Announcements

- Read for next time Chap. 4.6, Chap 8.5
  - Strings
  - More on arrays
- Assignment 5 out
- Classwork from Jan 31 is due Feb 2
- Classwork from today is due Tues, Feb 6
  - Finish before the next class
- Reading Quiz for next time

What we will do today

- Lecture
  - Loops, Arrays
- Review Classwork from last time
- Classwork today
  - APTs using arrays and loops

Both while and for loops

- Initialization
- Condition
- Body
- Increment
while

```java
public void printFencePost(int numberPosts) {
    String rail = "===";
    String post = "I";

    int num = 1;
    System.out.print(post);
    while (num < numberPosts) {
        System.out.print(rail);
        System.out.print(post);
        num++;
    }
    System.out.println(" ");
}
```

while (cont)

```java
x.printFencePost(6);
x.printFencePost(5);
x.printFencePost(12);
```

I==I==I==I==I==I
I==I==I==I==I
I==I==I==I==I==I==I==I==I==I==I==I==I

for loop

```java
public void printFencePostfor(int numberPosts) {
    String rail = "===";
    String post = "I";

    System.out.print(post);
    for (int k = 1; k < numberPosts; k++) {
        System.out.print(rail);
        System.out.print(post);
    }
    System.out.println(" ");
}
```

Array

- Declare an int array
  ```java
  int[] values = new int[12];
  ```
- Initialize it to these values:
  
  8 3 4 3 8 2 4 4 6 2 8 4
- Access item in slot 6 in the array
  ```java
  values[6]
  ```
- Array is fixed size. The size is:
  ```java
  values.length
  ```
ArrayList

- Better to use than an array
- ArrayList
  - Can grow and shrink
  - Has methods for common tasks (see API)
  - Only holds objects
- Can’t have an ArrayList of int or double
  - There is a special Integer (an int that is an object) and Double (note the capital letters!)

ArrayList (cont)

- Create an ArrayList
  ArrayList<Integer> idlist = new ArrayList<Integer>();
- Add an element to the ArrayList
  idlist.add(8);
- Modify kth element in an ArrayList
  idlist.set(k, 8);
- Sum the elements in the ArrayList
  // sum up integers in the ArrayList
  int sum = 0;
  for (Integer current : idlist)
  {
      sum += current;
  }

Better to use ArrayList than array

- Sort an arrayList called numbers - easy
  Collections.sort(numbers);
- If you are given an array as a parameter
  - Copy values to an ArrayList
  - Then can work with the ArrayList
- If you need to return an array
  - Copy values from ArrayList to an array
- For Example, you’ll need to do both of these for APTs – they only use arrays, but you can convert the array to ArrayList and then back to array.

Example: singleNumbers

- Given an integer array that could have duplicates, return an array that has only unique numbers from the original array (get rid of duplicates!)
- For example if the parameter array is:
  - 8 5 5 8 5
- Then the array to return should be:
  - 8 5
First convert array to ArrayList

```java
public int[] singleNumbers(int[] ids) {
    // convert the array "ids" into an ArrayList "idlist"
    ArrayList<Integer> idlist = new ArrayList<Integer>();
    for (int k = 0; k < ids.length; k++) {
        idlist.add(ids[k]);
    }
}
```

Second, find unique numbers

```java
// create an ArrayList that will hold unique numbers
ArrayList<Integer> singles = new ArrayList<Integer>();
singles.add(idlist.get(0)); // first number is unique
for (Integer current : idlist) {
    boolean isIn = false;
    for (Integer currentSingle : singles) {
        if (current.equals(currentSingle))
            isIn = true;
    }
    if (!isIn)
        singles.add(current);
}
```

Third, convert ArrayList to Array

```java
// convert ArrayList to array
int[] answer = new int[singles.size()];
int position = 0;
for (Integer currentSingle : singles) {
    answer[position] = currentSingle;
    position++;
}
return answer;
```

Classwork Last time

- Red circle was painted to canvas
- Moved the red circle by moving center of circle and repainting
- Bounce red circle – checking if ball past one of the four edges, if so reverse direction
- Create a new class to represent Bouncer
  – Now can create multiple Bouncers easily