Recitation

- Recitation –
  - Must be completed before Friday
  - Sakai quiz
    - take it as many times as you want!

- Problems with Eclipse and Ambient
  - Go to the Link
  - Come to office hours (Grad TAs or Professor)
  - Post problem to Piazza
Today

- Java
  - Primitive and Object type
  - Classes
  - Code example
- Java
  - Arrays

Primitive types

- int myInt = 5;
- char myChar = ‘t’;
- boolean myBool = false; // or true
- double myDouble = 5.5;
Primitive types

• int \texttt{myInt} = 5;

• char \texttt{myChar} = ‘t’;

• boolean \texttt{myBool} = false; // or true

• double \texttt{myDouble} = 5.5;
**Primitive types**

- **value** of primitive stored in memory

```java
int myInt = 5;
char myChar = 't';
boolean myBool = false;
double myDouble = 5.5;
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>myInt</td>
<td>5</td>
</tr>
<tr>
<td>myChar</td>
<td>'t'</td>
</tr>
<tr>
<td>myBool</td>
<td>false</td>
</tr>
<tr>
<td>myDouble</td>
<td>5.5</td>
</tr>
</tbody>
</table>

**Cats and Dogs**

**Are you a cat person or a dog person?**

- Cat person: 20 (12%)
- Dog person: 113 (67%)
- I don't like: 9 (5%)
- I like both: 27 (16%)
Object type

- Object – has state and behavior
  - Dog
    - State – name, breed, age
    - Behavior – barking, eating, sleeping
```java
public static void main(String[] args)
{
    Dog max = new Dog("Mad Max", 7, "mutt");
    max.sleeping();
}
```
```java
public class Dog {
    String myName;
    int myAge;
    String myBreed;

    void barking(){
        System.out.println(myName + " is barking. Woof Woof");
    }

    void eating(){
        System.out.println(myName + " is eating. Yum yum");
    }

    void sleeping(){
        System.out.println("shhh " + myName + " is sleeping.");
    }
}
```
```java
public class Dog {
    String myName;
    int myAge;
    String myBreed;

    void barking(){
        System.out.println(myName + " is barking. Woof Woof");
    }

    void eating(){
        System.out.println(myName + " is eating. Yum yum");
    }

    void sleeping(){
        System.out.println("shhh " + myName + " is sleeping.");
    }
}

public static void main(String[] args){
    Dog max = new Dog("Mad Max", 7, "mutt");
    max.sleeping();
}
```
public class Dog {
    String myName;
    int myAge;
    String myBreed;

    public Dog(String name, int age, String breed) {
        myName = name;
        myAge = age;
        myBreed = breed;
    }

    void barking() {
        System.out.println(myName + "is barking. Woof Woof");
    }
}

State (Instance variables)

Behavior (Methods)
```java
public static void main(String[] args) {
    Dog max = new Dog("Mad Max", 7, "mutt");
    max.sleeping();
}

public Dog(String name, int age, String breed) {
    myName = name;
    myAge = age;
    myBreed = breed;
}
```

Create an Object (Constructor)
```java
public static void main(String[] args) {
    Dog max = new Dog("Mad Max", 7, "mutt");
    max.sleeping();
}
```

Let’s try it

- Create a class `Student`
- A student has a name, age, and major
- A student has behavior “changeMajor”
  - changeMajor has a String parameter that is the new major
  - changeMajor changes the current major to the new major
- Start by creating a constructor
Arrays

- An Array – a collection of items selected by index
  - fixed number of values
  - single type

```java
char[] duke = new char[5];
duke[0] = 'd';
```

- How do you create an array of 10 Strings?

```java
a. int[] a = new String[10];
b. String[] b = new String[9];
c. String[10] c = new String[10];
d. String[] d = new String[10];
```
Arrays

• Fixed length
  • int[] a = new int[5];
  • 5 is the length

• type specific
  • int, char, String, double, Dog, etc.

• Access length of an array
  • int len = a.length;

Arrays

• Coding!
  • In your Student class
  • Add method classGrades that
    • Creates an array of 12 Strings
    • Puts the grade “A+” into every entry
    • Returns the array
Public Service Announcement

• Make APT java project
  • Add new class for each APT
  • Class / method names EXACTLY match assignment
    • Helper methods can be named anything
  • Test online many times
  • Submit APT project with ALL APTs
    • Not each APT separately

• Testing/Debugging
  • Output appears in online APT tester
  • Write your own `main`
  • Use the debugger!
APT Grading

- Complete required number by due date
- “Required” is fair game for exams / recitation
- If you skip an APT you can go back!
  - Keep trying. Hand it in later!
- You can always do more than is required!
  - This will help your grade! (2%-3%)

APT Grading

- Example

<table>
<thead>
<tr>
<th>APT checkpoint</th>
<th>APT 1 (50)</th>
<th>APT 2 (80)</th>
<th>APT 3 (120)</th>
<th>APT 4 (160)</th>
<th>Total (160)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student A</td>
<td>40</td>
<td>40</td>
<td>90</td>
<td>140</td>
<td>140</td>
</tr>
<tr>
<td>Student B</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>160</td>
<td>80</td>
</tr>
</tbody>
</table>

- If you fall behind once - make up for it next time.
  - This won’t hurt your grade.
- Not handing in APTs until the end of the semester WILL hurt your grade!
Hangman

• Assignment 1 – Hangman

• Read it!
• Snarf it!
• Ask question on Piazza!

• Next time – Hangman (read the assignment BEFORE class)