COMP6991 23T3

Slices & Lifetimes

Ownership & Borrowing

...recapped

Type	Requirements	Access
`T`	Exactly one owner	Read & Write
`&T`	Only shared borrows can coexist	Read only
`&mut T`	No other borrows can coexist	Read & Write

Slices 🍕

Example time!

> Example: slice.rs

Slices

Type	Layout	Access
`[T; N]`	Contiguous, exact length	Owned (or Copy)
`Vec<T>`	Contiguous, dynamic length	Owned
`&[T]`	Shared borrow of a contiguous subsequence	Read only borrow
`&mut [T]`	Exclusive borrow of a contiguous subsequence (cannot extend nor shrink)	Read write borrow

Slices

Type	Layout	Access
`String`	Contiguous, dynamic length	Owned
`&str`	Shared borrow of a contiguous subsequence	Read only borrow
`&mut str`	Exclusive borrow of a contiguous subsequence (cannot extend nor shrink)	Read write borrow

Lifetimes

Example time!

> Example: dangling.rs

Annotating lifetimes

Example time!

> Example: longest.rs

Annotations on structs & enums

Example time!

```
> Example: struct_lifetime.rs
```

'static lifetime

> What type is a string literal? e.g. "foo"

> What is the lifetime of that literal?

> What about borrowing a global variable?

Eliding lifetimes

Example time!

> Example: elision.rs

Smart pointers

... if we have time

Type	Location	Borrowing	Limitations
T	Stack	Owned	Must have a fixed size known at compile-time
Box<T>	Heap	Owned	Peformance, memory usage
Rc<T>	Heap	Shared without lifetimes!	Read-only, performance, memory usage, reference cycles
RefCell<T>	Stack	Owned, allowing dynamic borrowck	Incorrect borrowing causes panic at runtime