Be in the know …

- **Add yourself to** compsci@duke.edu
  - Duke University mailing lists – add yourself
    https://lists.duke.edu/sympa
  - Compsci related events, jobs, research opportunities
- **Apply for Data+, CS+, Code+**
  - summer research at Duke, paid, hire lots of 1st year students, 2nd year, etc.
  - Apply in January!
    https://www.cs.duke.edu/undergrad/summer_research

---

My Digital Hand

- A fifth of you are not in the know……
- Sign up for my digital hand
- Check your email……

---

C is for …

- **Class**
  - Framework for creating objects
- **Collections and Collection**
  - See java.util.* for details
- **Collaboration**
  - Review the policy
Plan for the Day

- Review Object concept: classes, P0
  - What is a class, object, instance variable

- Review arrays in Java: methods and concepts
  - Required for APTs due next week
  - Move toward ArrayList and other collections

- Coding and helper functions
  - Efficient programming and not efficient programs

Strings: Example from last time

- You can’t modify a string, always create new String

```java
String s = new String("joy");
// s is "joy"

String t = s;

s = t + t;
```

Strings: Example from last time

- You can’t modify a string, always create new String

```java
String s = new String("joy");
// s is "joy"

String t = s;
// t is "joy"

s = t + t;
```
Strings: Example from last time

- You can’t modify a string, always create new String

```java
String s = new String("joy");

String t = s;

s = t + t; // create new String
// s is "joyjoy"
// t is "joy"
```

Class

- Adjective, noun, close to a verb
  - Show some ___ you’re in a great ___, that’s a ___ act, let’s ___-ify that

- Fundamental part of object-oriented programming
  - All Java code is in a class, alas the primitives
  - In Python int is a class, has no upper bound
  - In Java int is a primitive, $2^{31}-1$ maximal value

Class encapsulates state and behavior

- Class is a template, object has characteristics
  - Dogs have fur, speed, temperament, size, …
- Typically we don’t use examples like this, but they can help build intuition and understanding
  - Class dog, retriever extends dog, method bark()

Class and Object

- If we had a Retriever class we could instantiate an instance of the class, i.e., create an object
  ```java
  Retriever ks = new Retriever("kelsey");
  ```
- Class is an object factory, calling new creates a new object that is an instance of the class
  - We could call a method: `ks.bark()`
Classes in Java

- Define class **Foo** in **Foo.java**
- Create object by calling **new Foo(..)**
- Access object by calling methods: **obj.doSomething()**
- Some methods return a value, use it!

Classes in Java

- **State**: instance variables: private
- ** Constructors**: initialize instance variables
- **Methods**: functions aka behavior
- **Documentation**: Javadoc and other comments

Work-Flow for Assignments

- What is the work-flow for P0 and Assignments?
  - Login to gitlab
  - Code URL to P0 in gitlab: [https://coursework.cs.duke.edu/201spring20/p0-person-sp20](https://coursework.cs.duke.edu/201spring20/p0-person-sp20)
  - Fork it (makes a copy in the cloud)
  - Clone with ssh
Using a shell

- Place to type shell commands
- On Mac use Terminal, Windows use Bash Git
- What is this?

```
Susan@LAPTOP-NTK3PPUK MINGW64 ~/IdeaProjects/spring20
```

$ is a prompt, ready for a shell command

A few shell commands

- `pwd` – display current path
- `cd` – change into main folder/directory
- `cd name` – change into folder named name
- `cd ..` – change back into parent folder
- `ls` – show files in current folder

Let’s see some of those…

Back to Work-Flow for Assignments

- Clone with ssh

  ![Clone with SSH](image)

- Go to your shell
  - `cd` (to folder you want to put your P0 in)
  - `git clone` (SSH URL you copied)
  - `ls` (will show your files)

- Using IntelliJ complete the assignment
  - Save code often to gitlab!
**Work-Flow for Assignments (cont)**

- Send code back to gitlab (DO OFTEN)
  - `cd` (into project folder)
  - `git add .`
  - `git commit -m "comment on what you did"`
  - `git push`

- Now to Gradescope and submit project
  - Don’t like results - fix code, push code, run on Gradescope again

---

**Classes and P0**

- How many Person objects created?
  - Each has a name and an age, different for each instance. Thus: instance variables

```java
public class PersonDriver {
    public static void main(String[] args) {
        Person p = new Person();
        Person q = new Person( name: "Sam", age: 21);

        System.out.println(p.getName());
        System.out.println(q.getName());
        System.out.println(p.getAge());
        System.out.println(q.getAge());
        System.out.println(p);
        System.out.println(q);
    }
}
```

- To create? Call new which invokes a constructor
  - No return type, initialize instance variables

- Access levels: private only within class, public from other classes
  - Technically there is a package access, we ignore
Constructor

- Same name as class
  - No return type

- Overload with different parameters
  - Each should initialize all instance variables

- Factor out common code into helper method if lengthy
  - Can call another constructor using `this(…)`

What is `this`?

- An object instance refers to itself
  - Method or constructor: object references itself
  - Every reference to an instance variable `myVar` could be written as `this.myVar`

- Code for an object to pass itself:
  - `callMethod(this,"hello");`
- Constructor can call other constructor
  - `this("hello");`

Running a Java Program

- On laptop/desktop launch/run point is the main method in any class
  - Driver programs in P0, runs/drives the code
  - Method signature `required` to run program

```java
public class Person {
    private String myName;
    private int myAge;

    public Person(String name, int age) {
        myName = name;
        myAge = age;
    }

    public Person() { this("NoName", age: 13); }

    public String getName() { return myName; }
    public int getAge() { return myAge; }

    @Override
    public String toString() {
        return String.format("%s %d", getName(), getAge());
    }
}
```

```java
public class PersonDriver {
    public static void main(String[] args) {
        Person p = new Person();
        Person q = new Person( name: "Sam", age: 21);

        System.out.println(p.getName());
        System.out.println(q.getName());
        System.out.println(p.getAge());
        System.out.println(q.getAge());
        System.out.println(p); System.out.println(q);
    }
}
```
WOTO (3 minutes)


Luis von Ahn

- Duke 2000, Math
- Duke Honorary Degree 2017
- CEO Duolingo
- Macarthur Award, 2006
- MIT-Lemelson Prize, 2018

“It’s amazing how motivating it is to sit with somebody and say, ‘What you’re doing is really important.’ I use that a lot.”

Arrays, APTs, and APIs

- Why is alliteration important in writing?
- Why are these important in programming?
- APIs create possibilities

Array Details

- Once array created, it's size is fixed, can't grow!
  - Indexable elements can be changed

- Using `a[k]` we can read/write values
  - Instance variable `a.length` is size of array
    - No parentheses, hence not a method
  - Notice dot notation: object dot name
Indexing for loops and arrays

• Constructing and initializing ...  
  ```java
  int[] a = new int[100];
  for(int k=0; k < a.length; k += 1){
      a[k] = 99;
  }
  ```

• Let an API-call fill in array: `java.util.Arrays`  
  ```java
  Arrays.fill(a,99);
  ```

• https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/Arrays.html

For each loops and arrays

• For each loop: no index, no changing what’s stored  
  ```java
  int[] a = {1,2,3,4,5,6,7,8,9,10};
  int sum1 = 0; int sum2 = 0;
  for(int k=0; k < a.length; k += 1){
      sum1 += a[k];
  }
  for(int value : a){
      sum2 += value;
  }
  System.out.println(sum1 == sum2);
  ```

For Loop Summary

• `for(init; boolean guard; update) {...}`  
• `for(int k=0; k < a.length; k+=1) {...}`  
  • Initialization happens once, **before guard checked for the first time, never again**  
  • Initialization can introduce variables: **loop scope**  

  • Guard checked, if true loop body executes  
  • After loop body, update executes, guard checked

Control Construct Summary

• `if (boolean) {...}`  
  • Block executed when guard is true  
  • `{.}` not needed for single statement, use anyway  

• `if (boolean) {...} else {...}`  
  • Code in else block when negation true

• `while(boolean) {...}`  
  • Check boolean guard, execute body, repeat  
  • Guard checked again after body executed
**From Control to APIs**

- List and ArrayList similar to array, but ....
  - Grow as needed, can't use [k] to access
  - Powerful APIs, e.g., as follows

```java
class Main {
    public static void main(String[] args) {
        String[] f = new String[] {"apple", "cherry", "banana", "melon"};
        int k = 0;
        System.out.println("found \n");
        System.out.println("found \n");
        System.out.println("found \n");
    }
}
```

```java
jshell> for(int k=0; k < f.length; k++)
    if (f[k].equals("apple")) System.out.print("found \n");
found 2
```

```java
jshell> f
```

```java
jshell> a
a ==> "banana"
```

```java
jshell> Arrays.asList(f).indexOf(a)
$6 ==> 2
```

**From Control to APIs**

- List and ArrayList similar to array, but ....
  - Grow as needed, can't use [k] to access
  - Powerful APIs, e.g., as follows

```java
class Main {
    public static void main(String[] args) {
        String[] f = new String[] {"apple", "cherry", "banana", "melon"};
        int k = 0;
        System.out.println("found \n");
        System.out.println("found \n");
        System.out.println("found \n");
    }
}
```

```java
jshell> for(int k=0; k < f.length; k++)
    if (f[k].equals("apple")) System.out.print("found \n");
found 2
```

```java
jshell> f
```

```java
jshell> a
a ==> "banana"
```

```java
jshell> Arrays.asList(f).indexOf(a)
$6 ==> 2
```

**Solving an APT Together**

- Totality (see APT page on course site)

- Solve by hand: `a = {20,30,40,50,60}` stype="odd"

- Use what you know, but implement in Java
  - Check ideas using jshell (Java 9 and later)
  - Command line is your friend!

**Think Before You Code**

- Solve by hand ... Check your understanding of examples ... think about solution you'll write ... 
  - Then think before fingers on keys
Coding Interlude

- Working on Totality APT in IntelliJ
  - Odd? Even?
  - Control: if, if-else, …

WOTO (3 minutes)


Josh Bloch

- Led design of Java Collections Framework
- Formerly Java Chief Architect at Google
- Professor of the Practice CMU

*APIs should be easy to use and hard to misuse. It should be easy to do simple things; possible to do complex things; and impossible, or at least difficult, to do wrong things.*

Summary of Java-isms

- Loop using indexes over an array
  - The for-loop: initialize; guard/check; update
- Totality: loop over odd indexes only?
  - In some cases, …
- How do we check for String equality?
  - .equals compared to ==

- How do we submit an APT?
  - Test, Grade, REFLECT
  - APTS – one grace day, NO LATE AFTER THAT