CS 302: INTRODUCTION TO PROGRAMMING IN JAVA

Lecture 16
REVIEW

- What is aggregation?
- Object variables are what type of variables?
- What does null mean?
- How do you test for null?
- How is a toString() method useful?
- What does the "this" keyword refer to?
- Predicate method means?
ENCAPSULATION

- Use a public interface to hide implementation details
- Use a public interface to ensure data protection
  - Don't let a balance in a BankAccount go negative
  - Make sure a Date object has a valid year/month/day
  - etc.
PARALLEL ARRAYS

- Sometimes you have 2 (or more) part data (ex. a list of food items and their prices)
- One way to solve: use 2 parallel arrays, one for the food items and one for their prices

```java
System.out.println("Name: "+ foodItems[i] + "price:" + prices[i]);
```
SOLUTION: USE AN OBJECT

- Each slice of the 2 individual arrays represents 1 thing (an Item object) that has 2 properties (a name and a price)
- Create an Item object with name and price as instance data and then you only need one array of Item objects

foodItems (Item array)

```
Item 1
Item 2
Item 3
...  
```

To get the name of item 3 the call would be:
```
foodItems[2].getName()
```
What if we want a variable that is shared amongst all objects of a class?

Ex. A BankAccount might have an account number and each additional account should have a sequential account number (i.e. The first BankAccount will have account number = 1001, the second will have account number 1002, etc.)

Solution: use static variable

How do we access static fields?
public class BankAccount
{
    private double balance;
    private int accountNumber;
    private static int lastAssignedNumber = 1000;
    public BankAccount()
    {
        balance = 0; //do we need this line?
        BankAccount.lastAssignedNumber++;
        accountNumber = lastAssignedNumber;
    }
}
Each new BankAccount will have its own balance and accountNumber, but all have the same lastAssignedNumber (accessed with BankAccount.lastAssignedNumber)
STATIC VARIABLES AS CONSTANTS

- Static variables, like instance variables, should be declared private.
- Exception: if the static variable is a constant it can be declared public.
- Ex. A BankAccount might have an overdraft fee common to every account (that never changes).

```java
public class BankAccount {
    public static final double OVERDRAFT_FEE = 29.95;

    // ...
}
```

To Access: `BankAccount.OVERDRAFT_FEE`
STATIC METHODS

- Static Methods = methods that are NOT invoked on an object
- Ex. Math.sqrt(), Math.pow – there is no instance of a "Math" object that is changing or doing anything (we never say Math whatever = new Math())
- Ex. you might want a calcArea method in a Circle class
  ```java
  public static double calcArea(int radius)
  {
      return radius*radius*PI;
  }
  ```
- Call would be Circle.calcArea(x);
- Cannot directly interact with instance data (why?)
A Package is a set of related classes

Ex. java.util – utility classes such as Scanner, Random

To put a class in a package:
  - package packageName; //goes above class definition

Ex. Put BankAccount in a Financial package
package financial;
public class BankAccount {
IMPORTING PACKAGES

- Already know how to do
  - import java.util.Scanner; //import specific class within a package
  - import java.util.*; //import all classes within package
- Why don't we need to import anything to use the Math class?
  - Math class is in java.lang package
  - Every java source file automatically does a import java.lang.*
Design a program that detects course conflicts for students since you are working in IT division at Registrar’s Office.

Create a package called courseConflicts for 3 instantiable classes called Course, Student, TimeSlot.

A course has a name and a few time slots, users can 1) add slot to the course and 2) check if this course conflicts with another one.

A student can have many courses, he/she can 1) be enrolled in some course and 2). Get all course conflicts he/she has.

A TimeSlot have day, starting time and length, it can also check if it conflicts with another TimeSlot.