Multi-dimensional Arrays
Multidimensional Arrays

- A 2D array is just an array of arrays.
- (A 3D array is an array of 2D arrays ;)

![Diagram of multidimensional arrays](image-url)
7.6 – Two-Dimensional Arrays

- A two-dimensional array can be thought of as a table of elements, with rows and columns.
- In Java a two-dimensional array is an array of arrays.
- It is declared by specifying the size of each dimension separately.
  ```java
  int[][] table = new int[5][6];
  ```
- An array element is referenced using two index values.
  ```java
  value = table[3][5]
  ```
- The array stored in any one row can be specified using one index!
## 7.6 – Two-Dimensional Arrays

<table>
<thead>
<tr>
<th>Expression</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>int[][]</td>
<td>array of integer arrays (2D array)</td>
</tr>
<tr>
<td>table[3]</td>
<td>int[]</td>
<td>array of integers</td>
</tr>
<tr>
<td>table[3][2]</td>
<td>int</td>
<td>Integer</td>
</tr>
<tr>
<td>table[3].length</td>
<td>int</td>
<td>Integer (= 6)</td>
</tr>
<tr>
<td>table.length</td>
<td>int</td>
<td>Integer (= 5)</td>
</tr>
</tbody>
</table>
public class TwoDArray {

    public static void main (String[] args) {
        int[][] table = new int[5][10];

        // Load the table with values
        for (int row=0; row < table.length; row++)
            for (int col=0; col < table[row].length; col++)
                table[row][col] = row * 10 + col;

        // Print the table
        for (int row=0; row < table.length; row++) {
            for (int col=0; col < table[row].length; col++)
                System.out.print (table[row][col] + "\t");
            System.out.println();
        }
    }
}
Arrays of Objects

They are like Arrays of Strings.
Only instead of Strings we have other Objects...
A basic and powerful way to manage a collection of items

- CS230 is about software structures that allow us to manage collections of data
- We already saw that when we used an array of String objects:
  ```java
  String[] words = new String[5];
  ```
- We could use an array to store and manage a collection of any other objects, e.g. of CD objects, of Student objects, of Rectangle objects, of ...
```java
import java.text.NumberFormat;

/**
 * Represents a compact disc (an ancient technology for recording sound)
 * @author Java Foundations
 */

class CD {
    private String title, artist;
    private double cost;
    private int tracks;

    /**
     * Create a new CS with the specified information
     * @param name album title
     * @param singer artist name
     * @param price cost of CD in floating point
     * @param numTracks number of tracks on CD
     */
    public CD (String name, String singer, double price, int numTracks) {
        title = name;
        artist = singer;
        cost = price;
        tracks = numTracks;
    }
}

toString() method, getters, and setters are omitted
```
A collection of CD objects

collection[0] = new CD("Storm Front", "Billy Joel", 14.95, 10);
collection[1] = new CD("Come On Over", "Shania Twain", 14.95, 16);
collection[2] = new CD("Soundtrack", "Les Miserables", 17.95, 33);
7.3 – Arrays of Objects

- A UML diagram for the Tunes program
/**
 * Demonstrates the use of an array of objects
 * @author Java Foundations
 /**
 public class Tunes {
 /**
 * Creates a CDCollection object and adds some CDs to it. Prints
 * reports on the status of the collection.
 */
 public static void main (String[] args) {
 CDCollection music = new CDCollection ();

 music.addCD("Storm Front", "Billy Joel", 14.95, 10);
 music.addCD("Come On Over", "Shania Twain", 14.95, 16);
 music.addCD("Soundtrack", "Les Miserables", 17.95, 33);
 music.addCD("Graceland", "Paul Simon", 13.90, 11);

 System.out.println (music);

 music.addCD("Double Live", "Garth Brooks", 19.99, 26);
 music.addCD("Greatest Hits", "Jimmy Buffet", 15.95, 13);

 System.out.println (music);
 }
import java.text.NumberFormat;

/**
 * Represents a collection of compact discs.
 * @author Java Foundations
 */

public class CDCollection {
    private CD[] collection;
    private int count;
    private double totalCost;

    /**
     * Constructor: Creates an initially empty collection.
     */
    public CDCollection () {
        collection = new CD[100];
        count = 0;
        totalCost = 0.0;
    }
}
7.3 – CDCollection.java

```java
/**
 * Adds a CD to the collection, increasing the size of the
 * collection if necessary.
 * @param name album title
 * @param artist artist's name
 * @param price cost of CD in floating point
 * @param tracks number of tracks on CD
 */

public void addCD (String title, String artist, double cost, int tracks) {
    if (count == collection.length)
        increaseSize();

    collection[count] = new CD (title, artist, cost, tracks);
    totalCost += cost;
    count++;
}
```
/**
* @return string reporting on the CD collection
*/

public String toString() {
    NumberFormat fmt = NumberFormat.getCurrencyInstance();

    String report = "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n";
    report += "My CD Collection\n\n";

    report += "Number of CDs: " + count + "\n";
    report += "Total cost: " + fmt.format(totalCost) + "\n";
    report += "Average cost: " + fmt.format(totalCost/count);

    report += "\n\nCD List:\n\n";

    for (int cd = 0; cd < count; cd++)
        report += collection[cd].toString() + "\n";

    return report;
}
/**
 * Increases the capacity of the collection by creating a larger array and copying the existing collection into it.
 */

private void increaseSize () {
    CD[] temp = new CD[collection.length * 2];

    for (int cd = 0; cd < collection.length; cd++)
        temp[cd] = collection[cd];

    collection = temp;
}
Counting Letter Frequencies

Counting the number of occurrences of each letter in a sentence
Counting letters in a sentence

Enter a sentence:
To be, or not to be: that is the question: Whether 'tis nobler in the mind to suffer the slings and arrows of outrageous fortune, or to take arms against a sea of troubles, and by opposing end them?

A: 0 a: 11
B: 0 b: 5
C: 0 c: 0
D: 0 d: 4
E: 0 e: 17
F: 0 f: 5
G: 0 g: 4
H: 0 h: 7
I: 0 i: 8
J: 0 j: 0
K: 0 k: 1
L: 0 l: 3
M: 0 m: 3
N: 0 n: 12
O: 0 o: 18
P: 0 p: 2
Q: 0 q: 1
R: 0 r: 11
S: 0 s: 13
T: 1 t: 18
U: 0 u: 6
V: 0 v: 0
W: 1 w: 1
X: 0 x: 0
Y: 0 y: 1
Z: 0 z: 0

Non-alphabetic characters: 45

- Given a sentence (as a String), how do you count how many times each letter appears?
  - (Hint: You can access any char using charAt(i) )
- Where would you keep track of all the counters?
### ASCII control characters

<table>
<thead>
<tr>
<th>char</th>
<th>ASCII</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>NULL</td>
<td>(Null character)</td>
</tr>
<tr>
<td>01</td>
<td>SOH</td>
<td>(Start of Header)</td>
</tr>
<tr>
<td>02</td>
<td>STX</td>
<td>(Start of Text)</td>
</tr>
<tr>
<td>03</td>
<td>ETX</td>
<td>(End of Text)</td>
</tr>
<tr>
<td>04</td>
<td>EOT</td>
<td>(End of Trans.)</td>
</tr>
<tr>
<td>05</td>
<td>ENQ</td>
<td>(Enquiry)</td>
</tr>
<tr>
<td>06</td>
<td>ACK</td>
<td>(Acknowledgement)</td>
</tr>
<tr>
<td>07</td>
<td>BEL</td>
<td>(Bell)</td>
</tr>
<tr>
<td>08</td>
<td>BS</td>
<td>(Backspace)</td>
</tr>
<tr>
<td>09</td>
<td>HT</td>
<td>(Horizontal Tab)</td>
</tr>
<tr>
<td>10</td>
<td>LF</td>
<td>(Line feed)</td>
</tr>
<tr>
<td>11</td>
<td>VT</td>
<td>(Vertical Tab)</td>
</tr>
<tr>
<td>12</td>
<td>FF</td>
<td>(Form feed)</td>
</tr>
<tr>
<td>13</td>
<td>CR</td>
<td>(Carriage return)</td>
</tr>
<tr>
<td>14</td>
<td>SO</td>
<td>(Shift Out)</td>
</tr>
<tr>
<td>15</td>
<td>SI</td>
<td>(Shift In)</td>
</tr>
<tr>
<td>16</td>
<td>DLE</td>
<td>(Data link escape)</td>
</tr>
<tr>
<td>17</td>
<td>DC1</td>
<td>(Device control 1)</td>
</tr>
<tr>
<td>18</td>
<td>DC2</td>
<td>(Device control 2)</td>
</tr>
<tr>
<td>19</td>
<td>DC3</td>
<td>(Device control 3)</td>
</tr>
<tr>
<td>20</td>
<td>DC4</td>
<td>(Device control 4)</td>
</tr>
<tr>
<td>21</td>
<td>NAK</td>
<td>(Negative acknowledge)</td>
</tr>
<tr>
<td>22</td>
<td>SYN</td>
<td>(Synchronous idle)</td>
</tr>
<tr>
<td>23</td>
<td>ETB</td>
<td>(End of trans. block)</td>
</tr>
<tr>
<td>24</td>
<td>CAN</td>
<td>(Cancel)</td>
</tr>
<tr>
<td>25</td>
<td>EM</td>
<td>(End of medium)</td>
</tr>
<tr>
<td>26</td>
<td>SUB</td>
<td>(Substitute)</td>
</tr>
<tr>
<td>27</td>
<td>ESC</td>
<td>(Escape)</td>
</tr>
<tr>
<td>28</td>
<td>FS</td>
<td>(File separator)</td>
</tr>
<tr>
<td>29</td>
<td>GS</td>
<td>(Group separator)</td>
</tr>
<tr>
<td>30</td>
<td>RS</td>
<td>(Record separator)</td>
</tr>
<tr>
<td>31</td>
<td>US</td>
<td>(Unit separator)</td>
</tr>
</tbody>
</table>

### ASCII printable characters

<table>
<thead>
<tr>
<th>char</th>
<th>ASCII</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>space</td>
<td></td>
</tr>
</tbody>
</table>
import java.util.Scanner;

public class LetterCount
{
    public static void main (String[] args)
    {
        final int NUMCHARS = 26;

        Scanner scan = new Scanner (System.in);

        int[] upper = new int[NUMCHARS];
        int[] lower = new int[NUMCHARS];

        char current;  // the current character being processed
        int other = 0;  // counter for non-alphabetics

        (more...)
System.out.println("Enter a sentence:");
String line = scan.nextLine();

// Count the number of times each letter occurs
for (int ch = 0; ch < line.length(); ch++) {
    current = line.charAt(ch);
    if (current >= 'A' && current <= 'Z')
        upper[current-'A']++;
    else
        if (current >= 'a' && current <= 'z')
            lower[current-'a']++;
        else
            other++;
}

// Print the results
for (int letter=0; letter < upper.length; letter++) {
    System.out.print((char) (letter + 'A'));
    System.out.print(":");
    System.out.print(" 	" + upper[letter]);
    System.out.print(" 	" + (char) (letter + 'a'));
    System.out.println(":");
    System.out.println("Non-alphabetic characters: " + other);
}
}