

## Quiz #2 Answers

1. In Java, arrays of objects are actually arrays of references to objects. How is this significant to you when you are programming? In particular, what additional power does it give us?

**Having the array store references to objects is extremely significant in one major way, it allows your arrays to be polymorphic. If the array stored actual objects, it would have to know the correct size of all the objects and different subtypes might have different sizes. By storing references the array can reference any subtype of the declared type of the array and things work just fine.**

**Answers that got partial credit included things like the fact that the array starts off as all nulls so you have to initialize them all before you can use them.**

**An odd side effect of this that you might never have to use is that you can make arrays with variable numbers of dimensions in Java. I've done this in one program where the user determines the dimensionality of the array. The type used is an Object[] which can hold references to any object types. Arrays are themselves object types, so this could be an array of arrays or not. I use the instanceof operator to determine if it holds arrays or just plain data and if it holds arrays, they are of type Object[] so the process can repeat to any depth.**

2. Make a call to a sort that uses a comparator and have it sort an array of Strings, strArray, in reverse ASCIIbetical order (just backward from the normal order it would do). You can call Arrays.sort or assume you have one of the sorts we wrote in class.

```
Arrays.sort(strArray,new Comparator<String>() {  
    public void compare(String s1,String s2) {  
        return -s1.compareTo(s2); // or s2.compareTo(s1) if you prefer.  
    }  
});
```

Extra Credit: What is one of the examples from the text used to illustrate stacks?

**The book used a reverse polish calculator and a program that solves mazes as the examples.**