CSE 3318 Lab Assignment 2

Due October 13

Goals:

1. Understanding of heaps.

2. Understanding of merging.

Requirements:

1. Write a C program to take `n` files containing strings in ascending order (no duplicates within a file) and produce a file `out.dat` containing a line for each string (in ascending order). Even if a string `str` appears in multiple files, it should be output *only once* and, for each string, you should also output the number of files (`k`) containing the string. This should be done using code similar to:

   ```
   fprintf(outfp,"%s %d\n",str,k);
   ```

2. Submit your program on Canvas by 5:00 pm on Thursday, October 13. Comments at the beginning of the source file should include: your name, your ID number, and the command used to compile your code on Omega (5 point penalty for non-compliance).

Getting Started:

1. Your program is to perform exactly one "heap assisted" merge of all `n` files simultaneously. At any time, there should be no more than one string from each of the input files being processed by your code. It will be useful to have a table of file pointers and a table of strings. Your heap implementation is not required to have "handles".

   Under no circumstance should your program use multiple binary merges!

2. You may use heap code (e.g. `intPQi.c`) from the course webpage or elsewhere to get started. Be sure to include comment(s) regarding this.

3. Your program will be driven by a file `in.dat`:

   a. The first line will contain the value for `n`.
   b. Each of the remaining `n` lines will contain a simple file name, i.e. there will not be a directory path.
   c. Each of the `n` files will contain at least one string. The strings will consist of no more than 50 letters and digits.

4. Pseudo-code:

   a. Open `in.dat`, each of the n files, and `out.dat`.
   b. Initialize the heap with the first string from each file. The strings will be the priorities, so you will have a minHeap with the smallest (`strcmp()`) string conceptually at the root.
   c. Processing to obtain the first output string:
      1. Remove (conceptually) the minimum string from the heap.
      2. `k=1`
      3. Attempt to read in another string from the same file as the string just removed.
      4. Put new string in heap (special case: there is no new string)

d. While at least one file has not been exhausted:
1. Remove (conceptually) the minimum string from the heap.
2. if the minimum string is different from the previous minimum
    Output the previous minimum and k . . .
   else
    k++
3. Attempt to read in another string from the same file as the string just removed.
   if EOF
    Heap gets smaller
   else
    Put string in heap
e. Final clean-up . . . including output of the last string.

in.dat	in0.dat	in1.dat	in2.dat	in3.dat	out.dat
4	abc	abc	def	efg	abc 2
in0.dat	ghi	efg	ghi	ghi	def 1
in1.dat	hij	ghi	mno	mno	efg 2
in2.dat	jkl	q	rst	rst	ghi 4
in3.dat	mno	w	x	w	hij 1
	p			y	jkl 1
	uv				mno 3
	w				p 1
					q 1
					rst 2
					uv 1
					w 3
					x 1
					y 1