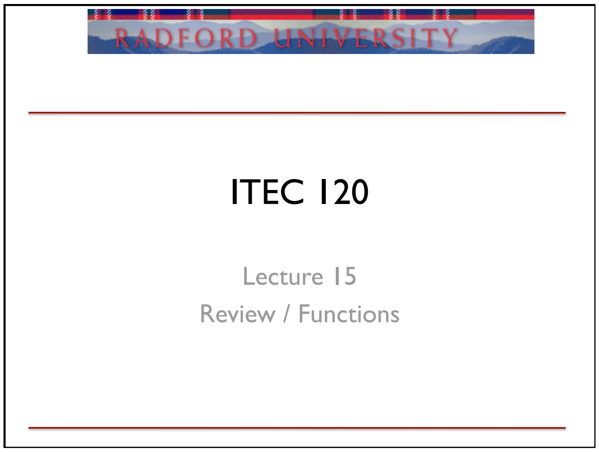
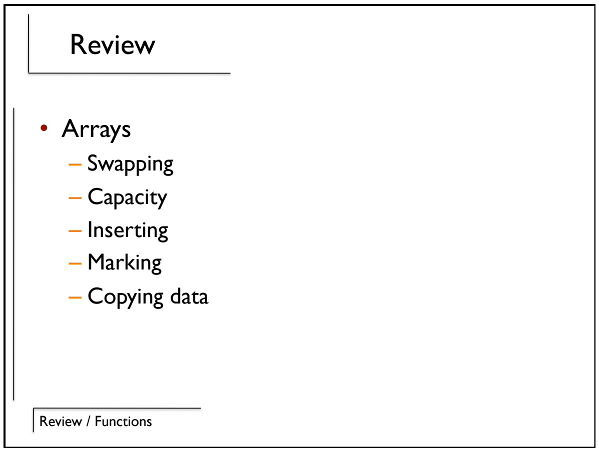


RADFORD UNIVERSITY

ITEC 120

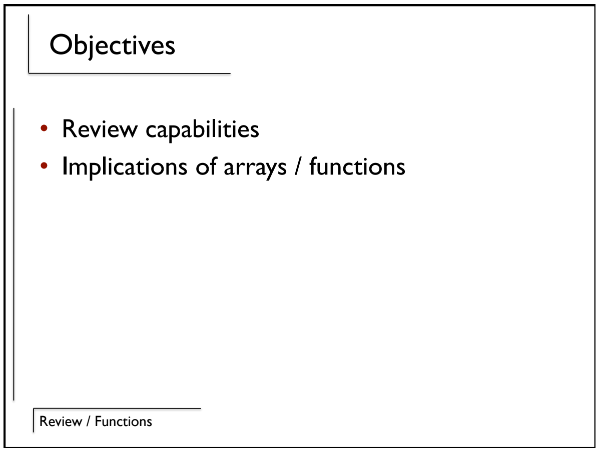
Lecture 15

Review / Functions



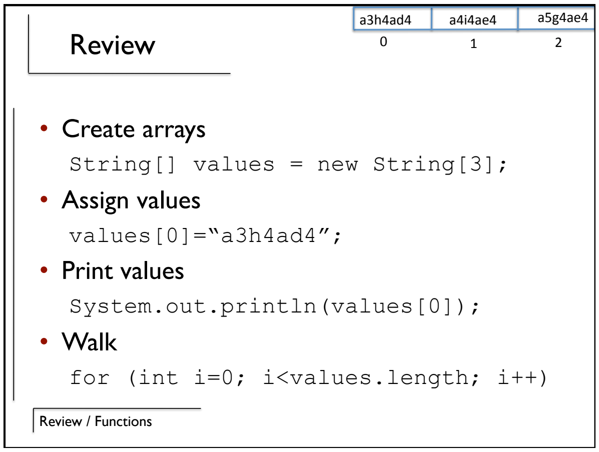
Review

- Arrays
  - Swapping
  - Capacity
  - Inserting
  - Marking
  - Copying data



Objectives

- Review capabilities
- Implications of arrays / functions



Review

a3h4ad4	a4i4ae4	a5g4ae4
0	1	2

- Create arrays

```
String[] values = new String[3];
```

- Assign values

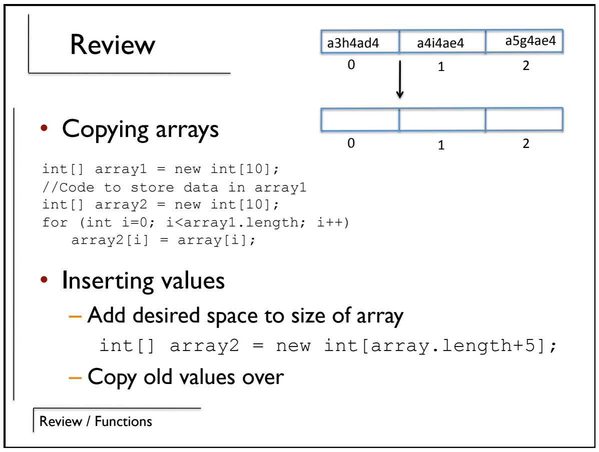
```
values[0]="a3h4ad4";
```

- Print values

```
System.out.println(values[0]);
```

- Walk

```
for (int i=0; i<values.length; i++)
```



Review

a3h4ad4
a4i4ae4
a5g4ae4
0
1
2
0
1
2

- Copying arrays

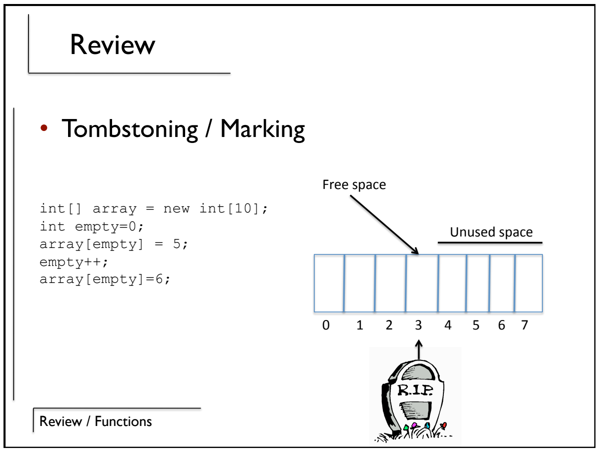
```
int[] array1 = new int[10];
//Code to store data in array1
int[] array2 = new int[10];
for (int i=0; i<array1.length; i++)
   array2[i] = array[i];
```

- Inserting values
  - Add desired space to size of array

```
int[] array2 = new int[array.length+5];
```

  - Copy old values over

Review / Functions



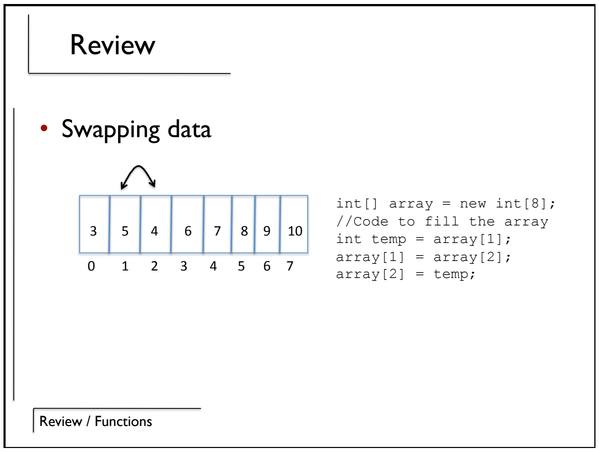
Review

- Tombstoning / Marking

```
int[] array = new int[10];
int empty=0;
array[empty] = 5;
empty++;
array[empty]=6;
```

Free space
Unused space
0 1 2 3 4 5 6 7
R.I.P.

Review / Functions



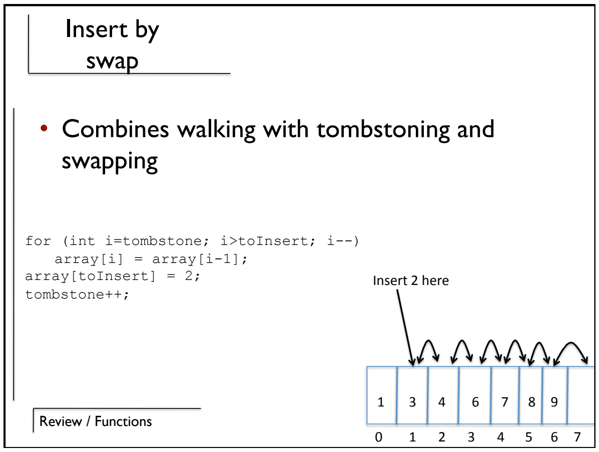
Review

- Swapping data

3 5 4 6 7 8 9 10
0 1 2 3 4 5 6 7

```
int[] array = new int[8];
//Code to fill the array
int temp = array[1];
array[1] = array[2];
array[2] = temp;
```

Review / Functions



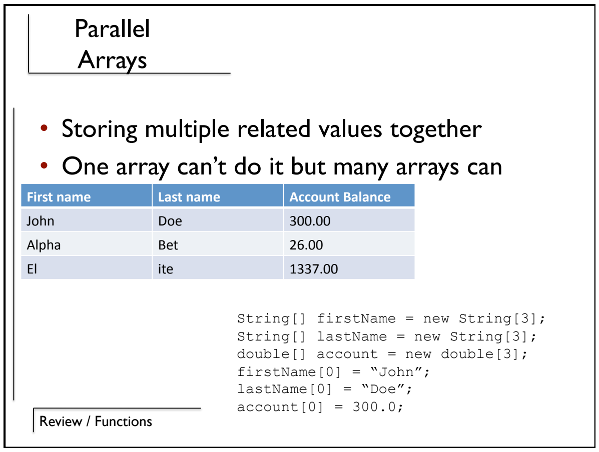
Insert by swap

- Combines walking with tombstoning and swapping

```
for (int i=tombstone; i>toInsert; i--)
   array[i] = array[i-1];
array[toInsert] = 2;
tombstone++;
```

Insert 2 here
1 3 4 6 7 8 9
0 1 2 3 4 5 6 7

Review / Functions



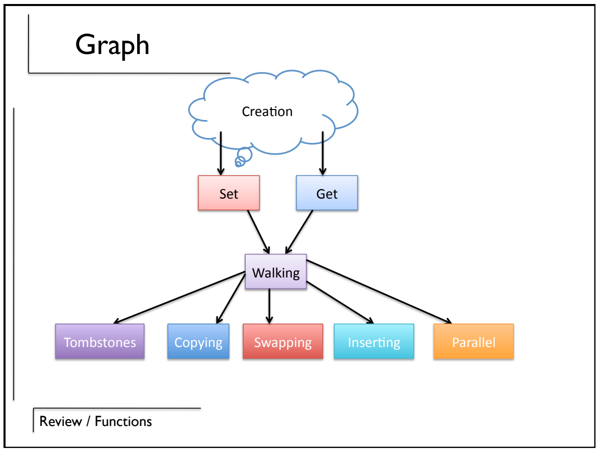
Parallel Arrays

- Storing multiple related values together
- One array can’t do it but many arrays can

First name	Last name	Account Balance
John	Doe	300.00
Alpha	Bet	26.00
El	ite	1337.00

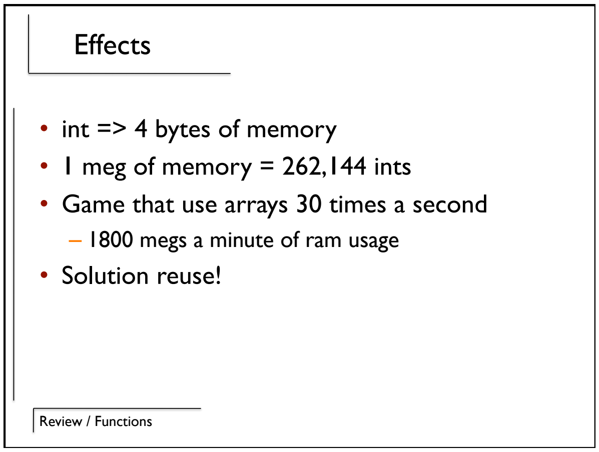
```
String[] firstName = new String[3];
String[] lastName = new String[3];
double[] account = new double[3];
firstName[0] = “John”;
lastName[0] = “Doe”;
account[0] = 300.0;
```

Review / Functions



Graph

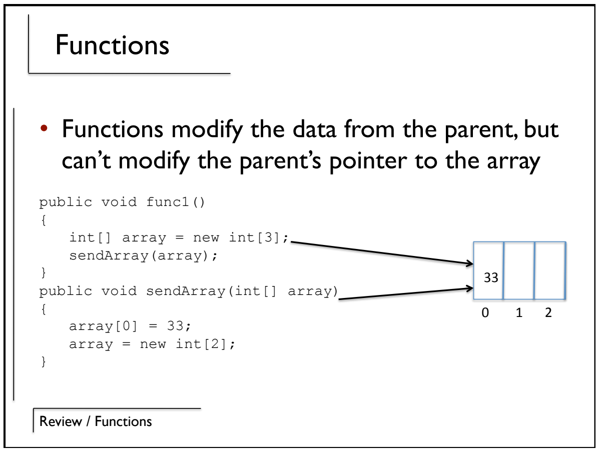
Review / Functions



Effects

- int => 4 bytes of memory
- 1 meg of memory = 262,144 ints
- Game that use arrays 30 times a second
  - 1800 megs a minute of ram usage
- Solution reuse!

Review / Functions



Functions

- Functions modify the data from the parent, but can’t modify the parent’s pointer to the array

```
public void func1()
{
    int[] array = new int[3];
    sendArray(array);
}
public void sendArray(int[] array)
{
    array[0] = 33;
    array = new int[2];
}
```

Review / Functions

Example

- Pass array to function
- Modify in function
- Check results

Review / Functions

Pass by reference

- Complex data is pass by reference
  - You get the real data
- Simple data is pass by value
  - You get a copy

Review / Functions

Comments

- Explain how your program works
- Scenario
  - Write code and don't look at it for 3 years

```
//Ignore until end of line
/*
    Ignore until till you get =>  */
```

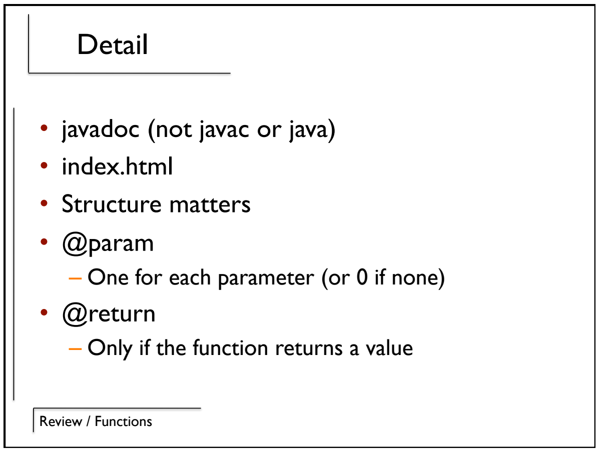
Review / Functions

Javadoc

- Use the right format, get a webpage

```
/**
* A description of the example function.
* The first sentence is used in the method table,
* The others are added to the detail section.
* @param one An integer sent to the example function
* @param two The weight of the apple
* @return The value
**/
public int myExample(int one, double apple)
{
    //Inline commenting here
}
```

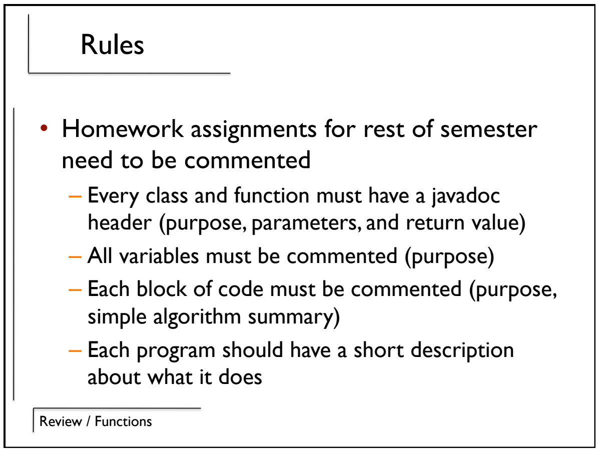
Review / Functions



Detail

- javadoc (not javac or java)
- index.html
- Structure matters
- @param
  - One for each parameter (or 0 if none)
- @return
  - Only if the function returns a value

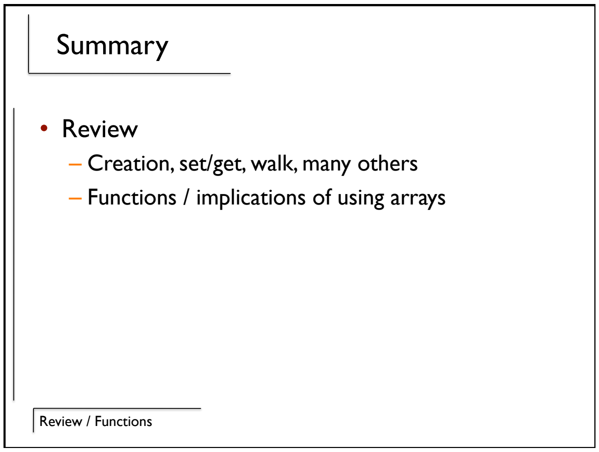
Review / Functions



Rules

- Homework assignments for rest of semester need to be commented
  - Every class and function must have a javadoc header (purpose, parameters, and return value)
  - All variables must be commented (purpose)
  - Each block of code must be commented (purpose, simple algorithm summary)
  - Each program should have a short description about what it does

Review / Functions



Summary

- Review
  - Creation, set/get, walk, many others
  - Functions / implications of using arrays

Review / Functions