

### ■ 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.

November 8, 2005

Lecture 21

1

### ■ 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

November 8, 2005

Lecture 21

2

## 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`

```
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");
)
```

November 8, 2005

Lecture 21

4

```
Interval i1 = new Interval(...);
Interval i2 = new Interval(...);
Interval i3 = new Interval(...);
/* Without assuming that i1 and i2
   overlap */
```

November 8, 2005

Lecture 21

6

```
public class Person {
    private String name;
    private int age;

    public static final int LEGALage=18;

    /** Constructor */
    public Person(String name, int age)
    { this.name=name; this.age= age; }

    /** =This Person is an adult */
    public boolean isAdult()
    { return age >= LEGALage; }

    /** =String description of this Person */
    public String toString()
    { return name + " is " + age; }
}
}November 8, 2005 class Person
```

Lecture 21

9

## 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`

November 8, 2005

Lecture 21

10

```

/** Make a friend with Person p */
public void makeFriend(Person p) {

}

/** Become a friend of Person p */
public void beFriendOf(Person p) {

}

```

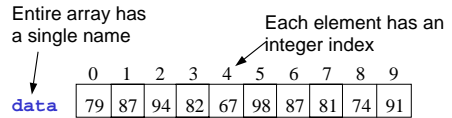
November 8, 2005

Lecture 21

12

## Arrays

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



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

November 8, 2005

Lecture 21

17

## Array declaration

```
type[] identifier;
```

Examples:

```

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

```

November 8, 2005

Lecture 21

18

## Array construction (instantiation)

```
new type[ size ]
```

Example:

```
new int[4]
```

must be an integer

Declaration & creation:

```

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

```

November 8, 2005

Lecture 21

19

## 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;
```

November 8, 2005

Lecture 21

20

## 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

```

November 8, 2005

Lecture 21

21

## Array creation with initializer list

Create an array using an initializer list:

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

Length of array is determined by length of the initializer list. **Shortcut:**

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

Only when declaring & creating in same statement!

November 8, 2005

Lecture 21

23

## Index operator [ ]

```
identifier[integer_expression]
```

Accesses an element of the array, e.g.:

```
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]++;
```

November 8, 2005

Lecture 21

24

## 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`

November 8, 2005

Lecture 21

26

## Pattern for processing an array

```
// 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])
}
```

November 8, 2005

Lecture 21

28

## Example

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

November 8, 2005

Lecture 21

29

```
// 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

November 8, 2005

Lecture 21

32