## Today
- Pointer Refresher

## Assignments
- HW3 out (due Tues)
- Quiz on Tues
Pointer Refresher

“Normal” variables imply plain-old values are stored in memory

```c
int x = 42;
```

```
42
mem loc @111
```

**Pointer** variables store the address of a memory location

```c
int* xptr = &x; // & is the "address of" operator
```

```
@111 42
mem loc @222 mem loc @111
```

**Dereferencing** a pointer accesses value at the pointer’s stored address

```c
*xptr = 57; // * here is the "dereference" operator
```

```
@111 42 57
mem loc @222 mem loc @111
```

Exercise 1
**Dynamically allocating** memory at runtime using **new**

```cpp
int* yptr = new int;  // allocate new memory on the fly

*yptr = 96;
```

- Note that **new** always returns a pointer
- The new memory is allocated on the “heap” (not the “stack”)

**Deallocating** memory at runtime using **delete**

- memory dynamically allocated lives on unless explicitly deallocated
- “**memory leaks**” imply nothing can access allocated memory

```cpp
delete yptr;  // deallocate the previously allocated memory
yptr = nullptr;  // reset the variable to be null
```

**Exercise 2**
Dereferencing object (class or struct) pointers

    // create a new object
    MyClass* ptr = new MyClass;

    // doesn't work since '.' has precedence over '*'
    *ptr.att1 = 42;

    // this works, but not good style
    (*ptr).att1 = 42;

    // this uses the preferred syntax
    ptr->att1 = 42;