Java

Data structures

Lecture objectives

- The Java programming platform contains two general-purpose List implementations –ArrayList and LinkedList, both of which offer better performance.

- To be able to understand
  - Tables –two dimentional arrays
  - ArrayList
  - LinkedList

2D arrays

- One dimensional arrays
  - Sorting –Array.sort(<Array Name>)
    - Topic 9 Task4 illustration
- Table is a commonly used data structure
  - Data arranged in rows and columns
  - Represented by a two dimensional array data structure
- Two dimensional array elements
  - String[][] names = new String[4][4];
  - Elements accessible via row& column index
  - Multi-dimensional arrays

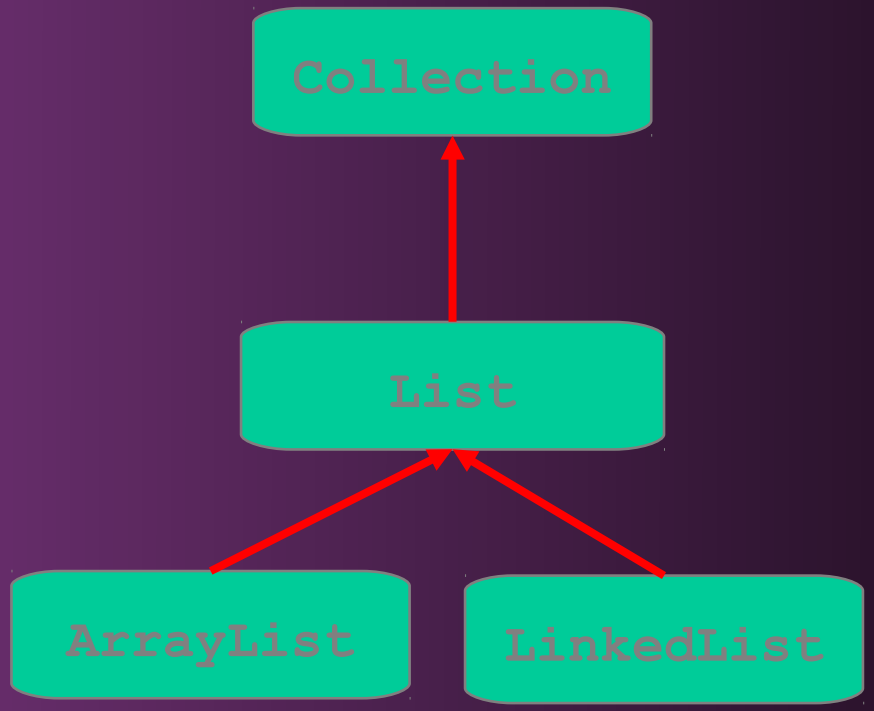
2D arrays

- Topic 9 Task 5

```
tab1[0][0] = "France";
tab1[1][0] = "Spain";
tab1[2][0] = "Australia";
tab1[3][0] = "USA";
tab1[0][1] = "Euro";
tab1[1][1] = "Euro";
tab1[2][1] ="AUS$";
tab1[3][1] ="US$";
```

List interface

- A list is an ordered collection
- It can contain duplicate elements, accessible via an index



ArrayList class

- Shortcomings of Arrays
  - Difficult to add/remove elements –shifting elements
  - Array size is fixed
- ArrayList overcomes these shortcomings
  - import java.util.ArrayList;
  - ArrayList<String> al = new ArrayList<String>();
  - Automatically resized
  - Random insertion/deletion of new elements
  - Supports any data type –primitives, instantiable classes
  - Elements identified by integer index

ArrayList methods

- Topic 9 Task 7
  - Illustration –handling primitive types
- add() –adding elements to ArrayList
- size() –compute size of ArrayList
- get() –access ArrayList element
- set() –change value of element in ArrayList
- remove() –delete element from ArrayList
- clear() –delete all elements from ArrayList

ArrayList complex objects

- Topic 9 Task 8
  - Illustration –handling complex objects
  - ArrayList<MyRecord> aLR = new ArrayList<MyRecord>();
  - Inner classes
    - ArrayList methods implementation
    - Complex object creation
  - Practice
    - Add methods for removing records

LinkedList class

- Shortcomings of ArrayList
  - Efficiency
    - Insertion and deletion involves movement of data
- LinkedList overcomes this
  - Collection of objects organised in a list, with pointers to following& preceding objects
  - Uses non-contiguous memory locations
  - LinkedList<String> al = new LinkedList<String>();

LinkedList class

- Disadvantages
  - Requires more memory for management process
  - Accessing elements is slower
- Topic 9 Task 9
  - Illustration
- Methods
  - Similar to those used in ArrayList, however, LinkedList class has additional methods –not necessary for the purposes of this module

Arrays | ArrayLists | LinkedLists

- Choosing appropriate data structure
  - Arrays
    - Problem spaces involves a number of independent variables
  - ArrayLists
    - Problem space requires direct access to elements and VERY few insertions and deletions
  - LinkedLists
    - Problem space requires a lot of insertions and deletions... busy Web applications

Lecture Outcomes

Today we have covered:

- Data structures
  - ArrayList
  - LinkedList

- Questions?