Assignment 2: Find palindromic words

Specification:

Develop a parallel multithreaded program to solve the following problem. There is A dictionary file "**words.txt**", which contains 25,143 words. A palindrome is a word or phrase that reads the same in either direction, i.e., if you reverse all the letters you get the same word or phrase. Your task is to find all palindromic words in the dictionary. A word is palindromic if its reverse is also in the dictionary. For example, "noon" is palindromic, because it is a palindrome and hence it's reverse is trivially in the dictionary. A word like "draw" is palindromic because "ward" is also in the dictionary. Do the input and output phases sequentially and the rest in parallel. Your program should write the palindromic words to a result file.

Implementation details:

Palindrome main file reads the dictionary file and creates a bag of tasks (which is a Hashmap in our case). Hashmap contains **key, value** pairs. In our case all the words which have same length, they will be assigned same key and key will be the length of the word.

HashMap<**String**, Array**List<String>**>

Here **String** is type for **Key** (in our case, it will be length of the word)

ArrayList is type for **Value** (in our case, it contains all the words which have same length)

So all the words having same length, they will be under the same key.

Example code:

HashMap<**String**, **ArrayList<String>**>   mapBagOfTasks = HashMap<**String**, **ArrayList<String>**> ();

ArrayList<String> arrFirstBag = new ArrayList<String> ();

//words of same length will be put in same ArrayList.

//For example, let suppose we put all words of the same length (let say of length 4) in arrFirstBag.

Now put this arraylist in map where key will be 4.

```
mapBagOfTasks.put(4, arrFirstBag);
```

In this way we have prepared a bag of tasks.

After we have bag of tasks, there is need to spawn worker threads. We create w (input by user) worker threads. (`PalindromeWorker.java`)

Pass bag (one or more) and a **shared ArrayList (sharedArr)** to each created thread.

Concurrently, There is another writer thread(`PalindromeWriter.java`), which polls palindrome words from the **shared ArrayList (sharedArr)** and writes in output file.

The worker threads, which find the palindromes from their assigned task, they put these palindrome words in the **shared ArrayList (sharedArr)** in critical region.

Shared ArrayList (sharedArr) object which is accessed by worker threads and writer thread is shared resource and it has locking mechanism whenever data is put or polled.

When a worker threads find its bag empty, its writes the palindromes count (e.g., “count_25”) to the sharedArr.

When all the threads finish their work, the writer has all the information about the activity done by all threads and it writes the information in the file.

Note: Don’t use built in semaphores or any other built in constructs for synchronization.

Wherever needed, you can use synchronized keyword (with method or blocks of code) for synchronization.

Sample output:

The your result file output should be similar to “results.txt”. Your program should also print how much each thread has found palindromes. For example as shown in file “result.txt”

```
Worker name: Thread 5,  palindrome_count: 284

Worker name: Thread 4,  palindrome_count: 15

…….

Total count for Palindromes = 299
```