Class Design

Joel Willoughby

COP 3503

Outline

1 Introduction

2 Pitfalls of Last Lab

3 This Lab

4 Wrapping Up

Agenda

- Lab attendance changes
- Talk about some of the recurring problems from last lab
- Turning a non-class design into a class

Lab Attendance

- Up to 40 percent of your labs from here on out will be deducted if you do not show up for lab
- If you must miss lab for some valid reason, notify me and Dr. Nemo before the lab happens and we will see what we can work out
- If notification is not received from you before the day of the lab, you will be considered absent
- Be sure to sign the sign-in sheet when you are in lab, or you will be counted absent
- To compensate for real emergency's, you will be allowed one dropped lab

Outline

1 Introduction

2 Pitfalls of Last Lab

3 This Lab

4 Wrapping Up

Compiling .h Files

- Be careful compiling .h files
- It is legal to do g++ whatever.h
- It gives you whatever.h.gch
- This is a pre-compiled header
- If you have a pre-compiled header, g++ will use that to compile your program instead of your .h file

Not initializing variables

```
for(int i; i<num; i++){
  arr[i] = something();
}
```

- What is the problem with the above section of code?

Not initializing variables

```
for(int i; i<num; i++){
  arr[i] = something();
}
```

- What is the problem with the above section of code?
- The variable i is never initialized, so the statement arr[i] could be accessing anything...
- These can be very tricky bugs to isolate because they are hard to reproduce
- This is the same reason you should write constructors

Outline

1 Introduction

2 Pitfalls of Last Lab

3 This Lab

4 Wrapping Up

Thinking OOP

- This is one way to think
- Write down (in English) a description of what your program should do
- Any recurring nouns in the description are likely to be classes or objects
- Any verbs or actions done by these objects are good candidates for methods or functions

Going Further

- Note that one item can be made up of other items. These will be the fields or members of the class but they can also be classes themselves
- Also, actions may be able to be broken down into smaller, repeatable actions. These are good auxillary or helper methods
- Often, we have objects in our description that interact with other objects but aren't necessarily a part of them. These will usually be parameters to the methods.
- Later we will see inheritance, templates, etc.

This lab: A Group of people

Open discussion in lab

Outline

1 Introduction

2 Pitfalls of Last Lab

3 This Lab

4 Wrapping Up

Questions

???