Assigned in: Lesson 5

Due before: Lesson 6

Upload your finished lab to the Learning Hub (Activities / Lab 4) before the start of lesson 6.

1. Create a functional interface called Wordable with the abstract function the following signature:

**String createString(String s, int n)**

2. Create a class called Dictionary.

Dictionary has the following List of words found at the end of this document (English words that begin with the letters "ja").

Dictionary has a method with the following signature:

**public String getWords(String word, int number, Wordable w)**

3. Create a class called Main.

Main has a **main()** method which does the following:

a) creates a Wordable variable called wordy and assigns it a lambda expression as follows:

b) If the first commandline argument is "concat" then Wordable returns a String with all the words concatenated together; for example: "jabjabbedjabberjabbered..."

c) If the first commandline argument is "repeat" then Wordable returns a String with all the words concatenated together, each word repeated args[1] times (the second commandline argument must be a number, which can be read by the code Integer.valueOf(args[1])); for example, if args[1] is 2, Wordable returns "jabjabjabbedjabbedjabberjabberjabberedjabbered..."

c) If the first commandline argument is "nth" then Wordable returns the nth word from your list of words (the second commandline argument must be a number - the index; for example, if args[1] is 6, Wordable returns "jabberwocky", given the following list:

The list of words (see below) is:

index 0: jab

index 1: jabbed

index 2: jabber

index 3: jabbered

index 4: jabbering

index 5: jabbers

index 6: jabberwocky

index 7: jabbing

etc...

d) If the first commandline argument is "reverse", then Wordable returns a String with all the words concatenated together but each word is in reverse order; for example: "bajdebbajrebbajderebbaj...". The StringBuilder class can easily reverse Strings.

e) At the end of main(), print the String returned by Wordable. Then use a method reference to print the entire list of words.

f) Also create a static method in the Dictionary class called reverseString, which reverses the characters in a given string. Then apply this method reference in the Main class within a loop to reverse each word in the list and print the result.

g) In main(), create an array of strings from the list of words and then use Arrays.sort with a method reference to a custom comparator method in the Dictionary class. This comparator method, called alphabeticalOrder, must sort words in alphabetical order.

h) Finally, use method references to perform conditional checks. Define a static method in the Dictionary class called isLengthAboveFive which checks if a word’s length is greater than five. In the Main class, loop through the list of words, use this method reference to check each word, and print only those words that satisfy this condition using System.out.println as a method reference.

(See examples for f-g-h after the list of words, below).

Here is the list of words you must use:

jab

jabbed

jabber

jabbered

jabbering

jabbers

jabberwocky

jabbing

jabot

jabots

jabs

jacaranda

jacarandas

jacinth

jack

jackal

jackals

jackanapes

jackanapeses

jackass

jackasses

jackboot

jackboots

jackdaw

jackdaws

jacked

jacket

jacketed

jacketing

jackets

jackhammer

jackhammers

jackie

jacking

jackknife

jackpot

jackpots

jacks

jackson

jacksonville

jaclyn

jacob

jacobean

jacobian

jacobin

jacobins

jacobite

jacobites

jacobs

jacquard

jacquards

jacqueline

jacques

jactitation

jactitations

jactus

jacuzzi

jacuzzis

jade

jaded

jadeite

jadeites

jades

jading

jaeger

jaffa

jag

jagged

jaggedly

jaggedness

jagger

jags

jaguar

jaguars

jai

jail

jailed

jailer

jailers

jailhouse

jailing

jailor

jailors

jails

jaipur

jakarta

jake

jalopies

jalopy

jalousie

jalousies

jam

jamaica

jamaican

jamaicans

jamb

jamboree

jamborees

jambs

james

jamey

jamie

jammed

jamming

jammy

jams

jan

jane

janeiro

janet

jangle

jangled

jangles

jangling

janice

janitor

janitorial

janitors

january

januarys

janus

japan

japanese

japanned

japanning

japans

jape

japed

japer

japers

japery

japes

japing

japonica

japonicas

jar

jardiniere

jardinieres

jarful

jarfuls

jargon

jargons

jarred

jarring

jarringly

jarrow

jars

jasmine

jasmines

jason

jasper

jaspers

jaundice

jaundiced

jaunt

jaunted

jauntier

jauntiest

jauntily

jaunting

jaunts

jaunty

java

javanese

javelin

javelins

jaw

jawbone

jawbones

jawboning

jawbreaker

jawbreakers

jawed

jawing

jaws

jay

jays

jayvees

jaywalk

jaywalked

jaywalker

jaywalkers

jaywalking

jaywalks

jazz

jazzed

jazzes

jazzier

jazziest

jazzing

jazzman

jazzmen

jazzy

Examples for requirements f-g-h:

public class Dictionary {

// Assuming other existing methods and properties...

// Method for reversing a string (Task f: String Manipulation)

public static String reverseString(final String s) {

return new StringBuilder(s).reverse().toString();

}

// Method for sorting words in alphabetical order (Task g: Array Operations)

public static int alphabeticalOrder(final String word1, final String word2) {

return word1.compareTo(word2);

}

// Method for checking if word length is above five (Task h: Conditional Operations)

public static boolean isLengthAboveFive(final String word) {

return word.length() > 5;

}

}

…and to call these from main()…

// Task f: Apply reverseString method reference to each word and print

for (String word : words) {

String reversed = Dictionary.reverseString(word);

System.out.println(reversed);

}

// Task g: Sort words using the alphabeticalOrder method reference

Arrays.sort(words, Dictionary::alphabeticalOrder);

System.out.println("Sorted words: " + Arrays.toString(words));

// Task h: Print words longer than five characters using method references

for (String word : words) {

if (Dictionary.isLengthAboveFive(word)) {

System.out.println(word); // or use System.out::println directly in a method call

}

}