ALGONQUIN COLLEGE

CST8130 – Data Structures

Professor : Dr. Anu Thomas
Email: thomasa@algonquincollege.com
Office: T314

Collection Classes

Collection Classes

Definition:

- A **collection** is an object that can hold references to other objects.
- **Collection** interfaces declare operations that can be performed on each type of **collection**.
- Classes and interfaces of **collections** framework are in package **java**.util.

ArrayList

- **ArrayList** is a resizable, ordered collection of elements.
- Internally, ArrayList implements a dynamically allocated array
- Declaration: **ArrayList<E>;** where E is the class type of element to be stored in the list.

From the Java documentation:

- Resizable-array implementation of the List interface.
- Implements all optional list operations, and permits all elements, including null.
- In addition to implementing the List interface, this class provides methods to manipulate the size of the array that is used internally to store the list

Capacity

- Each ArrayList instance has a capacity which is the size of the array used to hold the elements in the list
- It is always at least as large as the array list size
- As elements are added to the ArrayList, its capacity will grow automatically (note…by creating larger capacity array, and copying the smaller one to the larger one).

Constructors

ArrayList()

Constructs an empty list with an initial capacity of ten.

Example:

ArrayList<Planner> = new ArrayList<Planner>();

ArrayList(int initialCapacity)

Constructs an empty list with the specified initial capacity.

Example:

ArrayList<Planner> = new ArrayList<Planner>(maxEntries);

Selected Methods

boolean	**add**(**E** e)Appends the specified element to the end of this list.
void	**add**(int index, **E** element)Inserts the specified element at the specified position in this list.
boolean	**contains**(**Object** o)Returns true if this list contains the specified element.
void	**ensureCapacity**(int minCapacity)Increases the capacity of this ArrayList instance, if necessary, to ensure that it can hold at least the number of elements specified by the minimum capacity argument.

Methods - continued

E	get(int index)Returns the element at the specified position in this list.
int	indexOf(Object o)Returns the index of the first occurrence of the specified element in this list, or -1 if this list does not contain the element.
boolean	isEmpty()Returns true if this list contains no elements.
E	remove(int index)Removes the element at the specified position in this list.
boolean	remove(Object o)Removes the first occurrence of the specified element from this list, if it is present.

Methods – contd.

int	**size**()Returns the number of elements in this list.
void	**sort**(**Comparator**<? super **E**> c)Sorts this list according to the order induced by the specified **Comparator**.

Example

```
public static void main (String [] args) {
    ArrayList<String> names = new ArrayList<String>(5);
   names.add(“Anu”);
   names.add(“Thomas”);
   System.out.println (“The list is currently:”);
    System.out.println (names);
}
```

The list is currently:

[Anu, Thomas]

Example (contd.)

```
System.out.println (“The list is now:”);
  for (int i=0; i<names.size(); i++)
      System.out.println (names.get(i));
```

The list is now:

Anu

Thomas

Example continued (if you have a method to execute on each object)

```
System.out.println (“The list is also:”);
    for (String name:names)
        System.out.println (name);
```

The list is also:

Anu

Thomas

Questions?