Note

- Reading:
  - Theory: Ch 0, 5.1-5.3, 1.1-1.6, 4.1-4.4
  - Programming: Ch 1, Ch 2, 3.1, 3.1-3.7
- HW1 due today, February 12 at 5p
  - submit programming online
- HW2 is out. Due XX/XX/04.
  - start soon! It’s longer that HW1 and will take more time.

What we are covering today

- Quick review from lab 2
  - Casting
  - Arrays
- If...else
- Iteration/Looping
  - while statements
  - do statements
  - for statements
Arrays (1)

• Declaration:
  
  ```java
  int[] myArray = new int[10];
  ```

• Trying to grab cell outside the array bounds causes an error (runtime)
  ```java
  myArray[10] = 4; // error: arrayoutofboundsexception
  ```

```
0 1 2 3 4 5 6 7 8 9
myArray.length
```
if...else

```java
if (condition) {
    <statement1>
    <statement2>
}
else {
    <statement3>
    <statement4>
}
```

```java
if (myArray.length>0) {
    myArray[0]=10;
} else {
    System.out.println("myArray length is "+myArray.length);
}
```

if...else...if

```java
if (condition) {
    <statement1>
    <statement2>
}
else if (condition) {
    <statement3>
    <statement4>
}
else if (condition) {
    <statement5>
    <statement6>
}
```

```java
if (myArray.length<5) {
    myArray[2]=100;
} else if (myArray.length==5) {
    myArray[2]=50;
}
```

**Iteration/Looping**

- Often, we want to do things many times.
- Repetitive tasks often have a structure. Exploit it using loops.
- Let’s look at our array initialization from before:

```java
myArray[0] = 0;
myArray[1] = 10;
myArray[2] = 20;
myArray[3] = 30;
myArray[4] = 40;
```

- Is there a pattern?
The while loop (1)

- “While ‘condition’ is true, run my statements.”

while (condition){
    <statement1>
    <statement2>
}

- Similar to the if structure
- What makes it stop?
  – You must do something to make condition evaluate to false!

The while loop (2)

- Let’s solve our array initialization problem using while

while (what is true?) {
    <do what?>
}

Think about our pattern…

The while loop (3)

- What do we want to do?

while (what is true?) {
    myArray[i]=i*10;
}

The while loop (4)

• When do we stop?
while (we have not looked at all cells) {
  myArray[i]=i*10;
}

• How many cells are there?
  - 5 (actually myArray.length)
• Where do we start?
  - 0 (beginning of myArray indices)
• How does the index “a” change?
  - increment by 1

The while loop (4)

• Put it all together
int i = 0;
while (i < myArray.length) {
  myArray[i]=i*10;
  i=i+1;
}

The while loop (5)

Try it:
int i = 0;
while (i < myArray.length) {
  myArray[i]=i*10;
  System.out.println("i="+i+" and myArray["+i+"] is "+myArray[i]);
  i=i+1;
}
The **do** loop (1)

```java
do {
    <statement1>
    <statement2>
    ...
} while (condition);
```

- Similar to **while** statement. So what’s the difference?
  - The `<statement>`s are guaranteed to run at least once.

---

The **do** loop (2)

- Can we populate `myArray` as before using a *do* loop instead?
  - Yes!
  - What do we need?
    - An index.
      ```java
      i = 0;  // reuse index i
      ```
    - Statement.
      ```java
      myArray[i]=i*10;
      ```
    - Increment.
      ```java
      i++;  // same as i=i+1
      ```
    - Condition.
      ```java
      i<myArray.length
      ```
The **do** loop (3)

```java
i=0; //reusing prior index
do {
    myArray[i]=i*10;
    i++;
} while (i<myArray.length);
```

- Potential problems?

The **for** loop (1)

- What have we seen is needed for looping?
  - a looping variables (i in previous examples)
  - a start value for looping variable
  - a stop condition
  - a way to change the looping variable so we reach our stop condition
- The **for** loop is no different. Just a different structure.
  - not based on the ‘if’

The **for** loop (2)

```java
for (int var = start; check; update) {  
    <statement1>
    <statement2>
}
```

- int var = start (creating our loop variable)
- check (our condition)
- update (our changing the loop variable)
The for loop (3)

Let's do our myArray changes again with for:

```java
for (int i = 0; i<myArray.length; i++) {
    myArray[i]=i*100;
}
```

int var = start  (creating our loop variable)

check (our condition)
The for loop (3)

Let’s do our myArray changes again with for:

for (int i = 0; i<myArray.length; i++) {
    myArray[i]=i*100;
}

update (our changing the loop variable)

Our Statement.
That’s it!

Some things to think about

• How would we loop backwards using a for loop?
  – with a while? or do...while?
• Do we always have to change the condition variable by 1?
• Can we have complex conditions? (&&, ||, !, etc.)
Wrap up

- Next time:
  - Methods
  - More decision and control statements
  - Basic Input/Output (I/O)
- HW2 is out. Due TUESDAY XX/XX/04
  - Get started. Longer than HW1.