CS 367 Announcements
Thursday, February 5, 2015

Homework h1 due by 10 pm tomorrow, February 6th
- make sure your file is plain text
- make sure you use the name h1.txt
- submit to your `in` handin directory
- remember that late work is not accepted

Homework h2 assigned late tomorrow

Program p1 due 10 pm Sunday, February 15th (get started now!)

Handin directories have been created.
Assignment questions? Post on Piazza or see a TA during lab consulting hours.

Last Time

Exceptions Review
- throwing
- handling
- execution
- throws and checked vs. unchecked
- defining
- practice

Today

Programmer's Memory Model
Primitive vs. Reference Types
- assignment
- parameter passing

ADTs vs. Data Structures
Chains of Linked Nodes
Submitting Work

Next Time

Read: continue *Linked Lists*
Chains of Linked Nodes
- Listnode class
- practice

Java Visibility Modifiers

Programmer's Memory Model

Call Stack

The place where activation record for method call reside!
- the space for local variables and parameter variables

Birth	when declared
Death	when the variable falls out of scope

Heap

Arrays & objects
Need a reference variables to access

Birth	when using "new"
Death	when no longer referenced (garbage collected)

Static Data

where Literals reside
where class variable (static) reside

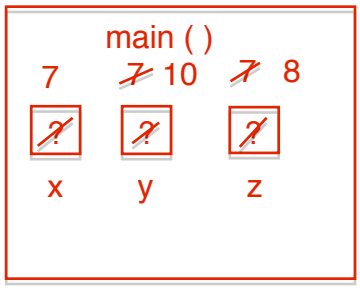
Birth	At the program execution
Death	At the end program's execution

Primitive vs. Reference Types: Assignment

assume in the main( ) method

Primitives

```
int x, y, z; // not init to 0!
x = 7;
y = x;
z = x;
y = 10;
z = 8;
```

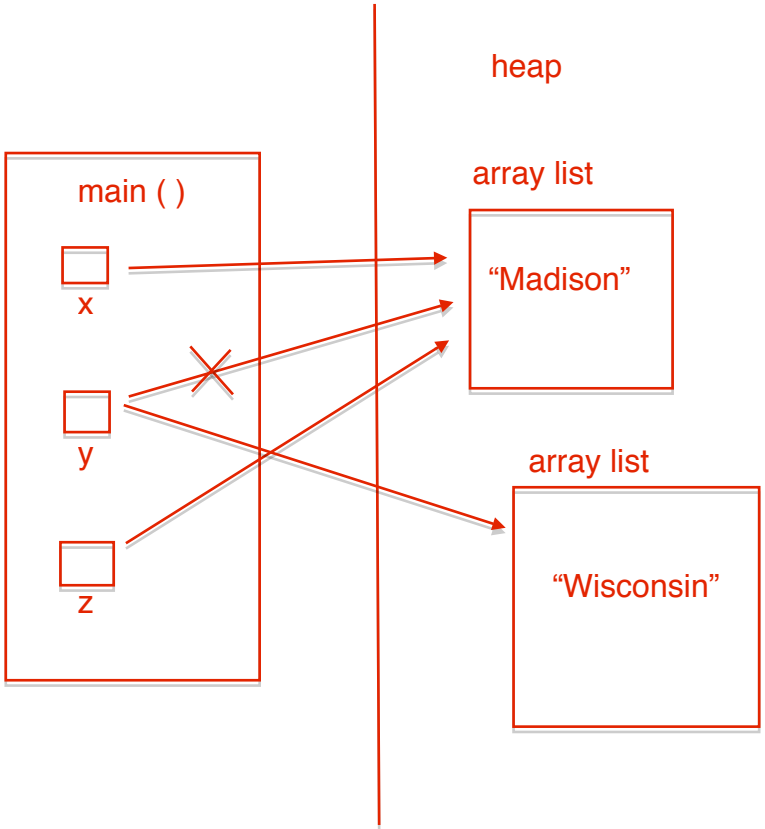


for primitive ( )
copy the assigned value

assume in the main ( ) method

References

```
ArrayList<String> x, y, z;
x = new ArrayList<String>();
y = x;
z = x;
y = new ArrayList<String>();
z.add("Madison");
y.add("Wisconsin");
```



x and y a alias, or references
to the same objects

this DOES NOT copy the object!
We just copy the ADDRESS

Assignments of references
Resulting in aliases

Primitive vs. Reference Types: Parameter Passing

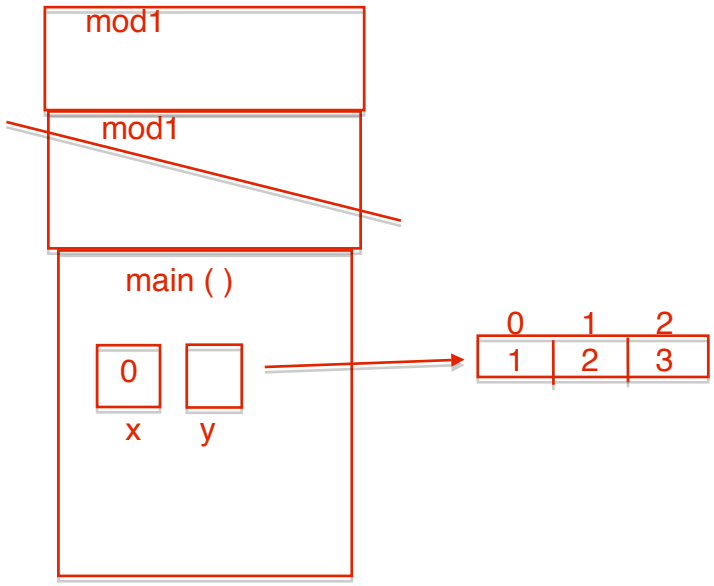
Primitives

Given:

```
void mod1(int x) {
   x = 7;
}
```
this does not affect x in the main

Execute (assume in main):

```
int x = 1;
int[] y = {1, 2, 3};
mod1(x);
mod1(y[2]);
```



References

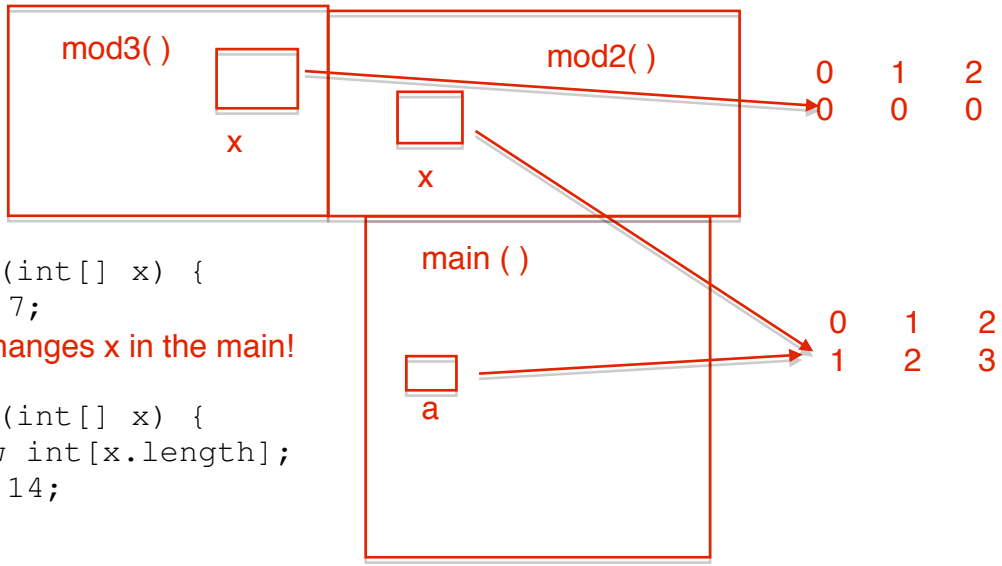
Given:

```
void mod2(int[] x) {
   x[0] = 7;
}
```
This does changes x in the main!

```
void mod3(int[] x) {
   x = new int[x.length];
   x[0] = 14;
}
```

Execute (assume in main):

```
int[] a = {1, 2, 3};
mod2(a);
mod3(a);
```



ADTs vs. Data Structures

Abstract Data Type (ADT)

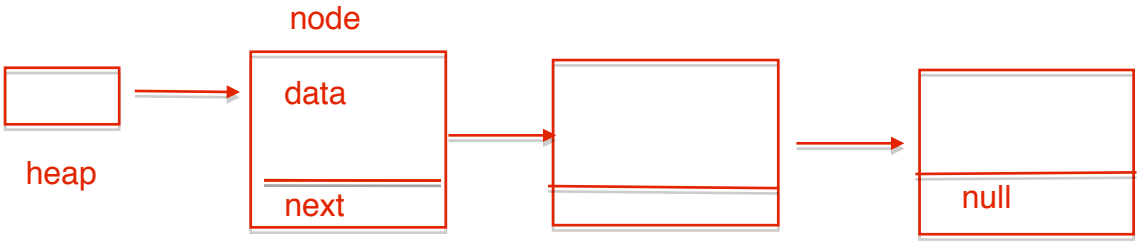
- A description of what can be done on a collect to items
- ADT is coded as interfaces
- it requires implementing class, which codes how the ADT works

Data Structure (DS)

A construct that you built within the implementing class
that is used to store the collection of items
- e.g. - Array
- Chain of Nodes

Chain of Linked Nodes a data structure

The Data Structure



A chain of node

Array vs. Chain of Nodes

Goal