ICS 141 Programming with Objects

Jessica Maistrovich

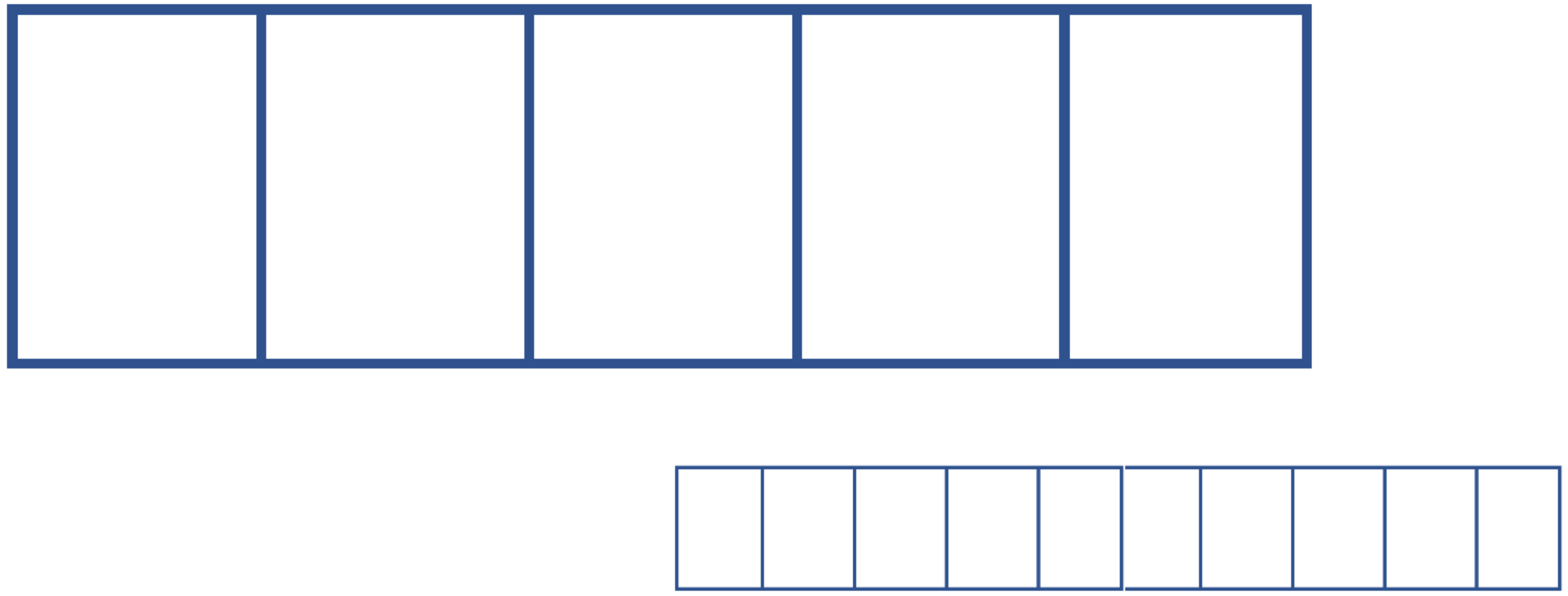
Metropolitan State University

Arrays

a.k.a. cubby shelves

What do you need to know to make a shelf?

What does it hold?
How many things can it hold?



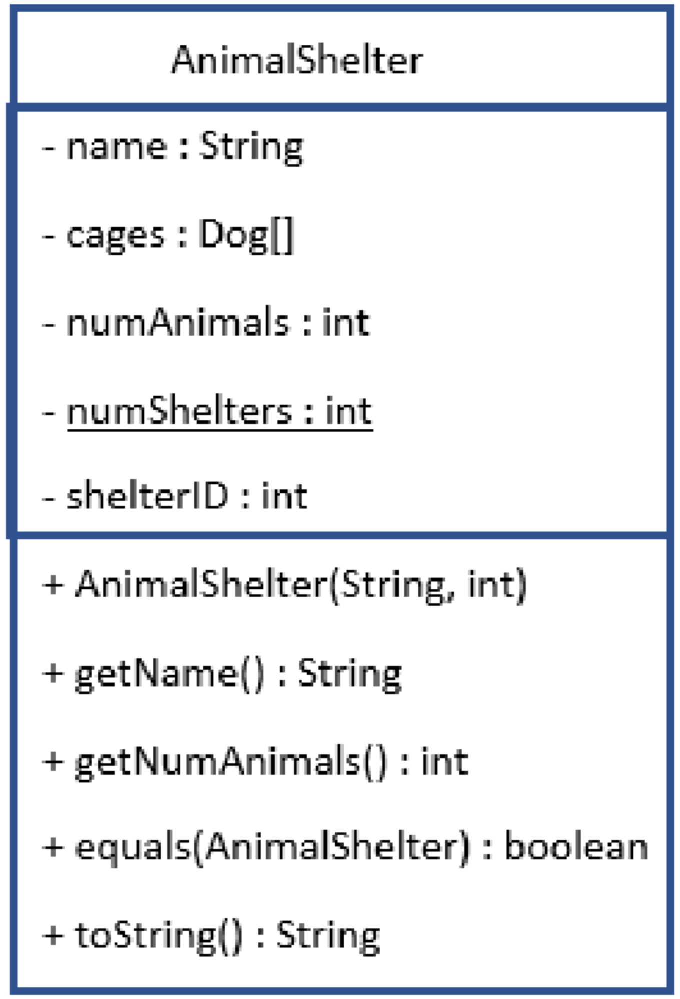
Array Declaration

```
__________[ ]    _______________  = new ____________ [______];
data-type        variable name          data-type     size
```

```
McDonalds[] listOfMcDonalds = new McDonalds[5];
int[] listOfIntegers = new int[10];
```

Try it!

Create a class in the DogApplication project based on the following UML:



AnimalShelter

- name : String
- cages : Dog[]
- numAnimals : int
- <u>numShelters : int</u>
- shelterID : int

+ AnimalShelter(String, int)
+ getName() : String
+ getNumAnimals() : int
+ equals(AnimalShelter) : boolean
+ toString() : String

- Constructor: accepts a String that represents the name of the shelter, and an int that represents the maximum number of animals this shelter can hold (use this to create the Dog array). Set the number of animals to zero. Use the number of shelters to create a unique id for the shelter.
- equals: two animal shelters are considered equal if they have the same id.
- toString: return the shelter id, shelter name, and a header as described here:

```
Shelter number <shelter number> named <shelter name> has the following animals available:
Name   Age
***************************
```

For example:

```
Shelter number 4 named Animal Humane Society has the following dogs available:
Name   Age
```

Thoughts on arrays

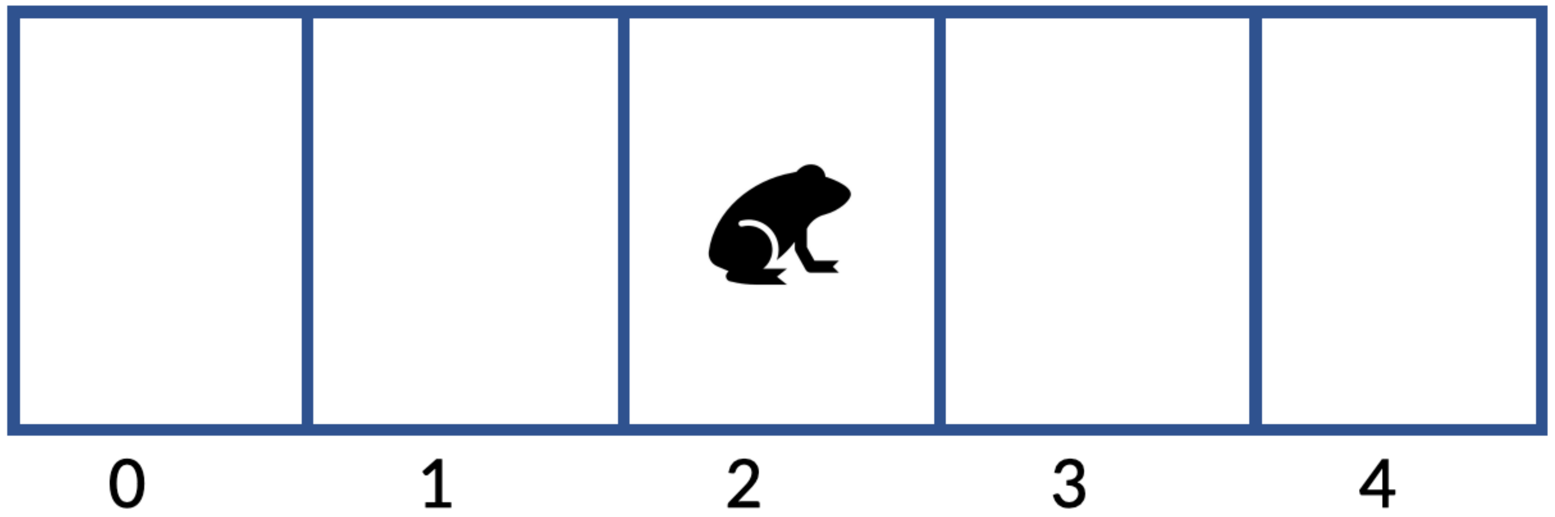
- Can the length of the array ever change?

- Can the number of things stored in the array change?
  - numItems variable

- How do we access items in the array?

Array indexing

Starts at zero.



Access it by saying the thing behind door number two. Notice first thing is behind door number zero.

If this array is stored in variable list, then the frog is at list[2]

Using Elements of the Array

- Array elements are used in the same way as variables

```
myArr[0] =  10*5 + 3;
System.out.println(myArr[0] );
```

- The element's position can be calculated using any mathematical operations:

```
int x = 3, y = 5;
myArr[y]= 10;
myArr[y-2] = 20;
myArr[y-x] = myArr[1] - myArr[2];
```

Very Important Note: Array Bounds

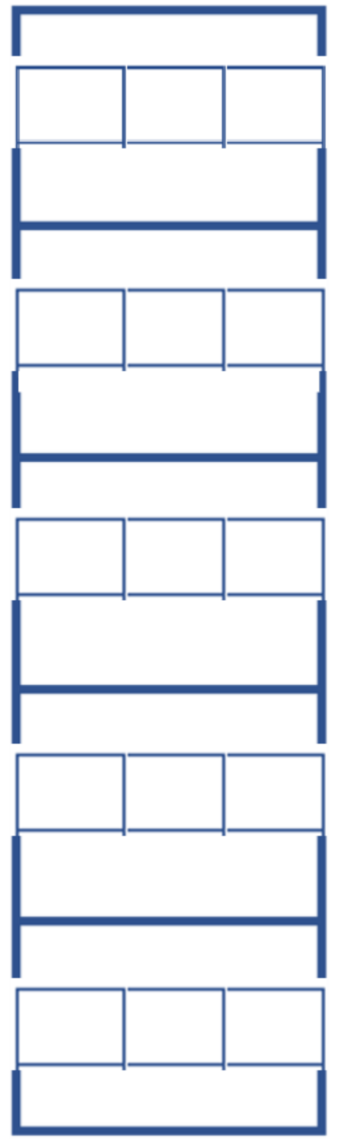
- When writing a loop to go over all elements in the array, make sure of the following:
  - Array subscript never goes below 0
  - Maximum array subscript should be < `length`

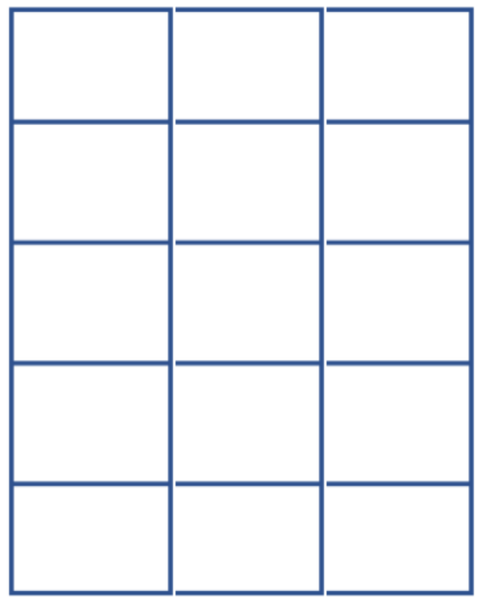
`int[] data = new int[10];`	
`data[ -1 ]`	always illegal
`data[ 10 ]`	illegal (given the above declaration)
`data[ 1.5 ]`	always illegal
`data[ 0 ]`	always OK if the array exists
`data[ 9 ]`	OK (given the above declaration)
`data[ j ]`	can't tell without more information (depends on the current value of j)

Long line of code what is happening?

```
array[i].getName().equals(array[i-1].getName())
```

Shelves can hold shelves – double array





```
int[][] test = new int[5][3];
```

5 rows 3 columns

Common things we do with arrays

- Modify (add/remove)
- Fill the array
- Print
- Search for something
- Accumulate values/compare values

Filling an array

Filling an array

- Create an empty array
  - Null
  - zero
- Create with values
- Put values in with a loop
- Put values in one at a time (add an element)

*Sentinel values or actual values

When create an array - filled with default value

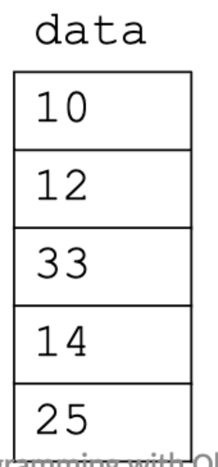
- Null value – means nothing is there. Can’t tell a non-existent thing to do something.
- 0 for primitive data types

- Empty string is not null. A String object that contains no characters is still an object. Such an object is called an **empty string**. It is similar to having a blank sheet of paper (different from having no paper at all).

Array Initialization using a list of values

```
int[]  data = {10,12,33,14,25 };
```

- The Java compiler creates an array where the size equals to the number of entries in the initializers list.



Moving through an array

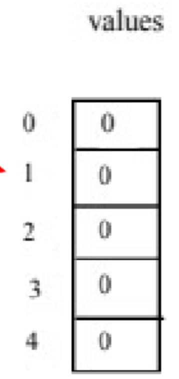
Same overall idea regardless of what “action” is being performed: print, search, fill with a loop, accumulate. Still looking at all the cubbies in the array.

Example: what is the output of the following piece of code?

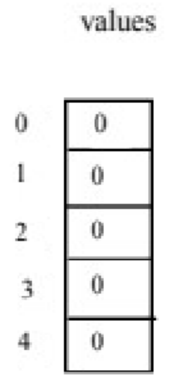
```
int[] values = new int[5];

for (int i = 1; i < 5; i++) {
        values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i++) {   System.out.println(i+"\t"+values[i]);
}
```

Declare an array variable. An array is created in memory and all values are initialized to 0

```
int[] values = new int[5];

for (int i = 1; i < 5; i=i+1) {
          values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {   System.out.println(i+"\
          t"+values[i]);
}
```

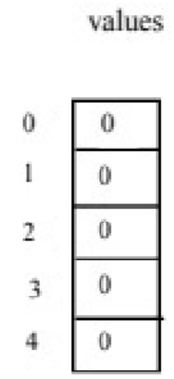


```
int[] values = new int[5];

for (int i = 1; i < 5; i=i+1) {
          values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {   System.out.println(i+"\
          t"+values[i]);
}
```



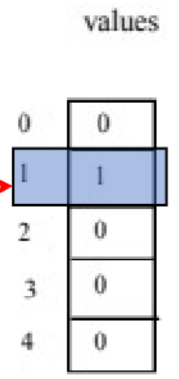
i (=1) is less than 5

```
int[] valu   = new int[5];

for (int i = 1; i < 5; i=i+1) {
          values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {   System.out.println(i+"\
          t"+values[i]);
}
```



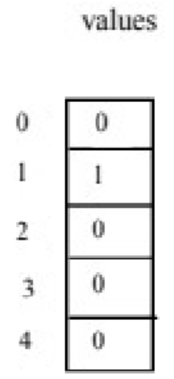
After this line is executed, `values[1] = 1`

```
int[] values = new int[5];

for (int i = 1; i < 5; i=i+1) {
      values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {    System.out.println(i+"\
            t"+values[i]);
}
```



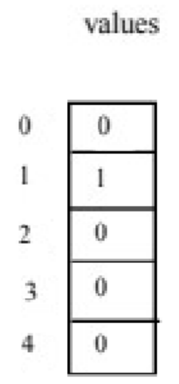
After `i=i+1`, i = 2

```
int[] values = new int[5];

for (int i = 1; i < 5; i=i+1) {
          values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {   System.out.println(i+"\
          t"+values[i]);
}
```



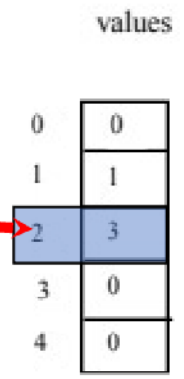
i (= 2) is less than 5

```
int[] values = new int[5];

for (int i = 1; i < 5; i=i+1) {
          values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {   System.out.println(i+"\
          t"+values[i]);
}
```



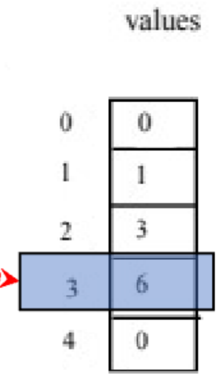
After this line is executed,
`values[2] = 3` (2 + 1)

```
int[] values = new int[5];

for (int i = 1; i < 5; i=i+1) {
      values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {   System.out.println(i+"\
            t"+values[i]);
}
```



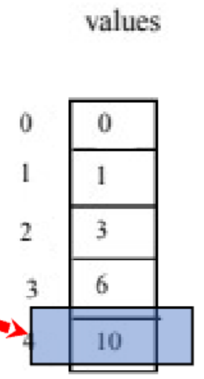
After this line, `values[3] = 6` (3 + 3)

```
int[] values = new int[5];

for (int i = 1; i < 5; i=i+1) {
        values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {   System.out.println(i+"\
          t"+values[i]);
}
```



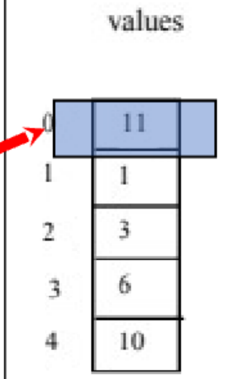
After this, `values[4] = 10` (4 + 6)

```
int[] values = new in

for (int i = 1; i < 5; i=i 1) {
    values[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {    System.out.println(i+"\
        t"+values[i]);
}
```

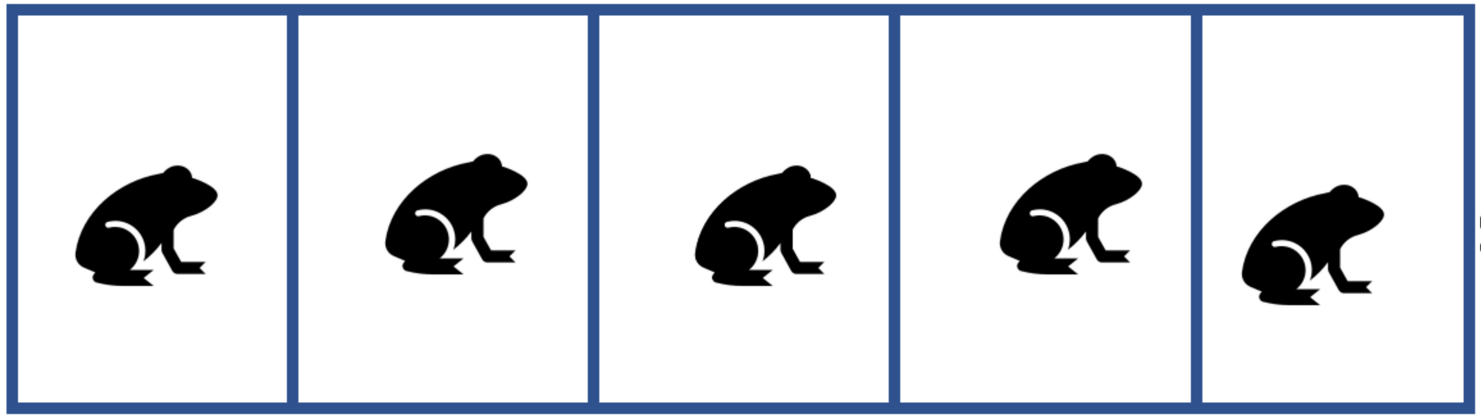


After this line, `values[0] =11` (1 + 10)

```
int[] values = t[5];

for (int i = < 5; i=i+1) {
        ues[i] = i + values[i-1];
}

values[0] = values[1] + values[4];

for (int i = 0; i < 5; i=i+1) {   System.out.println(i+"\
        t"+values[i]);
}
```



Adding to array



0 1 2 3 4

Length of array = 5

NumItems	Location
0	0
1	1
2	2
3	3
4	4
5	

- Array class has a length variable and it is public

Try it! (part 1)

- Create the following two methods in the AnimalShelter class.
  - addAnimal(): this method accepts an Dog object and puts it into the dog array. Don’t forget to increment the number of animals.
  - addAnimal(): an overloaded method that accepts the inputs needed to create a Dog instance. Instantiate a dog and add it to the array.

Try it! (part 2)

- Modify the toString in the Dog class so that it prints the name and age separated by a tab.
- Modify the toString for the AnimalShelter class so that it prints all the animals in the shelter (don’t delete what is already there, add to it). **You are going to be calling the toString of the Dog class to help with this.
- Create a driver class. Inside the main method, create a shelter, add at least one animal, and print the shelter. Experiment with adding more dogs to your shelter and printing the shelter.