CS/IT115 Lecture 3

2014-02-03
Instructor: Alex Dusenbery
Review of Arrays, Intro to Objects
Last Time: Arrays...

- Please pick up the class exercise at front of classroom
- We'll go over arrays together for a bit, then work on the exercise.
- First homework available, due next week
- First project posted today
...and some methods

```c
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
    // guaranteed to be random.
}
```
Strings are Objects

- See p. 162/1121 or:
  - [http://docs.oracle.com/javase/7/docs/api/java/lang/String.html](http://docs.oracle.com/javase/7/docs/api/java/lang/String.html) for useful methods.

- Example:
  - String fruit = “apple”;

- In this example, fruit is a variable that “points” to a String object.
String are Objects

- String fruit = “apple”;
- We can think of the memory representation of this as follows:
- We say that the fruit variable stores a reference to the String object “apple”, not the object itself.
Methods of String

• Examples of method calls on String objects:
  – fruit.length() → 5
  – fruit.startsWith("ap") → true
  – fruit.toUpperCase() → "APPLE"

• In the last example, "APPLE" is another new String object created (returned by) the call to the toUpperCase method.
Strings are Special

• The String class is special. Instances of it (that is, a String object) are immutable.
  – Instances of a Class $\leftrightarrow$ Objects

• They can also be constructed without using the “new” keyword.

• They can also be concatenated together using the “+” operator.
String are Immutable

- Immutable → Can't be changed once created
- Since we can't change Strings, we create new Strings out of old Strings instead
- Every String has a “state”, which is just the characters inside the string.
- Since Strings are immutable, we can't change their state – the characters inside them remain the same from creation through deletion
Arrays of Strings

• Can have an array of Strings:
  – String suits[] = {“Clubs”, “Diamonds”, “Hearts”, “Spades”};

• suits[0] is “Clubs”, … suits[3] is “Spades”

• We could use this in some card-playing scenario.
A Deck of Cards

- There are 52 cards in a deck. We number these 0 to 51.
- Cards 0-12 are clubs, 13-25 are diamonds, 26-38 are hearts, and 39-51 are spades.
- We could use integer division to compute the suit number from the card number – division by 13 tells us how many 13's are in a number
A Deck of Cards

• Card 22 has suit $22/13 = 1$, and $\text{suits}[1] =$ “Diamonds”, so card 22 is a Diamond.

• Card 32 has suit $32/13 = 2$, and $\text{suits}[2] =$ “Hearts”, so card 32 is a Heart.

• Generally, card $x$ has suit $x/13$ named suits[$x/13$].

• Now, let's do the class exercise.
Scanners are Objects

- Scanner: a more normal type of object
- A Scanner can be changed, needs the “new” keyword to construct.
- Scanner console = new Scanner(System.in);
- // print prompt
- double loan = console.nextDouble();
- // print prompt
- int years = console.nextInt();
Scanners are Objects

- Console is an object that keeps track of what characters have been used already, and arranges with the OS to get more chars as needed.
- Page 387-388 illustrates this.
- The kind of memory inside the object is called the object state.
Scanners

- It's possible to ask the Scanner if there is more data to work on (p. 335-338)

- The method on p. 338 has the following header:
  - public static int getInt(Scanner console, String prompt)

- Checks for int coming in with console.hasNextInt(), keep trying for int...
Changing Scanner Radix

- Radix → number base. We usually like base 10, or “radix” 10. But hexadecimal numbers (base 16) are important in CS/IT.
- Memory addresses are usually written in hex.
- Hex digits:
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f
  - 16 digits, values 0-15
Hex Numbers

• Example: a1

• In hex, a is 10, so a1 = 10*16 + 1*1 = 161

• Bigger example: 2af3 ->

• (2*16^3) + (10*16^2) + (15*16^1) + (3*16^0)

• → 10995
Scanner Radix Change

• The above program used decimal numbers, but we could switch the radix of a Scanner object using the method useRadix

• input.useRadix(16); //switch to hex

• Now input.nextInt() can read “A1” or “a1” into an int (value 161). Note that ints are stored as binary in memory, so it doesn't care how the human typed the number (hex or decimal)
Scanner Radix

- A Scanner object stores inside it the “current” radix – an int between 2 and 36.
- The radix starts at 10 by default, but can be changed using useRadix() method.
- We can “run” the Scanner using its methods.
- Changing the radix is like a customization of the Scanner object for our purposes.
- Could also customize the token delimiter with useDelimiter().