CS/IT115 Lecture 2

2013-01-29
Instructor: Alex Dusenbery
Review of Arrays, APIs
Last time - APIs

- We'll do a quick class exercise to reinforce the idea of an API (also on class web page)
- Then we'll look at an example of a program that uses an array as review
IT IS **BRUTAL** OUT. SO MUCH FOR GLOBAL WARMING, HUH?

* sigh *

THIS USED TO HAPPEN ALL THE TIME.

WHAT?

YOU’RE FROM ST. LOUIS, RIGHT?

ON AVERAGE, IT USED TO GET BELOW 0° F THERE A HANDFUL OF DAYS PER YEAR.

BUT YOU HAVEN’T HAD A DAY LIKE THAT SINCE THE NINETIES.

![Graph showing days with temperatures below 0°F from 1970 to 2010](source: RCC-ACUS/CLIMATECENTRAL)

THEN, IN 2014, WHEN THE FIRST POLAR VORTEX HIT, IT DIPPED BELOW ZERO FOR TWO DAYS.

AND EVERYONE FREAKED OUT

BECAUSE WHAT USED TO BE NORMAL NOW FEELS TOO COLD.

IT IS TOO COLD!

THE FUTURE:

LOOK AT THIS—ICE! IN ST. LOUIS! SO MUCH FOR GLOBAL WARMING.

* sigh *
Temperature2.java

- Program to read a series of high temps (daily), then report the average and # of days > avg
- Needs an array b/c it has to look at each number more than once.
- We don't know ahead of time how many days we'll see, so we can't just declare a separate numeric variable for each day.
Temperature2.java

• The workflow:
  – Ask for the number of days using Scanner's nextInt()
  – Set up an array with a slot for each day.
  – For each day:
    • Read a number, put in array, sum it up
  – Compute avg.
  – Count days over avg., report results
Arrays For the Win

• The array of ints for daily highs is the star here
• We needed to know number of days ahead of time to create the array, and we can't resize an array (serious restriction)
• We could, if necessary, move all ints into a new, bigger array (common trick)
• ArrayLists – better than arrays, will study after objects.
Array of High Temps

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>5</td>
<td>41</td>
<td>19</td>
<td>17</td>
<td>18</td>
<td>25</td>
<td>32</td>
<td>9</td>
</tr>
</tbody>
</table>

- We get the above array after entering temps
- So temps[0] == 16, temps[5] == 18, etc.
- Can get the length using temps.length (length is an attribute of arrays, not a method like String.length())
- Can get the last number of array using:
  - int lastTemp = temps[temps.length -1];
  - Could change a value: temps[2] = 44;
How to print arrays?

//Could use a basic for loop:
for (int i=0;i<temps.length;i++) {
    System.out.print(temps[i] + " ");
}

75 78 85 ...

//Or a for-each loop (better)
for (int tem: temps) {
    System.out.print(tem + " ");
}

Key Points of Program

- Use a Scanner named console, get an int from the user for numDays:
  - int numDays = console.nextInt();
- Set up an array for numDays ints:
  - int[] temps = new int[numDays];
- Loop through numDays cases:
  - for(int i = 0; i < numDays; i++) {//blah...}
Key Points of Program

- Get single int from user, place in array:
  - `temps[i] = console.nextInt();`
- For avg., need a sum and the number of observations.
- Use a running sum when getting user input.
- After loop, average = sum / `numDays`
- **Careful!** – integer division $\rightarrow$ inaccuracy
ARE YOU COMING TO BED?

I CAN'T. THIS IS IMPORTANT.

WHAT?

SOMEONE IS WRONG ON THE INTERNET.
**Integer Division**

- If `int sum = 14, numDays = 3`, then 
  \[ \text{sum/numDays} = 14/3 = 4 \] (int arithmetic). Should really be 4.666
- Need to use a cast to double:
  - `double average = (double)\text{sum/numDays};`
- After this, program just prints out results.
Refactoring

• Definition from wikipedia:
  - “Code refactoring is the process of restructuring existing computer code without changing its external behavior. Refactoring improves nonfunctional attributes of the software. Advantages include improved code readability and reduced complexity to improve source code maintainability, and create a more expressive internal architecture or object model to improve extensibility.”
Let's try to use methods

- One job that might be repeated in future: summing up the array
- We can write a method that accepts any array (of ints) and returns its sum:
  - Header: public static int sumArray(int[] a)
  - Call to it: int sum = sumArray(temps);
SumArray Method

- Method declaration: header + body

```java
public static int sumArray(int[] a) {
    int sum = 0;
    for (int i=0; i<a.length; i++) {
        sum += a[i];
    }
    return sum;
}
```
SumArray Method

- Could use the “for-each” loop instead:

```java
public static int sumArray(int[] a) {
    int sum = 0;
    for (int entry: a) {
        sum += entry;
    }
    return sum;
}
```
Refactoring

- Can change our Temperature2.java as follows:

```java
int[] temps = new int[numDays];

// record temperatures and find average
for (int i = 0; i < numDays; i++) {
    System.out.print("Day " + (i + 1) + "'s high temp: ");
    temps[i] = console.nextInt();
}
int sum = sumArray(temps);
double average = (double) sum / numDays;
```
More Array Stuff

- If we wanted, we could even change an array from inside a method, unlike an int method parameter.
- An array is an object, not a primitive type, so it's name when used as a method parameter is an object reference.
- We'll talk more about object references soon.