Announcements:
- Section this week in the classrooms. Bring your P5 if you have questions.
- Project 5 due Thursday at 6pm
- Prelim 3 on Tuesday, 11/15, at 7:30pm
- Review session: Sun 1-2:30 UP B7
- DrJava: remember to turn off backups! See 11/4 announcement on course website.
- P5 CallOption (and PutOption) method getShares should have int as the return type.

Previous Lecture:
- Defining a class:
  - Static variables and methods
  - Method overloading
- Take-home exercise: Person class

Today's Lecture:
- Wrap up Interval and Person classes
- 1-d array
- Selection sort
- Linear search
- Binary search (will discuss in section)

Reading:
- Sec 6.1, pp 382-386 of Sec 6.3

Chain invocation of methods
- Suppose there are 3 intervals: i1, i2, i3
- You know that i1 and i2 overlap
- Write code to find if the overlapped interval of i1 and i2 is in interval i3

```java
Interval i1 = new Interval(...);
Interval i2 = new Interval(...);
Interval i3 = new Interval(...);
// Assume i1 and i2 overlap
if (                                 )
    System.out.println("in i3");
else
    System.out.println("not in i3");
```

Modify Person class
- Modify Person class to store data about a Person's best friend: add another instance variable friend
- What should be the type of the field friend?
- Add two more methods to the class definition: makeFriend, beFriendOf
Arrays

- An array is an object
- An array is an ordered list of values (or objects)
- Each element is of the same type

Entire array has a single name

Each element has an integer index

An array of size N is indexed from 0 to N-1

<table>
<thead>
<tr>
<th>data</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>79</td>
<td>87</td>
<td>94</td>
<td>82</td>
<td>67</td>
<td>98</td>
<td>87</td>
<td>81</td>
<td>74</td>
<td>91</td>
</tr>
</tbody>
</table>

Arrays

- An array is an object
- An array is an ordered list of values (or objects)
- Each element is of the same type

Entire array has a single name

Each element has an integer index

An array of size N is indexed from 0 to N-1

<table>
<thead>
<tr>
<th>data</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>79</td>
<td>87</td>
<td>94</td>
<td>82</td>
<td>67</td>
<td>98</td>
<td>87</td>
<td>81</td>
<td>74</td>
<td>91</td>
</tr>
</tbody>
</table>

Array declaration

```
type[] identifier;
```

Examples:

- int[] counts;
- double[] price;
- boolean[] flip;
- char[] vowel;
- String[] names;
- Interval[] series;

Array construction (instantiation)

```
new type[ size ]
```

Example:

- new int[4]

Declaration & creation:

```
int limit= 4;
double[] price;
price= new double[limit];
```

must be an integer

Array declaration & construction

```
type[] identifier = new type[size];
```

Example:

- int[] counts= new int[4];

Then values can be assigned into the cells, e.g.:

```
counts[0]= 6;  counts[2]= 9;
```

Array length and default values

Once created, an array has a fixed length, held in the array’s constant called length:

```
int[] counts= new int[4];
System.out.println(counts.length);  // will print 4
System.out.println(counts[2]);  // Array components have default values. Above statement will // print 0
```
Array creation with initializer list

Create an array using an initializer list:

```java
int[] x = new int[]{6,3,4,8};
```

Length of array is determined by length of the initializer list. 
Shortcut:
```java
int[] x = {6,3,4,8};
```

Only when declaring & creating in same statement!

Index operator []

Accesses an element of the array, e.g.:
```java
int[] count = new int[101];
// declaration & instantiation
count[70+9] = 98;
// set count[79] to 98
int face = (int)(Math.random()*6);
count[face] = count[face] + 1;
count[face]++;
```

Elements in an array

If `count` is of type `int[]`, i.e., an array of ints, then the type of `count[i]` is `int` and `count[i]` can be used anywhere an `int` variable can be used.

Type of `count`: `int[]`
Type of `count[i]`: `int`

Pattern for processing an array

```java
// assume an array has been created and is referred to by variable A
for (int i=0; i<A.length; i++) {
    // perform some process
    // (on A[i])
}
```

Example

```java
// Create an array of length 6
// with random numbers in the range of 5 to 9. Calculate the sum.
```

Pattern for processing an array

```java
// Linear Search:
// f is index of first occurrence of z in array a
int f, k = 0;
while (a[k] != z && k < a.length)
    k++;
if (k == a.length) f = -1; // signal for z not found
else f = k;
```

Correct
Incorrect: f is off by one
Incorrect: while condition is wrong
Incorrect: if conditional is wrong

```java
// f is index of first occurrence of z in array a
int f, k = 0;
while (a[k] != z && k < a.length)
    k++;
if (k == a.length) f = -1; // signal for z not found
else f = k;
```