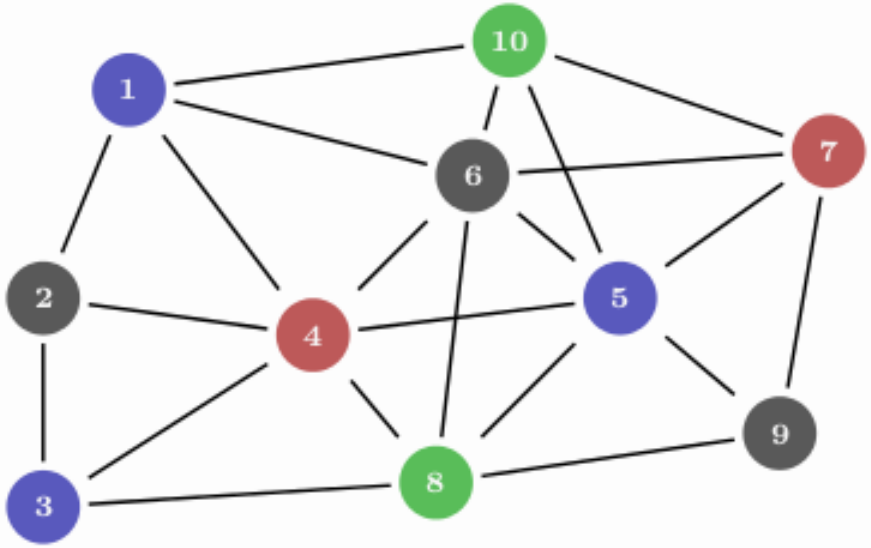
Rust 编程语言入门

杨旭，微软 MVP
Rust、Go 开发者

15.4 Rc<T>：引用计数智能指针

Rc<T>引用计数智能指针

- 有时，一个值会有多个所有者
- 例如：
- 为了支持多重所有权：Rc<T>
  - reference couting（引用计数）
  - 追踪所有到值的引用
  - 0 个引用：该值可以被清理掉

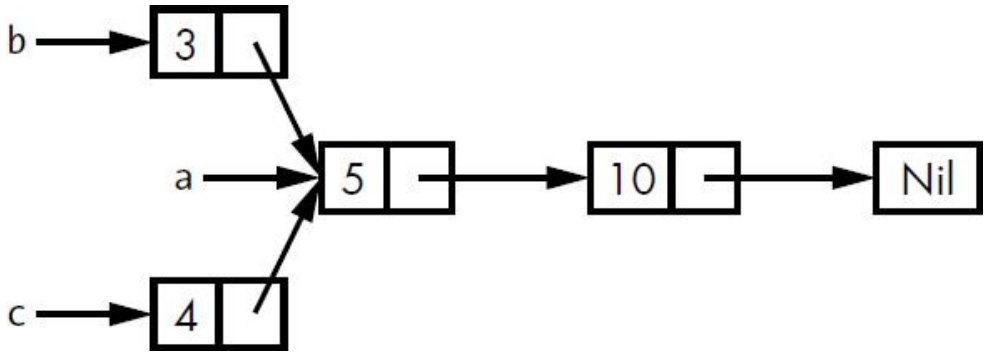


Rc<T> 使用场景

- 需要在 heap 上分配数据，这些数据被程序的多个部分读取（只读），但在编译时无法确定哪个部分最后使用完这些数据
- Rc<T> 只能用于单线程场景

例子

- Rc<T> 不在预导入模块（prelude）
- Rc::clone(&a) 函数：增加引用计数
- Rc::strong_count(&a)：获得引用计数
  - 还有 Rc::weak_count 函数
- （例子）
  - 两个 List 共享 另一个 List 的所有权



Rc::clone() vs 类型的 clone() 方法

- Rc::clone() ：增加引用，不会执行数据的深度拷贝操作
- 类型的 clone()：很多会执行数据的深度拷贝操作。
- （例子）

Rc<T>

- Rc<T> **通过不可变引用**，使你可以在程序不同部分之间共享只读数据
- 但是，如何允许数据变化呢？

再见