ECE 2574: Data Structures and Algorithms - List ADT

C. L. Wyatt

Today we will introduce the List ADT, write an abstract base for it, and write some tests.

- Warmup
- Ordered List ADT
- AbstractList
- Tests for List implementations

Warmup #1

Consider the following code using the List ADT as defined in Chapter 8.

```
List<int> l;
bool result = l.insert(0,42);
```

What is the value of result?

- True 29%
- False 71% (correct)

Warmup #2

Consider the following code using the List ADT as defined in Chapter 8.

```
List<int> l;
l.insert(1,42);
l.insert(1,24);
l.insert(3,-1);
```

What are the contents of the list written left-to-right in by increasing index?

- `[0 42 24 -1]` 6%
- `[42 24 -1]` 19%
- `[24 42 -1]` 75%
- `[24 42 -1 0]` 0%

The Ordered List ADT

- Test if a list is empty

`+isEmpty(): boolean`

- Get the number of entries in the list

`+getLength(): integer`

- Insert an entry at a given position in the list

`+insert(newPosition: integer, newEntry: ItemType) : boolean`
Note: uses 1-based indexing, shifts entries $>$ newPosition up

The Ordered List ADT

- Remove entry at given position from the list

`+remove(position: integer): boolean`
Note: uses 1-based indexing, shifts entries $<$ position down

- remove all entries (clear)

`+clear(): void`

The Ordered List ADT

- get a copy of the item at a given position

`+getEntry(position: integer): ItemType`
Note: uses 1-based indexing

- replace the value of the item at a given position

`+setEntry(position: integer, newValue: ItemType):`
`void`
Note: uses 1-based indexing

An abstract base class for List implementations

see code

Testing the List

see code

A simple fixed-length array-based list implementation

see code

Next Actions and Reminders

- Read CH pp. 265-271 (section 9.1)
- Complete the warmup before noon on Friday