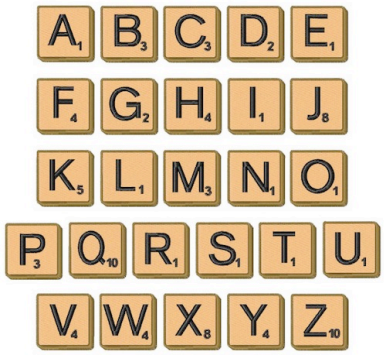
Mr. Cochran
AP Computer Science

Lab 03.02: *SCRABBLE*® Scorer

Lab Description:

The purpose of this lab involves several components:

1. Programmers must understand the concept of parallel lists.
2. Programmers must understand what a lookup table is: http://en.wikipedia.org/wiki/Lookup_table
3. Programmers must understand the concept of O-notation for measuring algorithm efficiency and why searching a sorted set is more efficient than searching an unsorted set.
4. Programmers must know how to parse string values one letter at a time.



Guidelines for Solutions:

To solve this problem, you must:

- Create a Java class and name it `ScrabbleScorer.java`.
- Use iteration to parse the user-input word one letter at a time
- Use an `int` array called `points` to store the scoring value for each letter
- Use an `ArrayList` of `String` values called `dictionary` that checks and validates user input from an imported text file (use the supplied `SCRABBLE_WORDS.txt` file)
- Create some sort of invalid input handling statement to confirm diabolic input and prompt for re-entry
- Loop the program indefinitely until the user enters 0

Sample input/output:

```
Enter a word to score or 0 to quit: quiz
quiz = 22 points
Enter a word to score or 0 to quit: abc
abc is not a valid word in the dictionary
Enter a word: hi there
hi there is not a valid word in the dictionary
Enter a word to score or 0 to quit: abc123
abc123 is not a valid word in the dictionary
Enter a word to score or 0 to quit: hello
hello = 8 points
Enter a word to score or 0 to quit: 0
Exiting the program thanks for playing
```

Point Values for Scrabble Letters:

Use the following letter scoring algorithm to score the word entry values:

A – 1	E – 1	I – 1	M – 3	Q – 10	U – 1	Y – 4
B – 3	F – 4	J – 8	N – 1	R – 1	V – 4	Z – 10
C – 3	G – 2	K – 5	O – 1	S – 1	W – 4	
D – 2	H – 4	L – 1	P – 3	T – 1	X – 8	

Required Methods:

Your version of `ScrabbleScorer.java` must contain the following methods:

```
public ScrabbleScorer()
public void buildDictionary()
public boolean isValidWord(String word)
public int getWordScore(String word)
public static void main(String[] args)
```

Handling User Input:

Your application should be driven by a do-while loop in main. The user should be prompted for input and given an exit input value as well (see the previous page of this assignment on what the user should enter to quit the application). Once the user has entered a string value, the application should check that entry against the pre-built dictionary based on the `SCRABBLE_WORDS.txt` text file. Your application must check the user's entry to make sure it is a valid, playable Scrabble word.

Improving Your Algorithm's Efficiency:

Your program must do two things to increase the execution speed of this algorithm.

1. Once the `SCRABBLE_WORDS.txt` file is successfully imported into the `dictionary`, you must call `Collections.sort` on the `dictionary` item to have all of the Scrabble words sorted in lexographic ascending order.
2. The method `public boolean isValidWord(String word)` must use a call to `Collections.binarySearch()` to determine if the parameter `word` is contained in the dictionary object.

Documentation Requirements:

Your version of `ScrabbleScorer.java` must include the following comments/documentation:

1. A standard 3-line Javadoc header comment (name, due date, explanation of the lab)
2. Method description, `@param` and `@return` Javadoc documentation for each method in your program, including the class constructor and `main`

What to hand in:

Please submit the file `ScrabbleScorer.java` through the Veracross drop box for this assignment.

Grading rubric:

Description	Points
`ScrabbleScorer.java` contains correct 3-line Javadoc header comment	3
`ScrabbleScorer.java` contains Javadoc method documentation for each of the five required methods	5
`ScrabbleScorer.java` correctly builds the dictionary and sorts all of the entries using `Collections.sort`	6
`ScrabbleScorer.java` uses `Collections.binarySearch()` to validate user input in O(log N) time	6
`ScrabbleScorer.java` compiles and executes without any runtime errors	6
`ScrabbleScorer.java` produces the exact input/output as specified on the previous page of this lab description, correctly handling invalid/diabolic input.	8
Student submits the correct file that is properly named to the Veracross drop box, meeting the lab submission deadline	6
TOTAL POINTS available for Lab 03.02 Scrabble Scorer	**40 points**

Honor Code policy:

You may ask a classmate for help with your code and any associated algorithms, but you may not directly share your code with a classmate. Sharing code in any manner (email, texting, printed copies, etc.) as well as precise and exact copying of a classmate's code is considered a clear-cut violation of Durham Academy's Honor Code. You may also use code or algorithm assistance found online. If you ask someone for help or look at a program online, you must list the names of your classmate(s) with whom you worked or put the URL of the example project in the header comment of your program. All programs will potentially be examined for copied code using the Unix `diff` utility program. If your code is very close to that of a classmate, at a minimum you will have a discussion with your teacher. *Most violations of this policy in years past have been sent to the school's Honor Council for additional investigation.*