Project 7

CS 1323, Fall 2015

# Learning Objectives

1. Declare and construct an array of String objects where the size is known in advance. (20 points)
2. Perform spelling checking using a standard dictionary. (20 points)
3. Use the binarySearch() method in the Arrays class. (10 points)
4. Use a for loop. (10 points)
5. Create three meaningful methods. (20 points)
6. Import two new packages. (10 points)

10 points will be awarded for the documentation of your program. That means using good names for variables, comments, proper and consistent indentation of code, and meaningful use of whitespace.

Section 10: When your program is completed and running, have the teaching assistants check it to get credit for the lab. If you do not complete the laboratory during the allotted time, you may submit it on Janux before Monday, October 12 at 11:59 p.m. Only people who attend the whole laboratory will be permitted to submit assignments on Janux.

Sections 1 and 995: Submit your .java file on Janux before Monday, October 12 at 11:59 p.m.

Remember that submissions to Janux are supposed to be in a zip file.

# Description

Many software programs have a spelling checker. Spelling checkers work by comparing each word in a given file to an existing dictionary to see if the word is in the dictionary. If it is, it is determined to be spelled correctly. If it is not, then it is not spelled correctly. This strategy has numerous well known shortcomings, particularly that the dictionary cannot tell if the word is being used properly, just if it was somewhere in the dictionary. So if you type “here” when you meant “hear” a spelling dictionary will not be able to detect the problem.

In this lab, you will implement a spelling checker that reads from a standard dictionary file. You’ll need to find the length of the dictionary so you can use it to create your dictionary array.

The method signature is:

public static int findSizeOfDictionary(String fileName)

Then you will read the words from the dictionary into the array. The words in the dictionary are in sorted order, so this is a big advantage.

The method signature is:

public static String[] readFile(String filename, int fileSize)

The program should then read words from the console using a Scanner and print out a warning if the word is not in the spelling dictionary. Make sure you take advantage of the fact that the dictionary is sorted by using binary search.

The method signature is:

public static void spellcheck(String[] dictionary)

The global dictionary file is stored in Dictionary.txt and is available on Janux. The best place to put this is in the Project main directory. If you put it anywhere else, you will have trouble finding it when you open the file.

Sample interaction is shown below:

Enter a word or QUIT to stop:

airplane

That word is spelled correctly

Enter a word or QUIT to stop:

amoebae

That word is spelled correctly

Enter a word or QUIT to stop:

goodbye

That word is not spelled correctly

Enter a word or QUIT to stop:

quit

This particular dictionary is pretty odd, so you may be surprised about what is and isn’t spelled correctly.

# Files

In order to do this project, you will need to be able to read a file. Luckily, the Scanner class handles files just as well as the keyboard input.

To open a file:

String filename = “Dictionary.txt”;

Scanner file = new Scanner(new File(filename)); // make the contents of the file available to read

Remember the new File part above. Without it, the Scanner will open the String filename and nothing will work properly.

To read data from a file, you use a Scanner just like we did with the console. I’m showing a String being read below, but all other types of data can be used as well.

String data = file.nextLine();

When you are through with a file, you close it (although Java 8 does this for you if you forget).

file.close();

Files have only a finite amount of data. So to tell if there is more data in the file, you will use:

while (file.hasNextLine())

{

// body of loop goes here

}

You’ll need to find the packages to import using the Java API. Look at the Scanner class as an example if you have trouble doing this.

Java considers using files to be risky business—and it is! Anytime you go to open a file, it’s possible that the file won’t be there. Therefore, any method that opens a file has to warn everyone in the world that things might go wrong. This is done by an announcement called a throws clause.

public static int findSizeOfDictionary(String fileName) throws FileNotFoundException

In addition to this method, the main method will also have to throw a FileNotFoundException, since the findSizeOfDictionary method will be called in the main method. Eclipse will help you figure out which methods need this clause.