



CMSC 131

Object-Oriented Programming I

Privacy Leaks, Copying Objects

**Dept of Computer Science
University of Maryland College Park**

This material is based on material provided by Ben Bederson, Bonnie Dorr,
Fawzi Emad, David Mount, Jan Plane

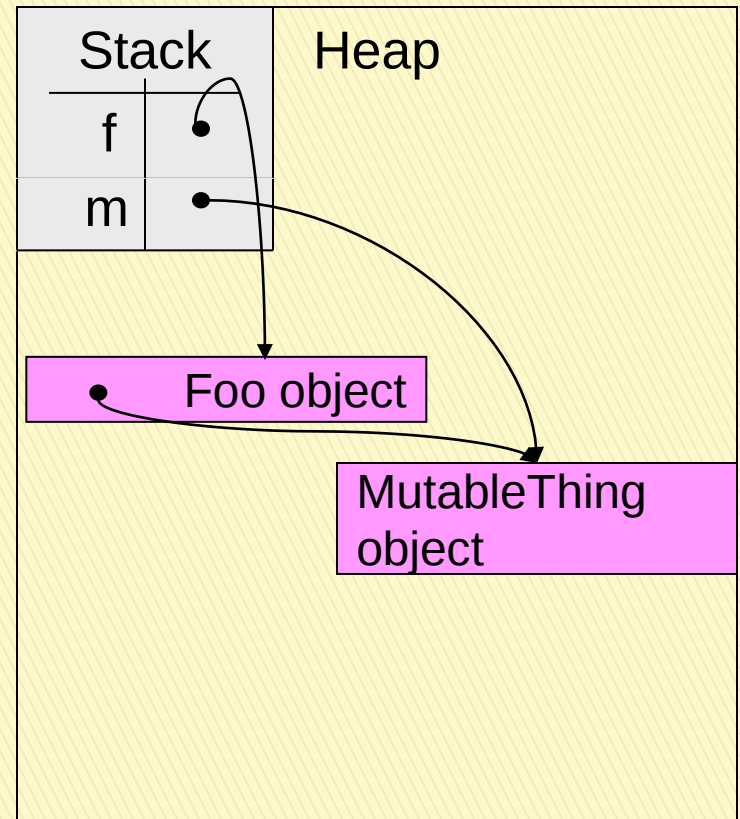
Overview

- ▶ Privacy Leaks
- ▶ Copying objects
- ▶ Two-Dimensional Arrays

Privacy Leaks

```
public class MutableThing {  
    ...  
    public void mutateMe() {...};  
}  
public class Foo {  
    private MutableThing q  
        = new MutableThing();  
  
    ...  
    public MutableThing getQ(){  
        return q;  
    }  
}
```

- ▶ Consider following code
Foo f = new Foo ();
MutableThing m = f.getQ();
m.mutateMe();
- ▶ After this executes, what happens?
- ▶ This phenomenon is called a privacy leak
 - Private instance variables can be modified outside class
 - Behavior is due to aliasing
- **Example:** privacyLeak package

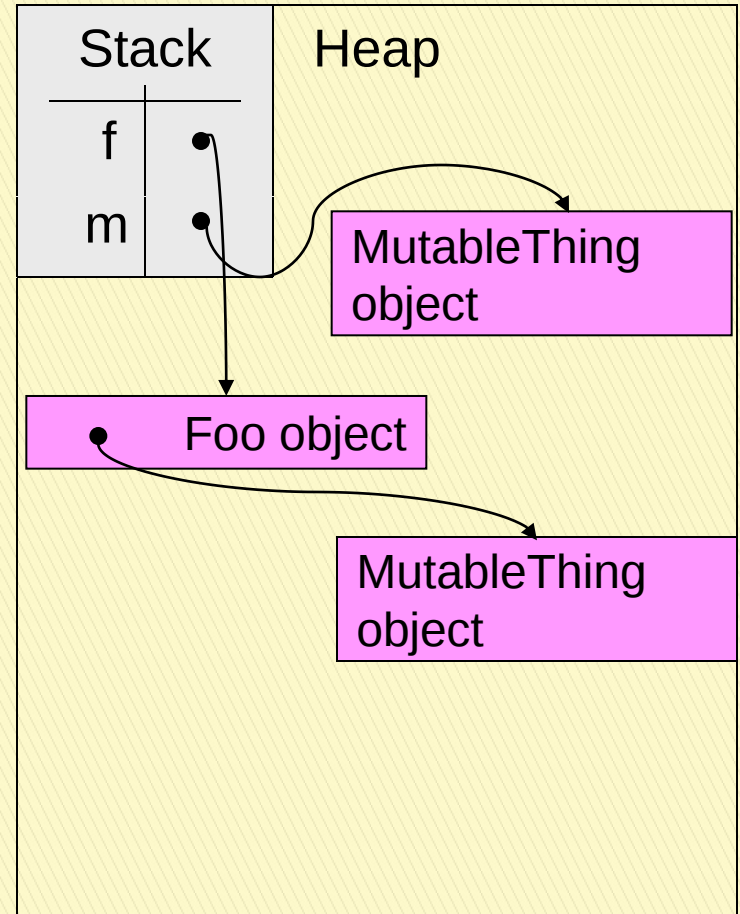


Fixing Privacy Leaks

- ▶ Return copies of objects referenced by instance variables
- ▶ To fix getQ method in Foo:

```
MutableThing getQ(){  
    return new MutableThing(q);  
}
```

 - This returns a copy of q
 - Changes made to this copy will not affect original



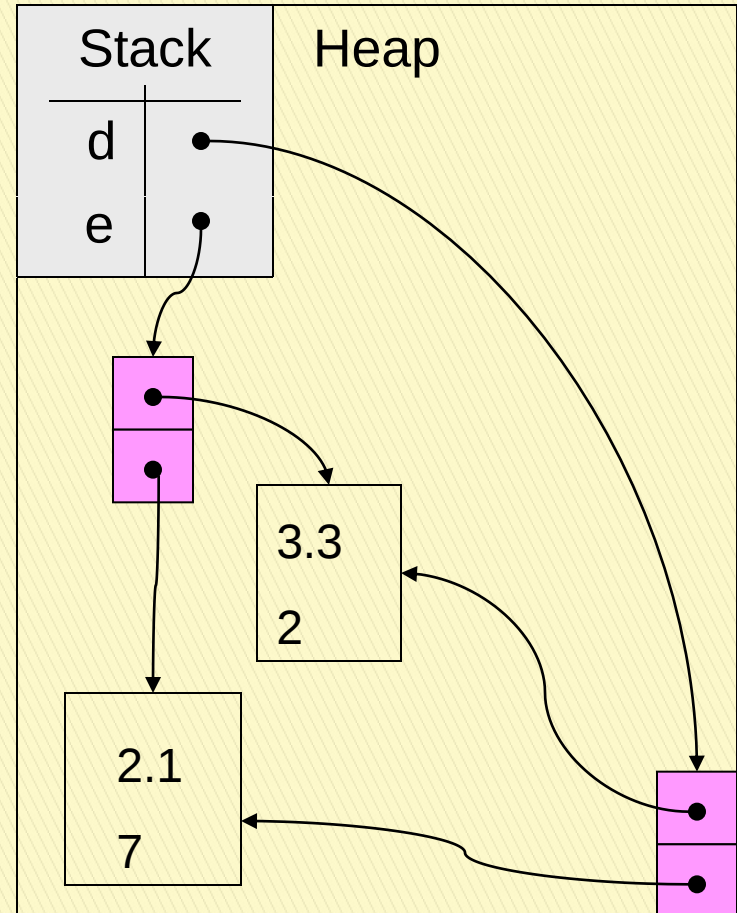
Copying Objects

- ▶ We can define three ways to copy objects
 - Shallow Copying
 - Reference Copying
 - Deep Copying
- ▶ Let's see examples in the context of arrays

Shallow Copying

```
Person[] d = {  
    new Person(2.1,7, ...),  
    new Person(3.3,2, ...)  
};
```

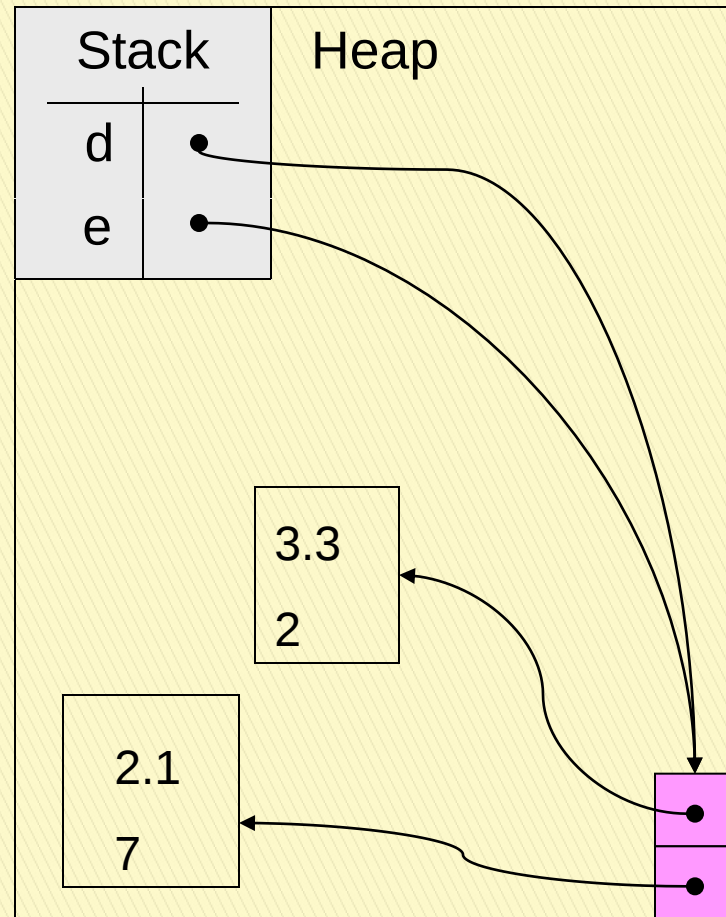
```
Person[] e = new Person[d.length];  
for (int i = 0; i < d.length, i++){  
    e[i] = d[i];  
}
```



Reference Copying

```
Person[] d = {  
    new Person(2.1,7, ...),  
    new Person(3.3,2, ...)  
};
```

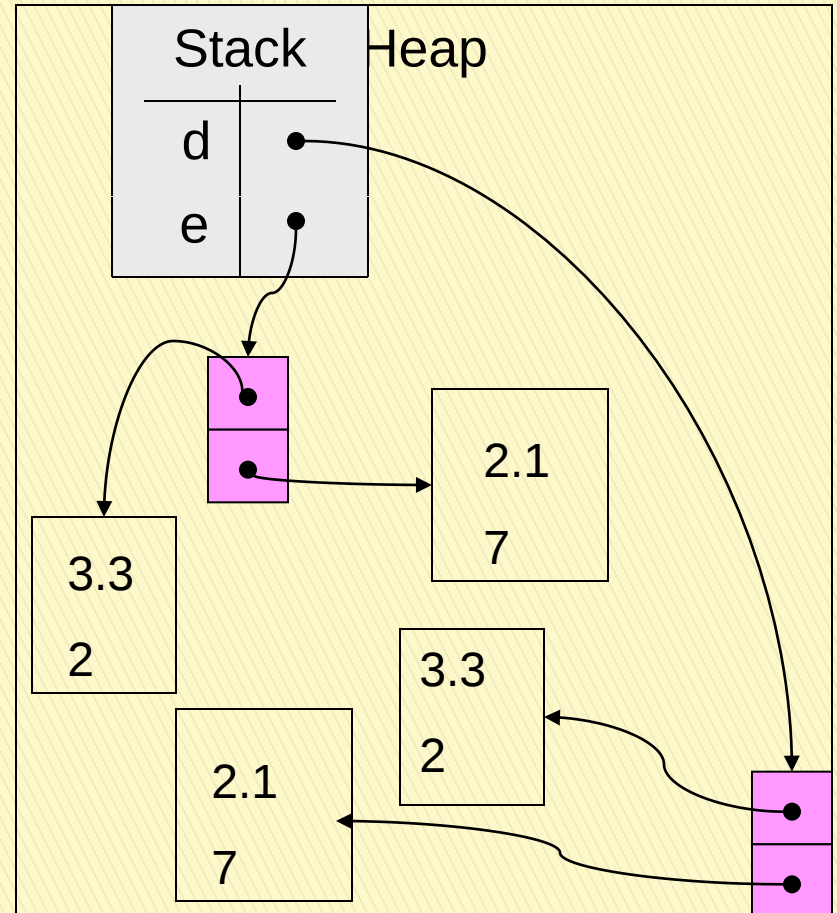
```
Person[] e = d;
```



Deep Copying

```
Person[] d = {  
    new Person(2.1,7,...),  
    new Person(3.3,2,...)  
};
```

```
Person[] e = new Person[d.length];  
for (int i=0; i<d.length; i++) {  
    e[i] = new Person(d[i]);  
}
```



Example

- ▶ CDCollector contains an array of CD's;
- ▶ ReCDCollector contains an array of rewritableCD's

- ▶ **Reference Copy**

```
public ReCD[] getCDsReferenceCopy() {  
    return myFavorites;  
}
```

- ▶ **Shallow Copy**

```
public ReCD[] getCDsShallowCopy() {  
    ReCD[] copy = new ReCD[myFavorites.length];  
    for (int i = 0; i < copy.length; i++)  
        copy[i] = myFavorites[i];  
    return copy;  
}
```

- ▶ **Deep Copy**

```
public ReCD[] getCDsDeepCopy() {  
    ReCD[] copy = new ReCD[myFavorites.length];  
    for (int i = 0; i < copy.length; i++)  
        copy[i] = new ReCD(myFavorites[i]);  
    return copy;  
}
```

```
ReCDCollectionOwner p =  
    new RECD...;  
ReCD[] a = p.getCD...();  
a[0] = otherCDalreadycreated;  
a[0].rewrite("other", "name");
```

When to Use What kind of Copying?

- ▶ Deep copying provides maximal protection against aliasing (but takes a lot of time and space if it is not necessary)
- ▶ Storage space and time used
 - Reference → least
 - Shallow → middle
 - Deep → most
- ▶ If the class is mutable, aliasing is something to be avoided and you must have true copies to prevent privacy leaks and modifications outside
- ▶ If you know the class is immutable, aliasing doesn't hurt but neither does making true copies (except wasted space and time)
- ▶ If storage is an issue, aliasing problems may be worth copying with but must be well documented

Array Application

- ▶ Remember that prime numbers
 - Prime number → positive integer that has exactly two positive integer factors, 1 and itself
 - Prime number → integer that is not the product of two smaller integers
 - 1 is not a prime number
- ▶ Computing prime numbers using Sieve of Eratosthenes
- ▶ **Example:** Driver.java, PrimeGenerator.java
- ▶ Information about Sieve of Eratosthenes
 - http://en.wikipedia.org/wiki/Sieve_of_Eratosthenes

Arrays Class (Honors)

- ▶ Arrays class (notice the s) contains various methods for manipulating arrays
 - Sorting
 - Searching (binary search)
 - Copying of arrays (equality)
 - Comparison of arrays (are they deep copies?)
 - Array initialization
 - String representation of arrays
- ▶ Let's see the Java API
 - <http://download.oracle.com/javase/6/docs/api/index.html>