

CS 91.204 Computing IV:
Advanced OO Programming and
Software Tools

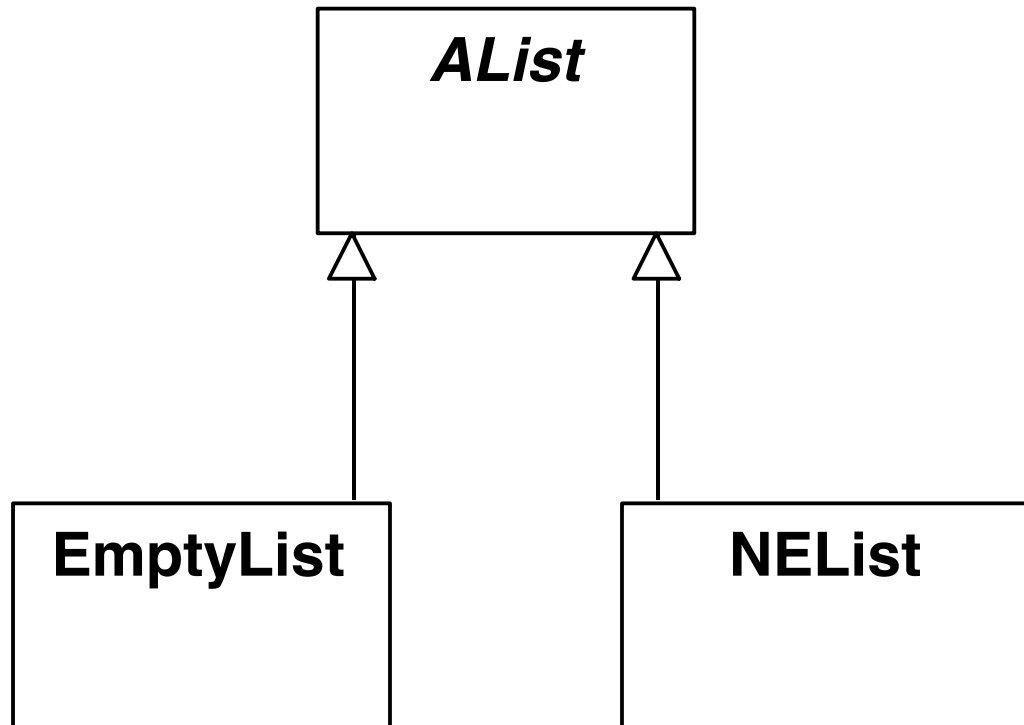
OO List II

Prof. Li Xu
Dept. of Computer Science
UMass Lowell
Spring 2009

Acknowledgements

- OO List used in 204 class is developed by Dr Dung Nguyen at Rice University.
- All rights are reserved

OO List Design



The Length Method

- Abstract base class

```
public abstract class AList
{
    public abstract Object getFirst();
    public abstract AList getRest();
    public abstract int length();
}
```

EmptyList

- getFirst, getRest not well defined
- null
- why null is not a good answer

EmptyList: The Operations

```
public class EmptyList extends AList
{
    public Object getFirst()
    {
        throw new IllegalArgumentException (
            "Empty List has no data.");
    }

    public AList getRest()
    {
        throw new IllegalArgumentException (
            "Empty List has no tail.");
    }
}
```

Exceptions

- An exception is an event that occurs during the execution of a program that disrupts the normal execution flow.
- When such an event occurs within a Java method, the method creates an exception object, which describes the exception, and hands it off to Java VM, which is responsible for finding code to handle the exception.

The Design: NEList

```
public class NEList extends AList
{
    private Object first;
    private AList rest;
    public NEList(Object data, AList l) {
        first = data;
        rest = l;
    }
    public Object getFirst() {
        return first;
    }
    public AList getRest() {
        return rest;
    }
}
```

EmptyList: length

- What is length of EmptyList
- EmptyList:

```
public int length()  
{  
    return 0;  
}
```

NEList: length

- How do we compute length of NEList?
 - NEList
 - first element
 - rest of list
 - What would you do?
- How to delegate to rest of list?

NEList: length

- What's the type of **rest** (of list)?
 - AList
 - So rest should be able to compute its length (it is an AList)
- Just call `rest.length()` and add 1
- Principle of OO: Delegation
 - Always see whether you can delegate
 - minimal design of objects
 - co-operative OO world view

NEList: length

- NEList

```
public int length()
{
    return 1 + rest.length();
}
```

OO List in Action

```
AList listo = new EmptyList();
```

```
AList list1 = new NEList(new Integer(5),  
                        listo);
```

```
AList list2 = new NEList(new Integer(55),  
                        list1);
```

```
System.out.println(listo.length());
```

```
System.out.println(list1.length());
```

```
System.out.println(list2.length());
```