Today’s Topics

• Comments and/or Questions?
• Searching
  – Finish implementation of Binary Search
  • figure out the condition of the loop
• Object Oriented programming
Binary Search

// method to perform binary search of an array
public static int binarySearch( int array2[], int key )
{
    int low = 0;                  // low element subscript
    int high = array2.length - 1;  // high element subscript
    int middle;                   // middle element subscript

    // loop until low subscript is greater than high subscript
    while ( low <= high )
    {
        // determine middle element subscript
        middle = ( low + high ) / 2;

        // if key less than middle element, set new high element
        else if ( key < array2[ middle ] )
            high = middle - 1;

        // key greater than middle element, set new low element
        else
            low = middle + 1;

        // if key matches middle element, return middle location
        if ( key == array2[ middle ] )
            return middle;

    } // end while loop

    return -1;   // key not found

} // end method binarySearch
Searching arrays (Binary search)

• Let’s analyze the binary search.
• To simplify the discussion, we can count the 2 comparisons in the if/else/if/else together to be 1 comparison.
• How long (that is, how many comparisons) does it take to find the value?
  – What’s the minimum number of comparisons it would take?
  – What’s the maximum number of comparisons it would take?
Object orientation

- Object orientation is a programming paradigm that views the pieces of large programs as objects with “attributes” and “behaviors”.

- Java is an object-oriented language. C++ is another widely used object-oriented language. Other languages with which you may be familiar, like C, Pascal, and Fortran, are not object-oriented. These are what are called procedural languages.

- So far in this course, we have used Java in a very procedural way and did not use it to its potential as an object-oriented language.
Object orientation

• An example of an object.
• If we wanted to write a program that worked with rectangles:
  – The attributes of a rectangle include its length and width.
  – Other attributes might be its color, ...
  – Behaviors of the rectangle could include: changing its size, computing its area, etc...
Object orientation

• So, in our example, a rectangle would have as its data:
  – length
  – width
  – color

• As its methods we would have things like:
  – setLength
  – setWidth
  – setColor
  – calculateArea
Object orientation

• This translates to creating a class called Rectangle which has length, width and color as instance variables. This class will represent a generic rectangle.

• The methods in the class Rectangle would be:
  – setLength
  – setWidth
  – setColor
  – calculateArea
  – among other possible methods
Class Rectangle

• The Rectangle class would be defined in its own file called Rectangle.java

• The Rectangle class represents what it means to be a rectangle -- it means having a length, width, and color and it means that we can change its length, width, and color. We can also calculate the area of a rectangle. If we wanted to be able to do more things to a Rectangle then we could add more methods such as calculating the perimeter of the rectangle.
class definition of Rectangle

• Let's partially create the class Rectangle.
• Essentially we are creating a new data type named Rectangle to represent rectangles.
• Then we can use this type as the type of variables.
  – A variable of type Rectangle is considered an object reference of type Rectangle
Object of type Rectangle

• An object of type Rectangle is ONE rectangle with a specific length, width, and color. An object of type Rectangle must be created to be able to use the data and methods in the Rectangle class.

• To do this, in some other class, we could instantiate an object of type Rectangle, by declaring a variable of type Rectangle (and calling its constructor – we'll get to that shortly).

• Then, through this object reference, we would have access to the methods within the Rectangle class to set its length, set its width, compute its area, etc... for that particular instance of the class.

• We might want multiple variables of type rectangle, each with their own width, length and color. These multiple variables would each be objects of type Rectangle.
Constructor methods

• A constructor method is a special method of a class that is called when an object of the class is instantiated (created.)
• Constructor methods usually initialize the instance variables.
• Constructor methods have the same name as the class and are defined WITHOUT a return type.
• A constructor for our class Rectangle should contain code that sets the values of length, width and color.
Constructor methods

- for example, we might have a constructor that looks like the following:

```
public Rectangle ( double len, double wid, String col )
{
    length = len;
    width = wid;
    color = col;
}
```

- Recall that the instance variables were named length, width and color that is why they are on the left side of the assignment statements --- they're the variables being set.
Constructor methods

• We might also want a default constructor (one that takes no arguments) so that we can create an object of type Rectangle without specifying its length, width and color.

```
public Rectangle ()
{
    length = 1.0;
    width = 1.0;
    color = "Green";
}
```

• This says the default rectangle is 1 by 1 and Green.
Constructor methods

- Assuming that the two constructors just described are defined within the Rectangle class, to instantiate an object of type Rectangle we would do one of the following:

// this would create a rectangle with the specified values.
Rectangle someRect = new Rectangle (3.2, 5.1, “Red”);

// this would create a “Green” rectangle with length = 1
// and width = 1
Rectangle anotherRect = new Rectangle ( );
example

• Let’s create a program containing two classes
  – Rectangle
    • let's add the code from the previous slides into Rectangle and also provide a method that returns the information about the rectangle (its length, width and color) as a String.
  – RectangleTester
    • will contain the main method and instantiate a couple of objects of type Rectangle with which we'll call the methods of Rectangle.
Programmer defined classes can be used as types, just like classes in the Java API can be used as types (e.g. String.)

Objects have attributes (data) and behaviors (methods.)

The data are the *instance variables* in a class.

An object of a class is instantiated by declaring a variable of that type and initializing the object by calling a constructor of that class with the new operator.

- e.g. `Time timeObject = new Time();`

here, we are creating an object (timeObject), of a programmer defined class (Time).
Let's create a class Time

• What instance variables might we have to represent a time?
• What types will they be?
• What are the valid values of each of these variables?
• We can make sure that these variables ALWAYS have valid data in them
  – make the instance variables private
  – and in any method that can change the value of an instance variable make sure to not allow it to have an invalid value