

# Lists

## Organizing Data Linearly

# 2A

ArrayList



## Lists



- A list is a linear sequence of data.
- List interface in Java

```
int size()
boolean add(E element)
void add(int index, E element)
E remove(int index)
E get(int index)
E set(int index, E element)
```

# ArrayList



- Implements the `List` interface
- Acts like an array but
  - The size of the `ArrayList` instance increases and decreases as necessary based on the number of elements in the data structure.
  - Each element has an index that starts with 0, just like arrays, but we do not use subscript notation.

```
ArrayList myList = new ArrayList();
...
myList.add(3, someObject);
Object someObject = myList.get(1);
```

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

3

## ArrayList (pre Java 5)

ArrayList holds references to Object



```
ArrayList myList = new ArrayList();
```

```
myList.add("Steelers");
myList.add("Pirates");
myList.add("Penguins");
```

← add method implicitly widens String type to Object

```
String a = (String)myList.get(0);
String b = (String)myList.get(1);
String c = (String)myList.get(2);
```

← typecasting is needed to explicitly narrow Object type to String

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

4

## ArrayList (Java 5: Generics)

ArrayList holds references to String



```
ArrayList<String> myList =  
    new ArrayList<String>();
```

```
myList.add("Steelers");  
myList.add("Pirates");  
myList.add("Penguins");
```

add method requires String argument only; otherwise an error will occur during compilation

```
String a = myList.get(0);  
String b = myList.get(1);  
String c = myList.get(2);
```

no typecasting is needed since returned object is a String

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

5

## Autoboxing



- ArrayLists can only store references to objects, not primitives.
- All primitive types have a corresponding object type (wrapper class).

example: `int`          `Integer`

```
Integer i = 23;  
Integer j = new Integer(45);  
Integer k = new Integer("67");
```

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

6

## Autoboxing (cont'd)



```
Integer m = i++;
Integer n = ++j;
int p = m + n;
System.out.println(
    m.toString() + n.toString());
System.out.println(m + n);

ArrayList<Integer> numList
    = new ArrayList<Integer>();
```

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

7

## Implementing Array Lists



- We can use an array to implement an array list, like we did in the phone book application.

```
public class CMUArrayList<E> {
    private E[] dataArray;
    private int numData;

    public CMUArrayList() {
        dataArray = (E[]) new Object[1];
        numData = 0;
    }
}
```

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

8

## Implementing Array Lists (cont'd)



```
public boolean add(E newEntry) {
    if (numData == dataArray.length)
        reallocate();
    dataArray[numData] = newEntry;
    numData++;
    return true;
}
```

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

9

## Implementing Array Lists (cont'd)



```
private void reallocate() {
    E[] newArray =
        (E[]) new Object[dataArray.length*2];
    System.arraycopy(dataArray, 0, newArray, 0,
        dataArray.length);
    dataArray = newArray;
}
```

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

10

## Implementing Array Lists (cont'd)



```
public E remove(int index) {
    if (index < 0 || index >= numData)
        throw new ArrayIndexOutOfBoundsException(index);
    E result = dataArray[index];
    for (int i = index+1; i < numData; i++)
        dataArray[i-1] = dataArray[i];
    return result;
}
```

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

11

## Implementing Array Lists (cont'd)



What is the worst-case order of complexity for  
add on a list with  $n$  data values?

What is the worst-case order of complexity for  
remove on a list with  $n$  data values?

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

12

## Implementing Array Lists (cont'd)



```
public E get(int index) {
    if (index < 0 || index >= numData)
        throw new ArrayIndexOutOfBoundsException(index);
    return dataArray[index];
}
public E set(int index, E newValue) {
    if (index < 0 || index >= numData)
        throw new ArrayIndexOutOfBoundsException(index);
    E oldValue = dataArray[index];
    dataArray[index] = newValue;
    return oldValue;
}
```

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

13

## Implementing Array Lists (cont'd)



What is the worst-case order of complexity for  
get on a list with  $n$  data values?

What is the worst-case order of complexity for  
set on a list with  $n$  data values?

15-121 Introduction to Data Structures, Carnegie Mellon University - CORTINA

14

## Implementing Array Lists (cont'd)



Write a method for

```
public void add(int index, E newEntry)
```