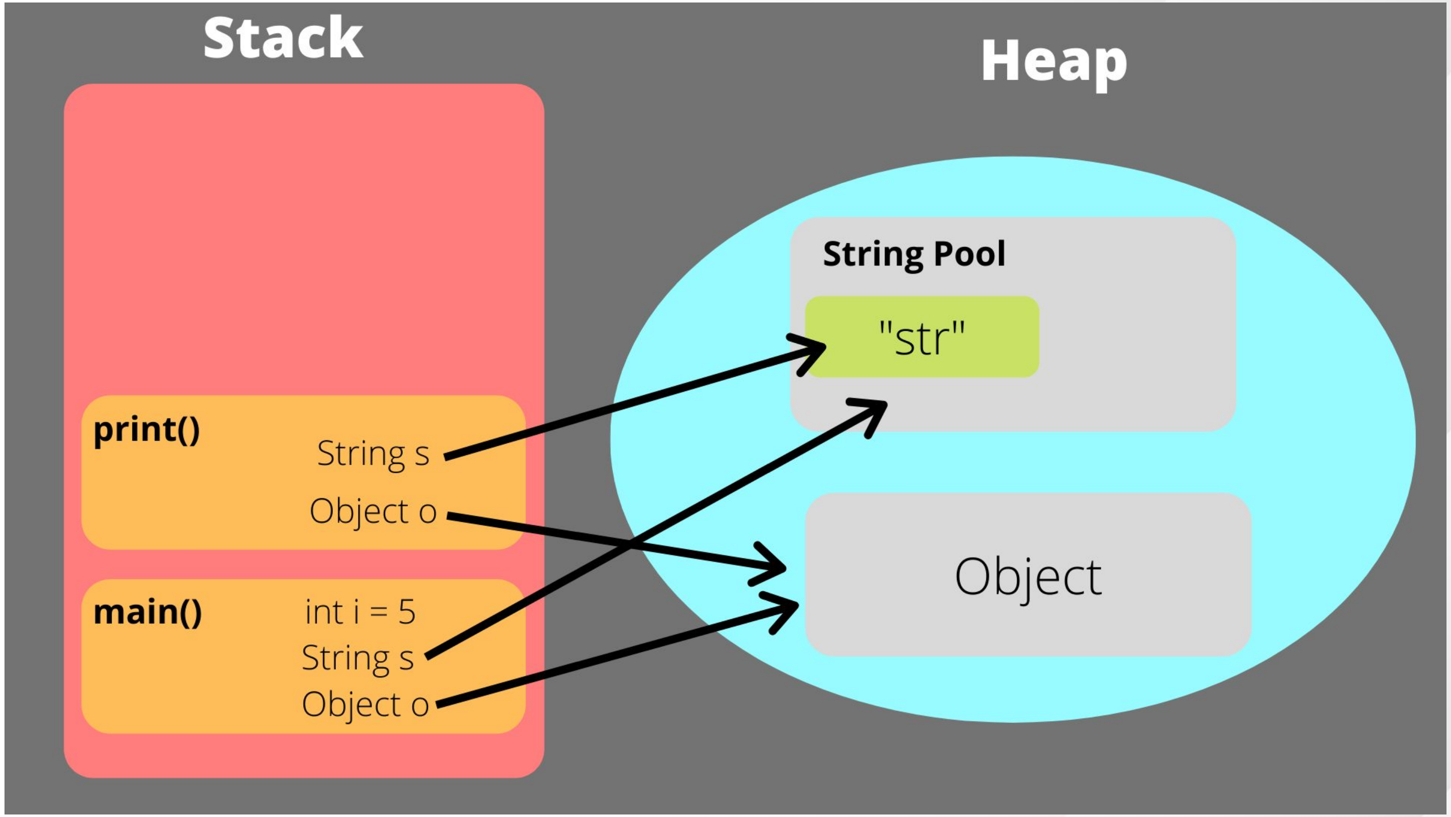
ILLINOIS

Ownership of Memory

Learning Objectives

1. Differentiate between stack and heap memory
2. Identify ownership responsibility in programs
3. Enumerate the “Rule of Three”

Memory Management - Ownership



Memory Management - Ownership

Stack

Heap

Example

```
class GymMember {
        public:
            GymMember() {
                locker = new int(LockerSize);
            }

            ~GymMember() {
                delete locker;
            }

            int getStoredWeight() const {
                return *locker;
            }

        private:
            int* locker;  // Owned memory
};
```

```
class GymOwner {
public:
    GymOwner(int* memberLocker) {
        lockerView = memberLocker;
    }

    int inspect() const {
        return *lockerView;
    }

private:
    int* lockerView;
};
```

Example

Locks

I/O Devices

Network Connections

Why is ownership important

1. Explicit - Crash
2. Implicit - Memory is consumed unnecessarily

Rule of Three

If you must define one of these functions in a class, then you must define all of them

1.

2.

3.

Rule of Zero

“Classes that declare custom destructors, copy/move constructors or copy/move assignment operators should deal exclusively with ownership. Other classes should not declare custom destructors, copy/move constructors or copy/move assignment operators”

–Scott Meyers