CS342 Spring 2015
Project 2
Multi-threaded Programs

Assigned: 02.03.2015
Due date: 14.03.2015, 23:55

Write a multi-threaded C program that will count the number of occurrences of words in an input set of text (ascii) files. There will be N input text files (indexed as 0..N-1) containing words of alphanumerical characters (N can be at most 100). An alphanumerical character can be a letter (lowercase or uppercase) from English alphabet or a digit (0-9). The maximum length of a word can be 255 characters. The names of the files will be taken from the command line.

Each input file will be processed by a mapper-thread that will read file content and partition the words into R intermediate temporary files (R can be at most 50). The partitioning rule is the following: a word w coming from an input file j (0<=j<=N-1) will go to an intermediate file "tempj-i" where i = hash(w) mod R. Here 0<=i<=R-1. The hash(w) will be found by summing the bytes of w. If w is "the", for example, the three characters, casted to integer, will be summed. In this way the words in an input file are partitioned into R files (at most). As a result, the same word may appear many times in an intermediate file. Each word will appear in a separate line of an intermediate file. Since each of N mapper threads can produce at most K files, there can be at most N*K intermediate files produced.

There will be R reducer-threads. Each reducer thread will perform counting. A reducer thread will read N partitions (N intermediate files), sort the content read according to strcmp(word), find the count of occurrences of each unique word, and will emit to a temporary output file the unique words and their counts. For example, the reducer thread 0 will read intermediate files (partitions) temp0-0, temp1-0, temp2-0, …, temp(N-1)-0. The reducer thread 3, for example, will read the files temp0-3, temp1-3, …, temp(N-1)-3. A reducer thread may emit <word, count> pairs as follows (one pair per line).

```
ankara      7
bilkent    10
computer    3
science    18
```

At the end, R temporary output files will be produced. Then a merger thread will merge these files and and will produce a single final file that contains all unique words and their counts in sorted order (use strcmp for comparing two words). The name of the final file will be taken from the command line as the last argument of the program. Name your program as wcount. It will be invoked as follows:

```
wcount <N> <R> <infile1> … <infileN> <finalfile>
```

An example invocation is:

```
wcount 3 2 infile1.txt infile2.txt infile3.txt final.txt
```

Experiments: Do some experiments. Measure the time of execution, for example, for various input sizes, input data, N and R values. Draw diagrams, tables, or charts. Try to explain the results and try to draw some conclusions. Try to think and design some other experiments.

Submission: Submit through Moodle. Put your wcount.c file, a Makefile, a report.pdf file, and a README file into a project directory; tar and gzip the directory; and upload your project2.tar.gz file.