

08-0: **Arrays**

- ArrayLists are not part of Java proper
 - Library class
 - Created using lower-level Java construct: Array

08-1: **Arrays**

- Arrays are like a stripped-down ArrayList
 - Arrays are objects (like ArrayLists)
 - Access elements using [] notation (like python lists)
 - Can't do anything fancy: no negative indices, no ranges
 - Need to declare the size of the array when it is created
 - Can't change the size of an array once it is created
 - Get the length of the array using public length instance variable

08-2: **Arrays**

- Two ways to declare arrays:

```
<typename>[] variableName;  
<typename> variableName[];
```

- Examples:

```
int A[];    // A is an array of integers  
int[] B;    // B is an array of integers  
String C[]; // C is an array of strings
```

08-3: **Arrays: New**

- Like all other objects, Arrays are stored on the heap
- `int A[]` just allocates space for a pointer
- Need to call `new` to create the actual array

```
new <type>[<size>]
```

08-4: **Arrays: New**

- Show contents of memory after each line:

```
int A[];  
int B[];  
A = new int[10];  
B = new int[5];  
A[7] = 4;  
B[2] = 5;  
B[5] = 13;    /// RUNTIME ERROR!
```

08-5: **Arrays**

```
void foo()
{
    int i;
    int A[];

    A = new int[5]

    for (i = 0; i < 5; i++)
    {
        A[i] = i;
    }
}
```

Trace through, show memory

08-6: **Arrays: Copying**

```
int A[] = new int[SIZE];
int B[] = new int[SIZE];

// Code to store data in B
A = B;
```

- What do you think this code does?
- What happens when we assign *any* object to another object?

08-7: **Arrays: Copying**

```
int A[] = new int[SIZE];
int B[] = new int[SIZE];

// Code to store data in B
A = B;
```

- How could we copy the data from B into A
- (A and B should point to different memory locations, have same values)

08-8: **Arrays: Copying**

```
int A[] = new int[SIZE];
int B[] = new int[SIZE];

// Code to store data in B
for (int i = 0; i < B.length; i++)
{
    A[i] = B[i];
}
```

08-9: **Array: Copying**

```
int A[] = new int[5];
int B[] = new int[5];
int C[];

for (int i = 0; i < 5; i++)
    A[i] = i;

for (int i = 0; i < 5; i++)
    B[i] = A[i];

C = A;

B[2] = 10;
C[2] = 15;
```

08-10: Arrays of Objects

- We can have arrays of objects, as well as arrays of integers

```
...
Point pointArray[] = new Point[10];
pointArray[3].setX(3.4);
```

- What happens?
 - (refer to Java documentation for Point objects)

08-11: Arrays of Objects

```
Point pointArray[] = new Point[10];

for (int i = 0; i < 10; i++)
{
    pointArray[i] = new Point();
}
```

- Is this OK?

08-12: Arrays of Objects

```
Point pointArray[] = new Point[10];

for (int i = 0; i < 10; i++)
{
    pointArray[i] = new Point(i, i);
}
```

- Note that you can pass an integer to a parameter that expects a double (but not the other way around!)

08-13: Arrays of Objects

```
Point pointArray[] = new Point[10];

for (int i = 0; i < 10; i++)
{
    pointArray[i] = new Point(i, i);
}
```

- How would you calculate the average x value of all elements in the array?

08-14: Arrays of Objects

- How would you calculate the average x value of all elements in the array?

```
Point pointArray[] = new Point[10];
// Fill in pointArray
//
double sum = 0.0;
for (int i = 0; i < pointArray.length; i++)
{
    sum = sum + pointArray[i].getX();
}
sum = sum / pointArray.length;
```

08-15: Arrays of Objects

- Arguments to Java program: What is this args variable?

```
public static void main(String args[])
{
}
}
```

- Array of strings of command line arguments
- java MyProgram arg1 arg2
- Using Run Dialog in Eclipse

08-16: Arrays of Objects

- Arguments to Java program

```
public static void main(String args[])
{
    for (int i = 0; i < args.length; i++)
    {
        System.out.println(args[i]);
    }
}
```

08-17: 2D Arrays

- We can create 2D arrays as well as 1D arrays
 - Like matrices

- 2D array is really just an array of arrays

08-18: 2D Arrays

```
int x[][];    // Declare a 2D array
int[][] y;    // Alternate way to declare 2D array

x = new int[5][10]; // Create 50 spaces
y = new int[4][4];  // create 16 spaces
```

08-19: 2D Arrays

```
int x[][];          // Declare a 2D array
x = new int[5][5];  // Create 25 spaces

x[2][3] = 11;
x[3][3] = 2;
x[4][5] = 7;    // ERROR! Index out of bounds
```

08-20: 2D Arrays

- How would we create a 9x9 array, and set every value in it to be 3?

08-21: 2D Arrays

- How would we create a 9x9 array, and set every value in it to be 3?

```
int board[][];
board = new int[9][9];
for (int i = 0; i < 9; i++)
    for (int j = 0; j < 9; j++)
        board[i][j] = 3;
```

08-22: Using Arrays

- Need to declare array size before using them
- Don't always know ahead of time how big our array needs to be
- Allocate more space than we need at first
- Maintain a second size variable, that has the number of elements in the array we actually care about
- Classes that use arrays often will have an array instance variable, and a size instance variable (how much of the array is used)

08-23: Using Arrays

```
public class StringArrayList
{
    String data[];
    int listSize;

    public StringArrayList()
    {
        data = new String[10];
        listSize = 0;
    }
    /// other methods
}
```

08-24: Using Arrays

```
public class StringArrayList
{
    String data[];
    int listSize;

    int size()
    {
        // Fill me in!
    }
    // other methods
}
```

08-25: Using Arrays

```
public class StringArrayList
{
    String data[];
    int listSize;

    int size()
    {
        return listSize;
    }
    // other methods
}
```

08-26: Using Arrays

```
public class StringArrayList
{
    String data[];
    int listSize;

    void add(String newString)
    {
        // Fill me in!
    }
    // other methods
}
```

08-27: Using Arrays

```
public class StringArrayList
{
    String data[];
    int listSize;

    void add(String newString)
    {
        data[listSize] = newString;
        listSize++;
    }

    // other methods
}
```

08-28: Using Arrays

```
public class StringArrayList
{
    String data[];
    int listSize;

    void add(int index, String newString)
    {
        // Fill me in!
    }
    // other methods
}
```

08-29: Using Arrays

```
public class StringArrayList
{
    String data[];
    int listSize;

    void add(int index, String newString)
    {
        for (int i = listSize; i > index; i--)
        {
            data[i] = data[i-1];
        }
        data[index] = newString;
    }
    // other methods
}
```

08-30: **Midterm in 1 Week**

- Topics
 - Java Syntax
 - Using {}, knowing where ;'s go
 - Components of a class
 - data members (instance variables)
 - constructor(s)
 - methods

08-31: **Midterm in 1 Week**

- Topics
 - Methods
 - Return type
 - Parameter list
 - Method body
 - Calling methods
 - Pass by value (Objects can be tricky here!)

08-32: **Midterm in 1 Week**

- Topics
 - Variables
 - Variable Declaration, specifying type
 - Primitive types versus reference types (objects)
 - NullPointerException

08-33: **Midterm in 1 Week**

- Topics
 - Object instantiation
 - When to use new
 - What happens when new is called

08-34: **Midterm in 1 Week**

- Topics
 - Statements vs. Expressions
 - = vs ==
 - Relational operators <, >, <=, >=, ==, !=
 - Boolean operators &&, ||, !

08-35: **Midterm in 1 Week**

- Topics

- Conditionals (if statements)
 - else and else if
 - “dangling else”

08-36: Midterm in 1 Week

- Topics
 - Iteration (for, while, do while)
 - control variable
 - initialization, condition, and update

08-37: Midterm in 1 Week

- Topics
 - Arrays and ArrayLists
 - Declaring arrays and ArrayLists
 - Getting and Setting values in arrays and ArrayLists

08-38: Midterm in 1 Week

- Topics
 - Strings
 - compareTo method
 - == and what it does

08-39: Midterm in 1 Week

- Sample Questions:
 - Write a method that calculates x^n , for integers x and n
 - `int power(int x, int n)`

08-40: Midterm in 1 Week

- Sample Questions:
 - Write a method that takes as input an ArrayList of integers, and returns the sum of all elements in the list
 - `int sum(ArrayList<int> list)`

08-41: Midterm in 1 Week

- Sample Questions:
 - Given two strings S1 and S2, what is the difference between the following two expressions
 - `S1 == S2`
 - `S1.compareTo(S2) == 0`

08-42: Midterm in 1 Week


```
class Odd {
    public void foo(int x) {
        x = x + 1;
        System.out.println("In foo: x = " + x);
    }
    public void bar(int x) {
        x = x + 1;
        System.out.println("In bar: x = " + x);
        foo(x);
        System.out.println("In bar: x = " + x);
    }

    public static void main(String args[]) {
        Odd o = new Odd();
        int x = 3;
        System.out.println("In Main: x = " + x);
        o.bar(x);
        System.out.println("In Main: x = " + x);
    }
}
```

08-43: Midterm in 1 Week

```
class Driver {
    public void doubleInt(int x) {
        x = x * 2;
    }
    public void doubleArray(int A[]) {
        for (int i = 0; i < A.length; i++)
            A[i] = A[i] * 2;
    }

    public static void main(String args[]) {
        Driver d = new Driver();
        int Array[] = new int[3];
        for (int i = 0; i < 3; i++)
            Array[i] = 1;
        int x = 1;
        d.doubleInt(x);
        d.doubleArray(Array);
        System.out.println("X = " + x);
        for (int i = 0; i < 3; i++)
            System.out.printon("Array[i] = " + Array[i]);
    }
}
```