Recall Arrays

- Arrays: sequences of elements from the same base type
  - `int[] a;` // array of ints
  - `Date[][] d;` // array of references to Dates
- Base type may be:
  - Primitive (i.e. `int`)
  - Reference (i.e. `Date`, other objects)
- Arrays are also objects.
- Notice the similarities:
  - Arrays created using `new`
  - Array elements stored on heap
  - Array variables store references to space on the heap

Allocation of Space

- Syntax for allocating space for the 1st level array:
  - `char[][] a;` // Array of char arrays
  - `a = new char[3][];` // Create array of 3 arrays
- Syntax for allocating space for the 2nd level of arrays:
  - `a[0] = new char[4];` // Create array of 4 char
  - `a[1] = new char[6];` // Create array of 6 char
Example

```java
char[][] a;
a = new char[3][];
a[0] = new char[4];
a[1] = new char[6];
a[2] = new char[3];
a[1][3] = 'a';
```

- This array has two dimensions: rows, columns
- This kind of array is called **ragged** because the rows are of unequal length

Questions

```java
char[][] a;
a = new char[3][];
a[0] = new char[4];
a[1] = new char[6];
a[2] = new char[3];
```
- What does `a[1][2] = 'x';` do? Set element in row 2, column 3 to 'x'
- What does `a.length` return? 3
- What does `a[1].length` return? 6
- What type is `a`?
  - a reference to an array of array references
- What Type is `a[0]`?
  - a reference to an array of characters
- What type is `a[0][0]`?
  - a character

Initializers

- In one dimension:
  ```java
  char[] a;
a = new char[3];
a[0] = {'a','b','c','d'};
a[1] = {'x','y','z'};
a[2] = {'m','n'};
  ```
- In two dimensions:
  ```java
  char[][] a = {{'a','b','c','d'}, {'x','y','z'}, {'m','n'}};
  ```
Rectangular Arrays

- Often we want 2-dimensional arrays in which rows have the same length
  - Tables
  - Matrices
- Java has a special short-hand syntax for creating rectangular arrays
  ```java
  int[ ][ ] a = new int[2][4]; // 2 rows, 4 cols
  ```
  Equivalent to:
  ```java
  int[ ][ ] a = new int[2][ ];
  a[0] = new int[4];
a[1] = new int[4];
  ```
- The short-hand takes care of allocating each row, initializing each cell in each row

Example

```java
int[ ][ ] a = new int[2][4];
```
- Note each cell is initialized to default value (0)
- Each row is a 1-dim array

2-D Arrays of Objects

- The array name when indexed with two indexes is of that indicated type
  - Of Strings:
    ```java
    String[ ][ ] s = new String[4][2];
s[0][0] = “Fred”;
s[1][1] = “Jane”;
    ```
  - Of Cats:
    ```java
    Cat[ ][ ] c = new Cat[4][2];
c[0][0] = new Cat(“Fred”);
c[1][1] = new Cat(“Jane”);
    ```