Välkomna till ++!

++-gruppens historia

- 2016: **Simon Lindholm** och **Johan Sannemo** (16 studenter)
- 2017: Simon Lindholm och **Andreas Wallström** (7 studenter)
- 2018: Simon Lindholm och **Love Brandfeldt** (8 studenter)
- 2019: Love och **Christoffer Nolander** (7 studenter)
- 2020: Love och **Rickard Mårtenson** (9 personer)
- 2021: **Anton Lilja** och Rickard (8 personer)
- 2022: Anton och **Morris Hansing** (8 personer)
- 2023: Anton och Morris Hansing (16 personer)
- 2024: Anton och **Ludvig Lindström** (19 personer)
- 2025: Ludvig och **Tommy Bergman** (x personer)

- Tidigare **++**:are (2023)
- Systemprogrammering
- Ljudprogrammering
- Små utvecklingsverktyg
- Spelar piano
- C, Zig, Bash

Presentation

- Namn?
- Favoritspråk?
- Intressen?
- Förväntningar?

Upplägg för INDA

- **DD1337/PROGIND**: Rust, läsa + skriva kod, samarbete
- **DD1338/ALGINDA**: Algoritmer och datastrukturer
- **DD1396/PALINDA**: Parallellprogrammering, golang (speedrun!)
- **DD1349/PROJINDA**: Projektuppgift i större grupp (slowrun?)

Tidigare projekt

- Slack-klon
- Kattis-klon
- Logisim-klon
- Kernel
- VM
- Spelmotor

Allmän information

- Övningar är obligatoriska
- Nya läxor (typ) varje vecka
- Ska var klar och uppladdad **innan** övningen
- AI-genererad kod förbjudet
- Möjlighet att byta grupp
- Möjlighet att göra regular-läxor
- GitHub-organisation
- Discord

Bedömning

- All inlämningar + närvaro: A :)
- Ett betygsteg lägre för varje saknad inlämning

- Systemnivåspråk
- Används för tillämpningar som kräver hög prestanda
- T.ex. ingen garbage collection
- Finns stöd för generiska funktioner/parametriska typer
- Finns vissa drag av objektorienterade koncept

Referenser

- Printa en string!

```
fn print_string(s: String) {
    println!("{}", s);
}

pub fn main() {
    let s = "Hej!".to_string();
    print_string(s);
}
```

Referenser

- Gör det igen!

```
fn print_string(s: String) {
    println!("{}", s);
}

pub fn main() {
    let s = "Hej!".to_string();
    print_string(s);
    print_string(s);
}
```

Referenser

- Ånej...

```
$ NO_COLOR=1 rustc example.rs
error[E0382]: use of moved value: `s`
 --> example.rs:8:18
  |
6 |     let s = "Hej!".to_string();
  |         - move occurs because `s` has type `String`, which does
not implement the `Copy` trait
7 |     print_string(s);
  |                  - value moved here
8 |     print_string(s);
  |                  ^ value used here after move
```

Referenser

- Referenser!

```
fn print_string(s: &String) {
    println!("{}", s);
}

pub fn main() {
    let s = "Hej!".to_string();
    print_string(&s);
    print_string(&s);
}
```

Mutable references

- Om man vill ändra ett värde via en referens

```
pub fn main() {
    let mut s = "Hej!".to_string();
    let t = &mut s;
    *t += " Hej igen!";
    println!("{}", *t);
}
```

Regler för referenser

Det går att ha:

- **Godtyckligt många vanliga referenser till ett värde**, eller
- **Exakt en muterbar referens till ett värde**

...men aldrig samtidigt!

`enum` i Rust

- Utökar `enum` som det fungerar i t.ex. C med tillhörande data.
- Kallas ibland **tagged union**.

```
enum MyEnum {
    Integer(i32),
    Name(&'static str),
}
```

```
fn print_enum(e: &MyEnum) {
    match e {
        MyEnum::Integer(value) => println!(
                "it's an integer with value {}",
                value
        ),
        MyEnum::Name(name) => println!(
          "it's a name: '{}'",
          name
        ),
    }
}
```

```
pub fn main() {
    let e = MyEnum::Integer(32);
    print_enum(&e);
    let e = MyEnum::Name("Rutger");
    print_enum(&e);
}
```

- Output:

```
$./example
it's an integer with value 32
it's a name: 'Rutger'
```

`Option` i Rust

- Motsvarar typ `NULL` i andra språk
- Kan användas när det inte alltid finns ett värde
- T.ex. hitta index för elementet x i en lista

```
enum Option<T> {
  None,
  Some(T),
}
```

`Result` i Rust

- Används konsekvent för felhantering
- Kan innehålla ett “lyckat” värde eller information om vad som gick fel

```
enum Result<T, E> {
  Ok(T),
  Error(E),
}
```

Veckans läxa

- Skapa ett schackbibliotek i Rust
- Vi håller på med denna läxa i två veckor
- Fundera över hur biblioteket ska användas
  - (Nästa läxa är att skriva ett GUI utifrån en kamrats bibliotek)
- Tester/testprogram?
- Några länkar:
  - Rustboken
  - Chess programming wiki
  - Skapa bibliotek med Cargo

Upplägg

- **Första veckan**: Fokusera på representation för bräde och pjäser, implementera vanliga drag.
- **Andra veckan**: Implementera schack + matt och specialdrag: en passant, rockad...

Discord/GitHub

- Mejla mig (ludviggl@kth.se):
  - Användarnamn på GitHub
  - Användarnamn på Discord
- Byt namn i servern till `<riktigt namn> + <kth id>`
- Mejla/säg till om nåt går snett