Announcements

- No class tomorrow
- Review this Friday July 5
- Midterm #2 next Friday July 12
  - Covers everything through next Wednesday
    - Cumulative but focused on new material
  - More study questions posted

Follow ups

- More string parsing for p4
  - Expression example
- Array refreshers
  - String[] args example
  - Sieve example

2D array syntax

```
//Allocate rectangular 2 x 4 grid
int[][] array2D = new int[2][4];
//Save the upper left element in upperLeft
int upperLeft = array2D[0][0];
//Store the value 50 in lower right
array2D[1][3] = 50;
```

upperLeft | 0 |

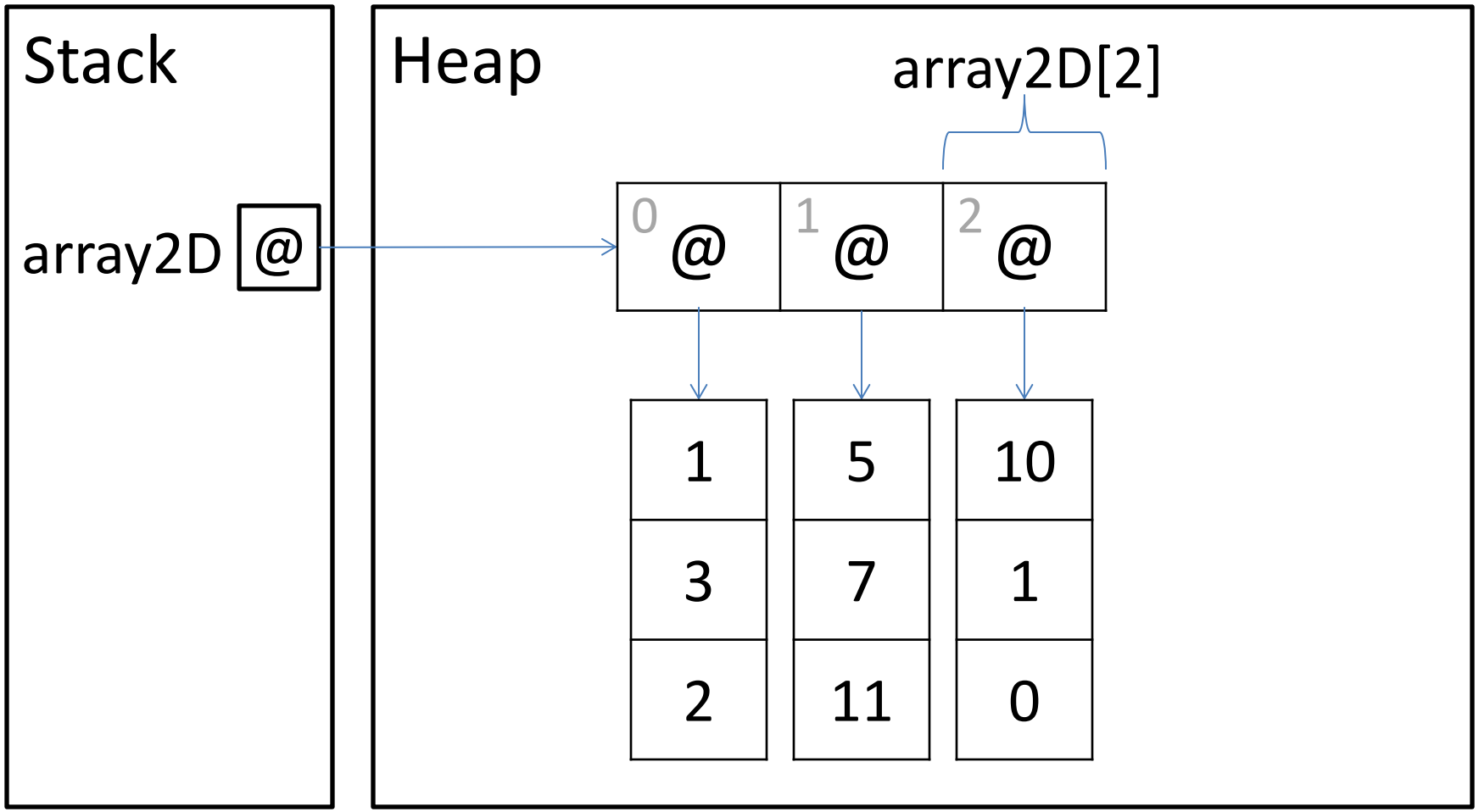
indices	0	1	2	3
0	0	0	0	0
1	0	0	0	50

2D Arrays

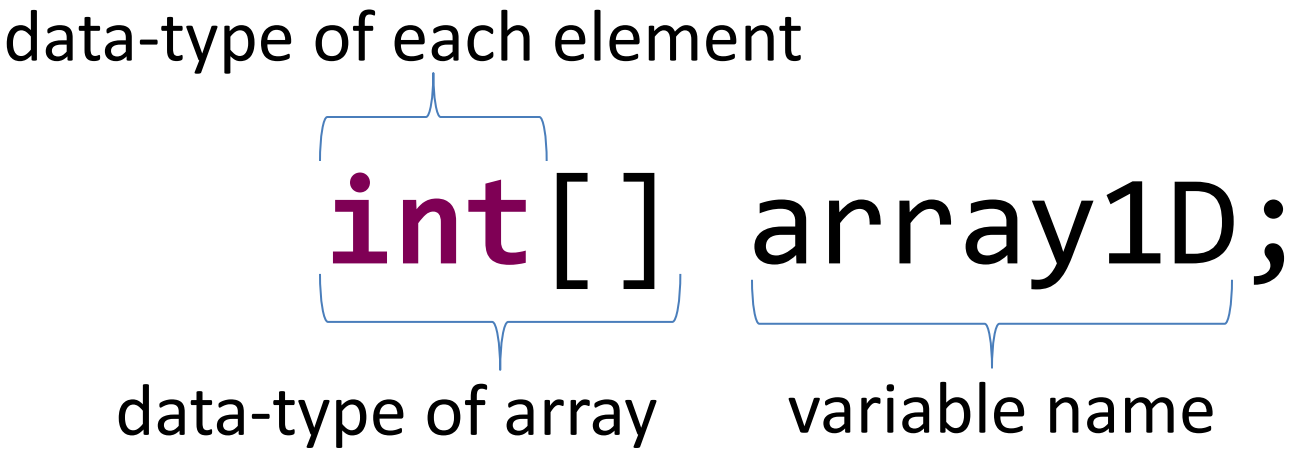
- A 2D array is an “Array of arrays”
  - array2D is of type int[][]
  - array2D[1] is of type int[]
  - array2D[1][3] is of type int
- Array2DBasics example

Arrays of arrays (2D)

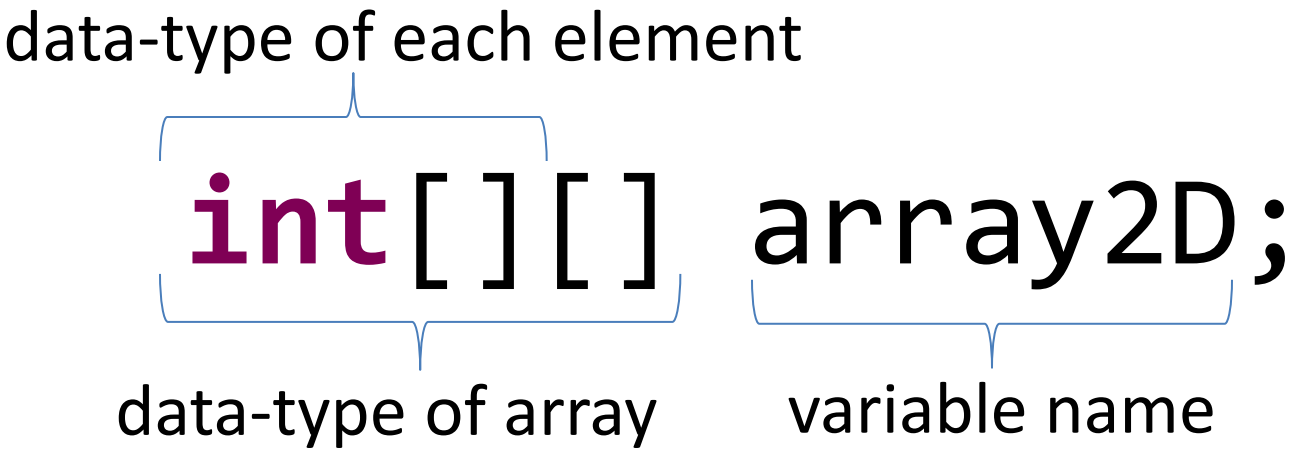
- Matrix Example, Game of life example



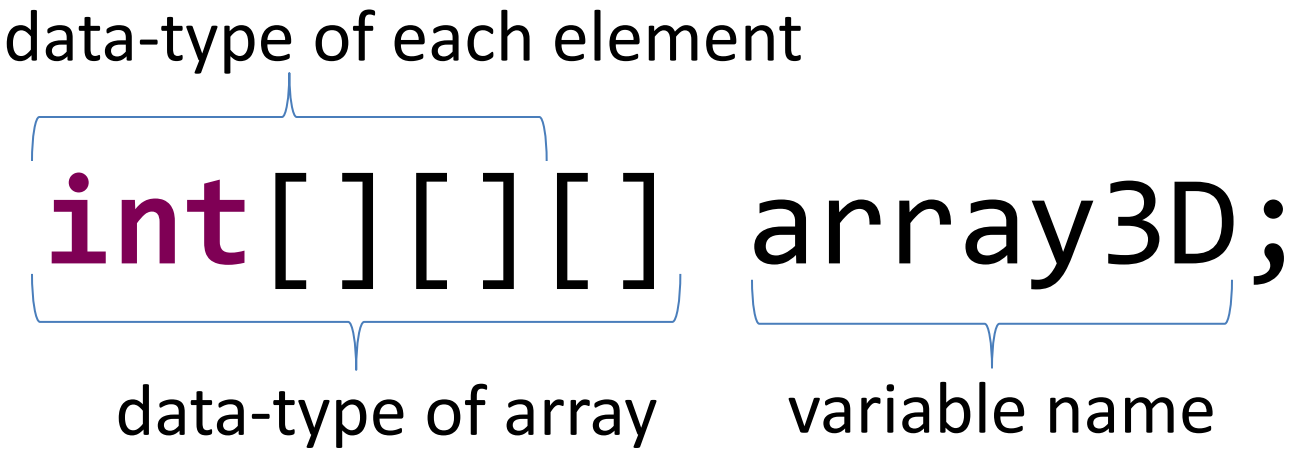
Higher dimensions



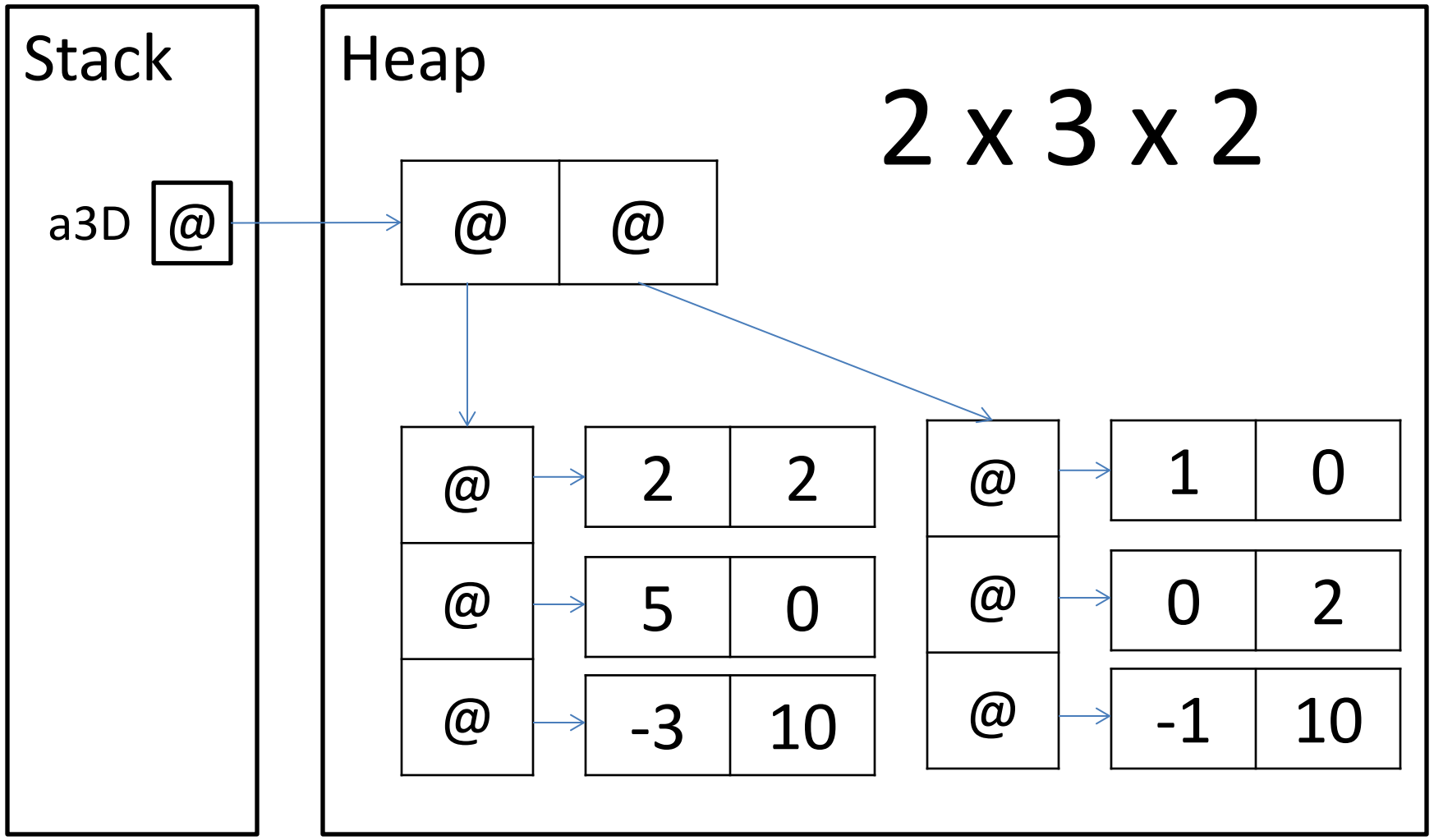
Higher dimensions



Higher dimensions



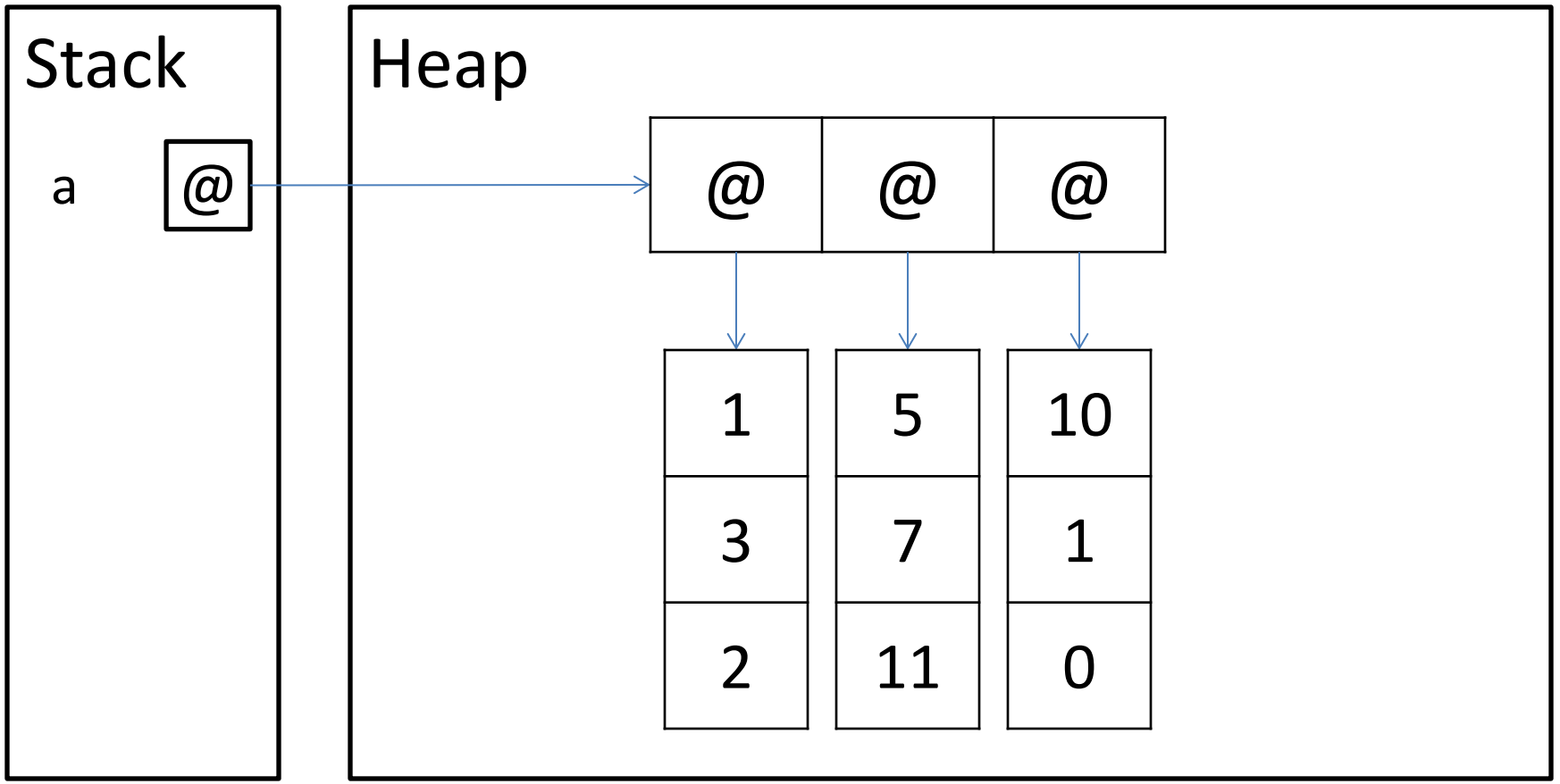
Arrays of arrays of arrays



“Ragged” Arrays

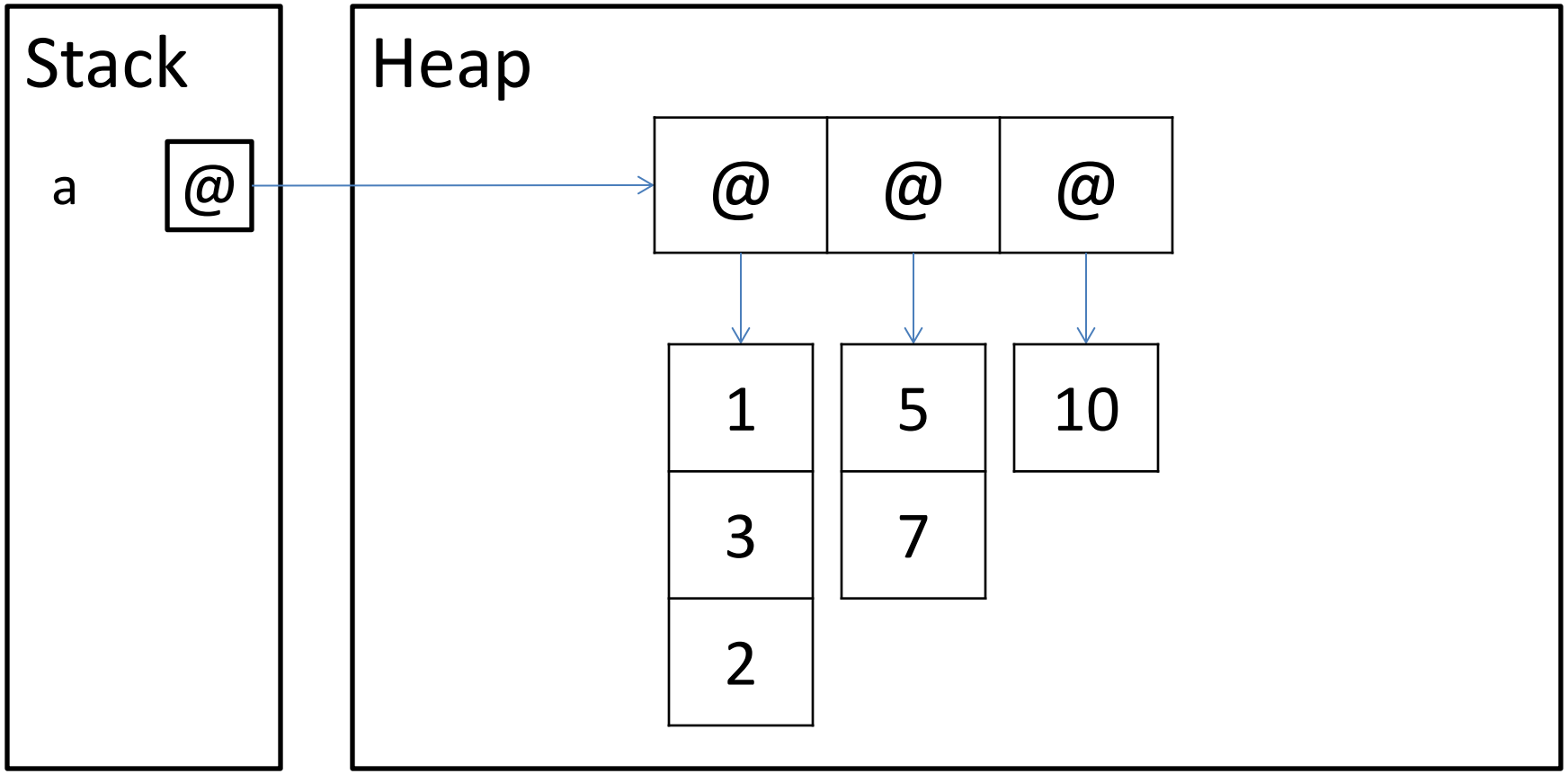
- Not every row of a (>=)2D array needs to be the same length.
  - array2D[0] is a reference to data-type int[]
  - array2D[1] is a reference to data-type int[]
  - No restriction that they must be the same length

Not ragged



Ragged

- Pascal's triangle example



Built-in array support

- Java.lang.System

```
public static void arraycopy(Object src, int srcPos,
                             Object dest, int destPos,
                             int length) {…}
```

- Java.util.Arrays
  - Copying
  - Sorting
  - Searching
  - toString

Immutable Objects

- Immutable objects can never be changed once they are constructed
  - Strings
  - Primitive wrapper classes (Integers etc.)
  - IntLists and CharLists?
- Advantages: Primitive-like.
  - Simple, bug-resistant, tamper-resistant
  - Never need to be copied
- Immutable classes vs enums?

Immutable design

- Classes are immutable *by design*
- Designing immutable classes:
  - Declare the class with the final keyword (more on this later)
  - Fields must be private or final
  - No setter methods
  - All mutable reference fields must be copied
    - When initialized
    - When returned by getters
  - Multi-threading considerations

Immutability examples

- Position/IntVector example revisited
- Privacy leaks: Primate/Dog example
  - If not designed carefully, private variables might still be accessible.
- No deep copies for immutable objects: StringArray example

StringBuffer

- Mutable version of String
  - Can replace characters
  - Can insert characters
  - Can increase length
- MyStringBuffer example