Using Classes

GEEN163 Introduction to Computer Programming

"The history of all previous societies has been the history of **`class`** *struggles."*

Karl Marx

Programming Assignment

- The first programming assignment has been posted on Blackboard under Assignments / Programming Assignments
- Upload your four Java files to Blackboard before midnight this evening
- Upload ONLY the .java files
- See the tutors in Cherry 124 or your instructor for help

Read

- Read sections 3.1 to 3.12 of the textbook

- The syllabus gives you the reading for each class period

MyCodeLab

- Answer the TuringsCraft questions for sections 2.3.1 – 2.3.5 and 3.2 – 3.7
- Due by midnight on Thursday, September 1
- You will earn 4 points for each correct answer up to a maximum of 100 points
- You can retry incorrect answers

Online Courses

- The University now has access to a large collection of online video tutorials
- You can access them at Lynda.ncat.edu
- Log in with your email ID and password

- There are several Java tutorials

Displaying Data on the Screen

- The System class can be used to display output on the console or the bottom window of jGRASP

```
int rat = 5;
System.out.println("Message "+rat);
```

- This will display

```
----jGRASP exec: java ExampleOutput

Message 5

----jGRASP: operation complete.
```

Reading Input from the Keyboard

- The java.util.Scanner class can be used to read values from the keyboard
- At the beginning of your program, you have to create a Scanner object

```
java.util.Scanner  frog = new
                          java.util.Scanner( System.in );
```

Reading an Int

- After you have created a Scanner object, you can use the nextInt() method to read an int from the keyboard

```
int toad;
toad = frog.nextInt();
```

- The program will pause here and wait for the human to enter a number

Reading a double

- After you have created a Scanner object, you can use it to read a number with a decimal point using nextDouble()

```
double tadpole;
tadpole = frog.nextDouble();
```

- The program will wait for the human to enter a number

Reading a String

- A Scanner object can read Strings
- The next() method reads one English word separated by spaces

```
String krait, cobra;
krait = frog.next();
```

The nextLine() method reads a whole line

```
cobra = frog.nextLine();
```

What reads a number into newt?

```
java.util.Scanner  toad = new
                          java.util.Scanner(System.in);
double newt;
```

A. newt = java.lang.Scanner.nextDouble();

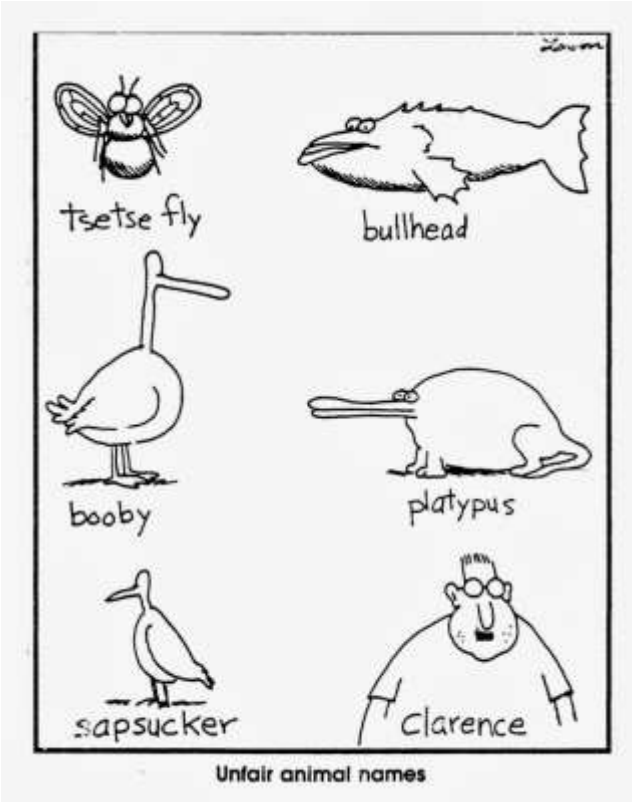
B. newt = toad.nextDouble();

C. toad = newt.nextDouble();

D. newt = toad.Scanner(System.in);

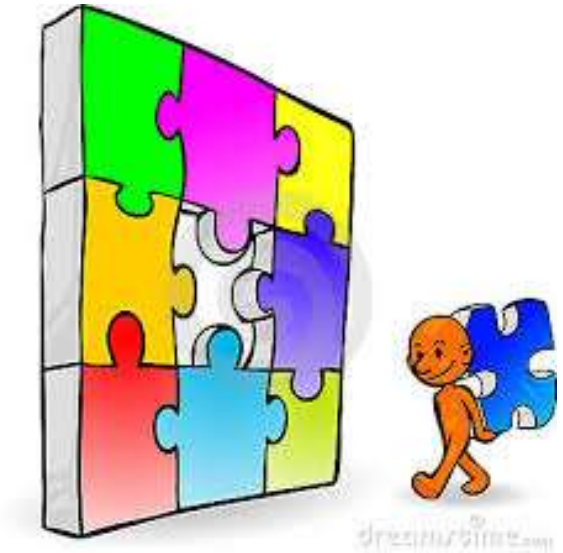
Not an animal

- Variable names should have meaning in the context of the program
- I use animal names in examples because the java segments have no context
- **keyboard** is my favorite name for a Scanner object



Problem Solving

- You have to know how to solve a problem before you can write a program to do it
- You have to know what it is you want to solve before you can write a program



Organizing a Program

- The general flow of a computer program is:
  - Read some input data
  - Calculate a value from the input
  - Display the results

- Some programs will repeat this process many times

Programs Are Sequential

- Java programs execute one line at a time
- You have to put a value in a variable before you can use it
  - A variable must appear on the left side of an equals sign before it appears on the right
- Data must be input before an equation
- Equations must execute before the output is displayed

Simple Program Structure

```
public class MyProg {
   public static void main(String[] dog) {
         data declarations
         input needed data
         calculate
         display results
   }
}
```

Input Values

- What data will the program need to solve the problem?
- Usually each input value will need a variable to hold the input
- What data type is the input?
  - int – when only whole numbers are used
  - double – when decimal values are needed
  - String – holds characters

Example Program

- Given a person's weight in pounds, tell them how much they weigh in stones

Example Program Design

- There are 14 pounds in an old English stone

- Input is the person's weight in pounds
- Output is the weight in stones

Example Program Variables

- The program will input the person's weight in pounds, so the program will need a variable to hold the pounds
- The program calculates the number of stones, so it will need a variable to hold the number of stones

```
double pounds;
```

What data type should the variable stones be?

A. int stones;
B. single stones;
C. double stones;
D. String stones;

User Friendly

- A good program will prompt the user for input
- Tell the user what to enter

```
System.out.println("Enter the weight in pounds");
```

Scanner Class

- The java.util.Scanner class can be used to read numbers and strings
- You must first create an object of the class Scanner

```
java.util.Scanner keyboard = new
                             java.util.Scanner(System.in);
```

- You should only do this once

Scanner Methods

- There are separate methods of the Scanner class to read different data types

```
int cat;    String bird;   double pounds;

cat = keyboard.nextInt();
bird = keyboard.next();


pounds = keyboard.nextDouble();
```

Calculations

- The calculations can range from trivial to extremely complex
- Often you may need additional variables to hold the results
- Sometimes you will need variables to hold temporary intermediate results

How can you calculate stones?

A. stones = pounds * 14.0;
B. stones = pounds / 14.0;
C. pounds = stones * 14.0;
D. pounds = stones / 14.0;

Displaying Results

- Display the results in an easy to understand format

```
System.out.println("weighs " + stones +" stones");
```

Typing the Program

- Most programs are not written from top to bottom
- You may occasionally need to go back to the top and declare another variable

```
/* Ken Williams
   Program to convert pounds to stones. */
public class Stones {
   public static void main(String[] rat) {
      double pounds;      // input weight in pounds
      double stones;      // calculated weight in stones
      java.util.Scanner keyboard =
                  new java.util.Scanner(System.in);

      System.out.println("Enter the weight in pounds");
      pounds = keyboard.nextDouble();

      stones = pounds / 14.0;

      System.out.println("weighs " + stones +
                                          " stones");
   }
}
```

Classes, Objects, & Methods

- Object-oriented programming uses classes, objects, and methods as basic programming components.
- These components help to
  - organize a large program into small modules
  - design and think about an intricate program
  - find and remove errors *(bugs)*

In our Java programs, we have used…

- Classes
  - In Java every program is a class
  - Many classes are already available such as Math, Scanner
  - We have used the String class
- Objects
  - An object is an **instance** of a class.

    Class: String

    Objects:

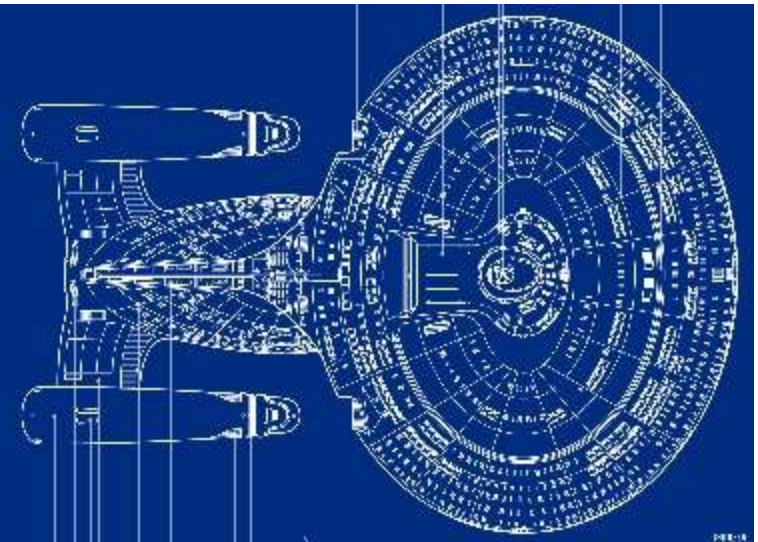
```
String  firstname, lastname, dog, cat;
```

Nomenclature

- **Class** – defines a kind of object
- **Object** – an instance of a class
- **Instantiate** – creating an object of a class
- **Instance** – An object is an instance of a class
- **Method** – program or function that an object can perform

Class or Object

- A class is a definition, blueprint or template of an object
- A class is NOT an object
- During execution, a program can create an object from a class definition



Why Use Classes?

- Usually, the data for a program is not simply one item
- Often we need to manage entities like students, books, flights, etc.
- We need to be able to manipulate such entities as a unit
- Classes allow us to separate the data for each object, while using common code to manipulate each object

Example

- *Student* class
  - Data: name, year, and grade point average
  - Methods: store/get the value of the data, graduate, etc.
- *Student* object

  Object name: *student1*

  Data: *Susan Smith, Sophomore, 3.5*

Reuse

- Once written, a class can be used in other programs
- We have used the Math class and the Scanner class, but we did not have to write them
- Library classes have been thoroughly tested
- Programs can also extend classes to add new capabilities

Declaring an Object

- Imagine we have a class **`Widget`**
- We can declare an object of type **`Widget`** just like we declare a variable to be an int or a double

```
Widget  lamb;
double  eagle;
Widget  sheep, ram;
```

- lamb, sheep and ram are objects of the type Widget

Reference Variables

- When you declare a primitive data item, such as a double or int, Java reserves some memory to store the data
- When you declare an object, Java does **not** reserve space for the object until it is created
- You can instantiate a new object with the keyword **new**

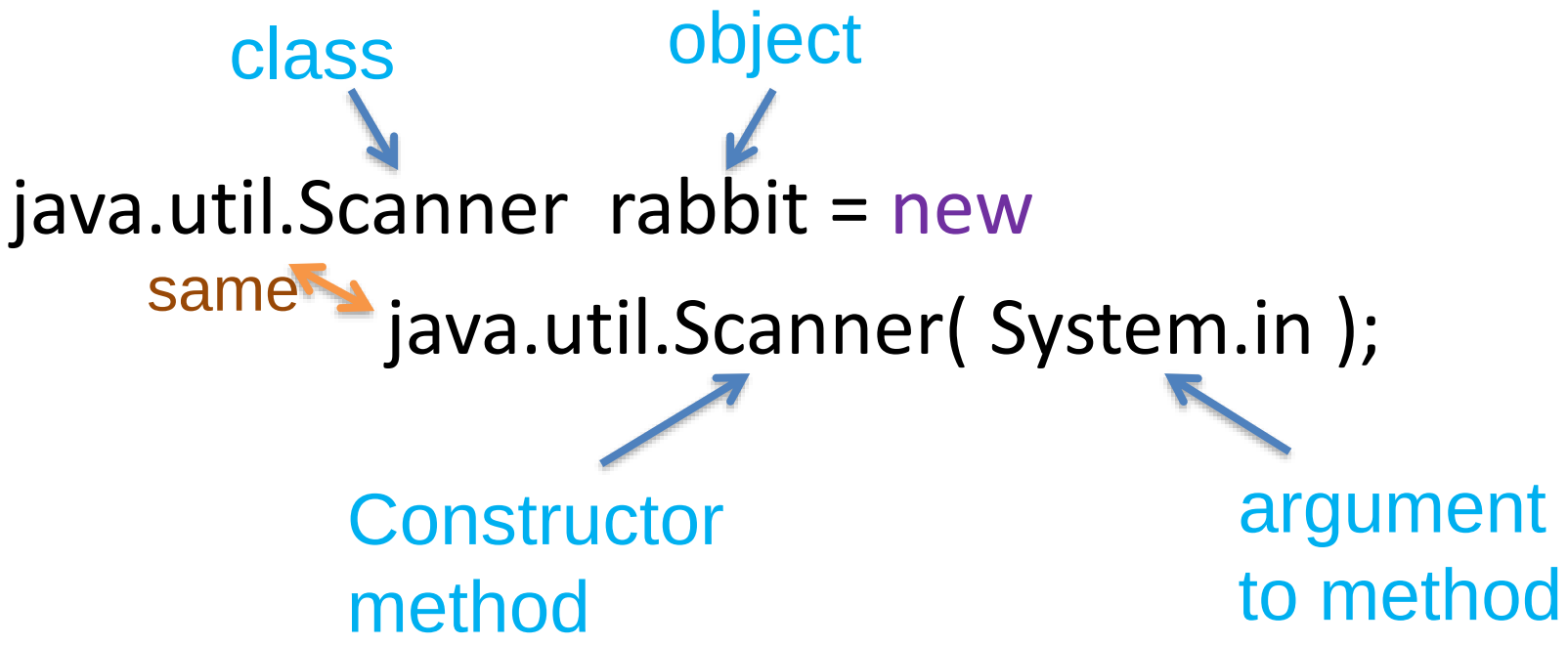
Instantiating Objects

```
Widget  gnu = new Widget();
Widget  zebra;
zebra = new Widget();
```

- After the word "**new**" is a call to a constructor method that helps create the object. The constructor method may or may not have parameters.

Instantiating Scanner Objects

- Our programs that read from the keyboard use an object of the Scanner class



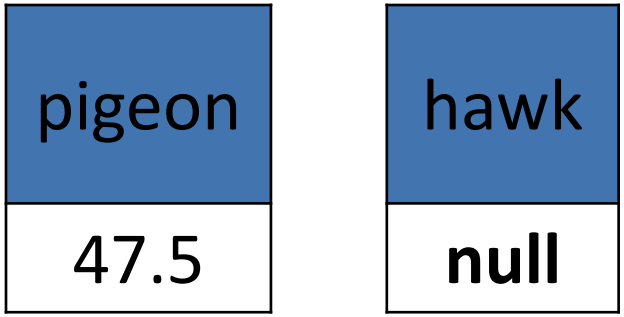
What are classes and objects?

```
java.util.Scanner cobra = new java.util.Scanner( System.in );
```

A. Scanner & cobra are classes
B. Scanner & cobra are objects
C. Scanner is a class, cobra is an object
D. Scanner is an object, cobra is a class

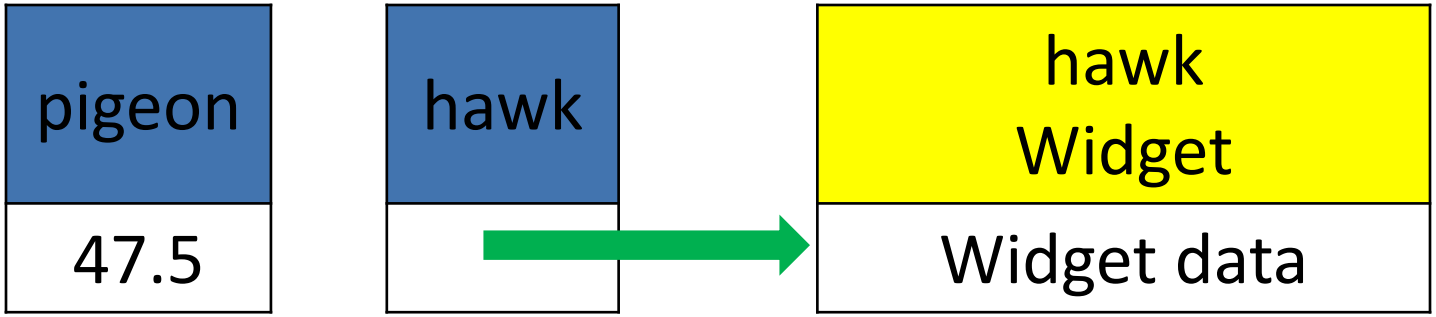
Reference Variables

```
double  pigeon = 47.5;
Widget  hawk;
```



Reference Variables

```
double  pigeon = 47.5;
Widget  hawk;
hawk = new Widget();
```



null

- null is a Java keyword that means “nothing”
- If an object reference is set to the value null, it means there is no object

```
Widget pig = new Widget();
     . . .
pig  = null;
```

Using Object Data and Methods

- You can reference a class or object's data or methods by putting a period after the class or object name
- method of the Math class

```
Math.sin( dog );
```

- data field of Math class

```
Math.PI;
```

- method of the keyboard Scanner object

```
keyboard.nextDouble();
```

Programming Assignment

- The first programming assignment has been posted on Blackboard under Assignments / Programming Assignments
- Upload your four Java files to Blackboard before midnight this evening
- Upload ONLY the .java files
- See the tutors in Cherry 124 or your instructor for help

Read

- Read sections 3.1 to 3.12 of the textbook

- The syllabus gives you the reading for each class period

MyCodeLab

- Answer the TuringsCraft questions for sections 2.3.1 – 2.3.5 and 3.2 – 3.7
- Due by midnight on Thursday, September 1
- You will earn 4 points for each correct answer up to a maximum of 100 points
- You can retry incorrect answers

Online Courses

- The University now has access to a large collection of online video tutorials
- You can access them at Lynda.ncat.edu
- Log in with your email ID and password

- There are several Java tutorials