# C++ SMART POINTERS

Just like a regular pointer but it automatically frees memory.

# 2 KINDS

- `shared_ptr`: multiple pointers can point to the same object
- `unique_ptr`: pointer “owns” object it points to.

# LIKE VECTORS

- they are C++ templates. need to supply additional info in a manner similar to vectors

- `shared_ptr<int> piCounter;`

  By default: NULL

- `shared_ptr<float> pfFrequency;`

# COMMON OPERATIONS

- `shared_ptr<T> sp;` Null smart pointer of type T
  `unique_ptr<T> up;`
- `*p` Dereference
- `p->x` Synonym for (*p).x

# SHARED OPERATIONS

- `make_shared<T>(args)`

  ```
  make_shared<int>(42);
  make_shared<Rect>(2, 2, 3, 8);
  ```

- `p = q`

# EXAMPLE

```
#include <iostream>
#include <memory>
using namespace std;

int main(){
    shared_ptr<int> piLife = make_shared<int>(42);
    shared_ptr<int> piLive = piLife;
    *piLive = 12;
    cout << *piLife << endl;
}
```

That’s all you need to know to get started!