Using a Scanner in Java (The Scanner Class)

- An important aspect of programming is receiving information from the user of the program. Sometimes we also want to read information from files.

- Setting up to use a Scanner: The Java library classes are divided into different packages. Up until this point, we have been using classes from the java.lang package, which is the basic package we automatically have access to in a Java program.

- When we want to use classes from a different package, we have to import the class from a package:

import packageName.className;

We may also import the entire package:

import packageName;

Though that’s usually not recommended if we’re not using a lot from the package.

- The import commands will go at the top of our program file above the class declaration line. The scanner class happens to be a part of the java.util package.

import java.util.scanner;

- The scanner:

Scanner input;

-Constructor: Note there is not a default constructor.

The scanner class has a variety of constructors but for our example we’re just interested in the one where we can read from the keyboard.

Scanner(InputStream dataSource);

Connecting to the keyboard:

- We need to give our scanner object an input stream connected to the keyboard. Java automatically provides us with such a thing. The general term for this in all programming languages is standard input.

- The specific object that Java provides is

System.in

Which is an object of type input stream.

- So our complete declaration-initialization for a scanner object to read from the keyboard will be:

Scanner input = new Scanner(System.in);

Scanner: class

Variable name: input

- Now that we have our scanner object, we need to know how to use it. We’ll start by saying how to read in an integer. We have a nextInt method that returns an int that is the number read in.

Example:

int nextInt( )

int age; (Declare the variable)

System.out.print(“Please enter your age: “) ;

age = input.nextInt( );

Notice we used ‘print’ and not ‘println’. It usually looks nicer if we can end our request for the data with a colon space and then have our user typing on the same line.

We also never want to have the user typing without first asking for a specific piece of data.

- The scanner provides two methods to read Strings.

String next( )

It basically returns the next word in the input. It skips whitespace characters (spaces, tabs, newlines) and then stops at next whitespace characters.

- Our next option is

String nextLine( )

Which returns a string up until the next newline character. So if we want to include whitespace characters, this is the method we will want to use.

WARNING: Mixing nextLine( ) with other methods can cause issues.

next( ), nextInt( ), etc. leave the rest of the line still to be read. They leave it there for the following method to deal with. Subsequent nextLine( ) reads the rest of that line. Not the next line we are hoping for.

- Getting a char from a Scanner: There is no method to read a char. We must use next( ) and then get the first character out of a String.

Example:

String tempString = input.next( );

char theChar = tempString.charAt( 0 );