributor
- The project is in **no_std**
- Commit are looked at, do not commit everything in one time, else it’s considered cheating
- If you take code from somewhere **it has to be credited**,else considered cheating as well
- Code quality (miri/mirai/fuzzers, other cargo utils ….) are bonus but testing is **MANDATORY**
- Unsafe must be thoroughly documented using rustdoc safety part

exam

- Comment your code and use rust doc, code exemple are appreciated for the allocation library
- No group, is to be done individually
- A report with your design choice is needed, slab allocators can be a hard thing to do, so I need to understand what you wanted to do in the first place
- Due date : 26/03/2024 23h59

exam

- If you have time (it’s adviced to do so)
  - In the second exam you’ll have to implement a FAT32 filesystem
  - You can start implementing a no_std compatible FAT32 parser
  - Won’t be taken in account for THIS exam, but will help you go faster for part II