

- Previous Lecture:
 - Polymorphism
- Today's Lecture:
 - **Object** class
 - **Abstract** (reading in textbook)
 - 2-d array
- Reading: Sec 8.9 (8.10 optional)

May 1, 2007 Lecture 27 3

The Object class

If a class is not explicitly defined to be the child of an existing class, it is assumed to be the child of the **Object** class

⇒ All classes are derived from the **Object** class

```

class Room
  is the same as
class Room extends Object
    
```

May 1, 2007 Lecture 27 11

abstract class

- A placeholder in a class hierarchy that represents a generic concept
- **Cannot be instantiated**
- Modifier: `abstract`

```
public abstract class Geometry
```
- Can contain abstract methods

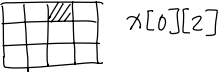

```
public abstract double Area();
```
- Subclasses of abstract classes will "fill out" these abstract methods

May 1, 2007 Lecture 27 15

2-d arrays

- A 1-d array is a *list* of values (references)
- A 2-d array is a *table* of values (references)
- Each component of a 2-d array is referenced using **two** index values
- A 2-d array in Java is really a **1-d array of 1-d arrays** (i.e., an array of objects)
 - Orientation (row, column) is only how we *choose* to visualize the "table"
 - **Convention: row-major**

May 1, 2007 Lecture 27 17



May 1, 2007 Lecture 27 18

Multi-dimensional array

- Can have as many dimensions as you want
- A 2-d array is a 1-d array of 1-d arrays. Each 1-d array has its own constant **length** ⇒ you can have a **ragged** (not rectangular) 2-d array.

May 1, 2007 Lecture 27 20

Creating a 2-d array

1. Declare a reference `x` for a 2-d integer array
2. Create a 2-by-3 integer array `y`
3. Create the following array:

2	4	6
8	1	3

May 1, 2007 Lecture 27 22

Accessing a 2-d array

- Given a reference `x` that points to a 2-d `int` array. . .
1. What is its height (# of rows)?
 2. What is `x[0]` ?
 3. What is the length of the first row?
 4. How to access last element in the second row?
 5. How to access last element in last row?

May 1, 2007 Lecture 27 26

Example 1

Given a 2-d integer array `x`, calculate the sum of all entries in the array. Assume the array is rectangular.

May 1, 2007 Lecture 27 27

What if . . .

- The array is ragged instead of rectangular? Suppose all rows exist but the rows have different lengths.
- Not all rows exist and the existing rows have different lengths?

May 1, 2007 Lecture 27 29

Example 2

Given a 2-d array `m`, re-order the rows such that the row with the **highest row sum** is the first row.



May 1, 2007 Lecture 27 32

What is the algorithm?

Given a 2-d array `m`, re-order the rows such that the row with the **highest row sum** is the first row.

```
//calculate row sums

//find index of row with max sum

//swap row of max sum with row 0
```

May 1, 2007 Lecture 27 33