MIDTERM

- 1 hr 50 min
- 9am 11-5-2010 in NH454
- Online students : SATURDAY 9:30-11:20 somewhere! SH212
- testing@pdx.edu

- Closed Book
- closed Notes
- Picture ID

Topics

- Single inheritance
- Dynamic Binding — RTTI (no code!)
- Copy Constructors — deep / shallow
- OOP — is a / has a
- Data Structures
  - LLL
  - DLL
  - CLL
  - BST

Midterm (202-SU10) *(100 points)*
CS 202: Programming Systems

Name: _____

US MAIL Address: _____

1. Short Answer

a. Why do we need a Copy constructor?

Do we need to write one for every class?

Do we send a pointer or a reference in as an argument?

Account (const Account &); ← f(obj)

Account (const Account *); ← passing value

b. Explain the difference between "is a" and "has a".

What is the syntax that makes a relationship "is a"

c. What does it mean to have a pure virtual function

And…what is the syntax:

d. Explain the benefit of passing constant references as arguments in general:

2. **Dynamic Memory**

Imagine that you were implementing a class that managed a **binary search tree** of **dynamically allocated arrays of characters**. Your underlying node structure is as follows. (You may use strcpy, strcmp, strlen)

```
struct node {
    char * name;
    node * right;
```

```
        node * left;
    };
```

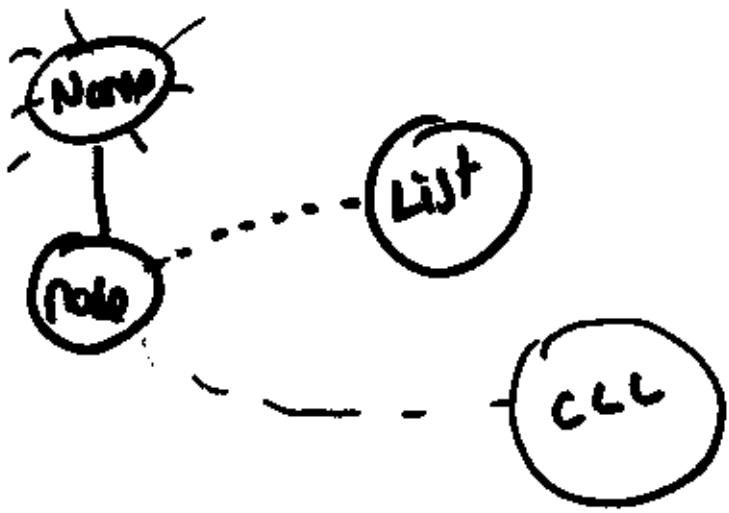
a. Implement the Copy Constructor
(If you call another function – you must write it!)
assume your class only has a root pointer as a private data member:

b. Now, assume you have a **linear linked list** with just a head pointer, write the code to remove the last node in the linear linked list

as a data member

3. Inheritance and classes

a. On the previous page, we used a node structure. Re-write the node as a 'class', following the rules of OOP; Write the class interface (you do not need to implement the member functions)



b. Explain why we use dynamic binding?

c. When do we use up-casting versus down-casting

d. Show the syntax needed to perform dynamic binding – show only relevant code and explain what is going on with the code

e. what is the power of downcasting

f. which needs a type conversion operation

1) Upcasting
2) downcasting
3) initialization lists
4) RTTI
5) derivation list
6) pure virtual
7) Abstract base class

e. Let's say for example you were to design an object oriented solution for your local coffee shop – they have regular coffee and decaf, specialized coffee drinks, premium hot chocolate, donuts and other pastries, and then juices and waters for sale. What classes might make sense – and what relationships (has a versus is a) could exist. *** NO CODE***

b. Show an example of an initialization list:

- Where does it go?

c. We have learned about abstract base classes.
   i. What are they?

ii. When might you want to use one?

iii. Is there ever the case when an abstract base class DOESN'T make sense?

iv. What is the syntax to create an abstract base class (for an "attribute" class)