

For Friday

- Read Weiss, chapter 1, sections 1-3

For Monday

- C++ Practice 5
 - Described on Blackboard
 - Submit from the Suns
 - Worth 2 homeworks

Pointers and Classes

- Can declare pointers to a class type (just like to an int or double)
- Have two choices for dereferencing.
- Can use
`(*objPtr).Method()`
or
`objPtr->Method()`

this

- Special variable to refer to the calling object.
- Most C++ programmers always call methods of the calling object using the *this* pointer.

Dynamic Allocation

- Pointer values created using the address operator are of limited utility
- Dynamic allocation creates variables in memory that only have a pointer (no direct variable name)
- Note that this allows us to create an array of arbitrary size at run-time

How Do We Do It?

- Memory allocation is done using `new`
- Examples:

```
int *intPtr = new int;
```

```
int *arrPtr;
```

```
arrPtr = new int[50];
```

```
Student *myStuPtr1, *myStuPtr2;
```

```
myStuPtr1 = new Student;
```

```
myStuPtr2 = new Student("Mary Smith", 3.45);
```

Freeing the Memory

- When we finish using statically allocated variables, the memory used by them is freed up (actually when the function ends).
- Dynamically declared variables must be freed by the programmer.
- Done using delete:

```
delete intPtr;
```

```
delete [] arrPtr;
```

Memory Leaks

```
int *intPtr = new int;
```

```
*intPtr = 5;
```

```
intPtr = new int;
```

```
*intPtr = 10;
```

```
// What happened to the first one?
```

Notes on Dynamic Arrays

- Need three pieces of information associated with an array
 - Where's the data?
 - What's the capacity?
 - What's the size (current number of elements)?

Dynamic Classes

- Classes that use dynamic memory to store some/all of their data members
- Constructor typically allocates/initializes the dynamic portions of the class

Destructors

- Used to free the dynamically allocated portions of an object when the object is destroyed (either by delete or by going out of scope if statically allocated).
- Called ~classname
- Like constructor, has no return type
- Also always has no parameters

Assignment and Initialization

- What's the difference?

Assignment

- Default assignment operator
 - does a byte-by-byte copy of the object
 - Why is this an issue?
- Solution:
 - Create your own assignment operator
 - Either copy the pointer or the data as appropriate for your class
 - Must be a member function
 - Always returns `*this`

Initialization

- Copy Constructor

Files in C++

- ifstream
- ofstream
- Names must be c-strings
- Same basic operators

File reading loops

- No look ahead . . .
- Think sentinel