第25节课内容总结

weak原理

weak在底层维护了一张weak_table_t结构的hash表，key是所指对象的地址，value是weak指针的地址数组。weak所引用对象的引用计数不会加1，对象释放时，会根据对象地址获取所有weak指针地址的数组，然后遍历这个数组把其中的数据设为nil，最后把这个entry从weak表中删除。

添加弱引用流程总结：

- 如果被弱引用的对象为nil 或这是一个TaggedPointer，直接返回，不做任何操作。
- 如果被弱引用对象正在析构，则抛出异常。
- 如果被弱引用对象不能被weak引用，直接返回nil。
- 如果对象没有再析构且可以被weak引用，则调用weak_entry_for_referent 方法根据弱引用对象的地址从弱引用表中找到对应的weak_entry，如果能够找到则调用append_referrer 方法向其中插入weak指针地址。否则新建一个weak_entry。

移除弱引用流程总结：

- 首先在weak_table中找出被弱引用对象对应的weak_entry_t。
- 在weak_entry_t中移除weak指针地址。
- 移除元素后，判断此时weak_entry_t中是否还有元素，如果此时weak_entry_t已经没有元素了，则需要将weak_entry_t从weak_table中移除。

自动释放池

AutoreleasePool（自动释放池）是OC中的一种内存自动回收机制，它可以延迟加入AutoreleasePool中的变量release的时机。在正常情况下，创建的变量会在超出其作用域的时候release，但是如果将变量加入AutoreleasePool，那么release将延迟执行。