CS 3423.02
Assignment 4: Implementing Word Count using C language
Due: 12:05am of 11/10/2014

1 Overall Description:

Find out the occurrences of each word (A Word Count Example) inside a directory or a file. Those files will be normal text files. Depending on whether you are using your designed doubly link list or not, you may turnin different files. If you are using your designed doubly link list, then you should turnin the following files.

`wordcount.c, dllist.c and dlist.h.`

Since changing your existing code will be a little bit difficult than using existing sTL link list, I am planning to give 20% bonus if you are taking this approach. Thus, if you get full points on this assignment, you will be getting 120 points in total.
If you are using the STL library's link list, you will only turnin the following files. Please refer to the following description about how to utilize the STL's link list, `http://www.cplusplus.com/reference/list/list/`.

`wordcount.cpp`

This program can accept three input parameters.

```
./wordcount INPUT NUMBER OUTPUT
```

- The INPUT parameter can be a directory or a file. If it is a directory, we should find out all regular files inside the given directory and count all words for these files. If it is a file, then we only count the occurrence of words on this file.
- The NUMBER parameter is used to tell the program to output the NUMBER of words with most occurrences.
- We will redirect all words related information into a OUTPUT file, but only display the NUMBER of words on the screen. But those words has to be sorted at first before installing into the OUTPUT file.
- Each line in the OUTPUT file will only hold information of one word: the occurrences of this word, the word itself. These two fields are divided using the Table space, printing using the "/t". Those information is similar to the last two assignments.

1.1 Description of words

Checking the words is similar to what we did before.

- There is no difference between uppercase and lowercase. For example, "We", "we" and "WE" are considered the same and should be counted together. Thus, in end of output, we only output those number of occurrences of lowercase. If we have ("we", "we", "WE"), we will output:

```
3 we
```

- We focus on those words made out of the alphabet, or connecting with hyphens. For example, "we're" will only count "we" here while "'re" will be discarded. However, "accident-prone" will be counted as a word.
- We can simply discard the following cases.
  - Anything that are not consisted of letters. For example, words using with an apostrophe (') or even period. For example, "We all in this class.", there are only four words inside: "we all in this". "class." will be discarded.
  - Words made with numbers. For example, 1234 and 1234a will be discarded.
  - HTML tags

2 Implementation Tips

- You may use `stat` to check whether the input is a file or a directory. See more details at `http://stackoverflow.com/questions/4553012/checking-if-a-file-is-a-directory-or-just-a-file`. Then you may use opendir() and readdir() find every entry inside one directory
- You may use the following library functions: open(), opendir(), read(), write(), stat(), readdir().
- You may use those memory related functions like malloc() and free().
- You have to think about how to sort those words according to their occurrences. You may use another link list or any data structure to hold words and their counts when you are trying to sort them in the end. But you are free to use any way your want.

2.1 Using dllist.c in Assignment3

If you are using your designed dllist.c of Assignment3, you have to change the data structure a little bit for this assignment.
Now valueT has to changed in order to hold two pieces of information. Here is one suggestion on doing this, but you can design your data structure here.

```
typedef struct wordCount {
   char * word;
   int    count;
} wordcount_t;

typedef wordcount_t * valueT;
```

Some of your code has to be changed too. For example, you may change the code whether a value is existing or not. But you have flexibility to design this project.

2.2 Using the STL list

You will have to use g++ to compile your code, instead of gcc. Otherwise, the compiling can fail with the following warnings:

```
undefined reference to `__gxx_personality_v0'
```

You can use all references that you can find to finish this homework. But you can't copy the code from anywhere.

3 General Rule

- All programs should be able to handle those errors in the first place. Failing to do this will loose some points. For example, if an input has a problem, your program should be able to point out the errors.

  ```
  printf("ERROR: YOUR_SPECIFIC_DESCRIPTION\n");
  exit(1);
  ```

  Since we are going to check your error report using script, then you should use the specific format for this error report. The error report will always start as

  `ERROR:`

- Please pay attention to readablity of a program. You may refer to some coding style listed at `http://www.maultech.com/chrislott/resources/cstyle/indhill-annot.html`.
- You can change your code in assignment3 to suit for this assignment. I will not test on dllist.c any more. Thus, you are free to do anything you want.

3.1 Suggested Tests:

Make sure that you can verify your scripts using the following test cases:

1. The number of arguments is not correct. We only accept three arguments.
2. INPUT is the same as OUTPUT.
3. OUTPUT is already existing.
4. INPUT is a file.
5. File name, either INPUT or files in the specified directory, can include space in the middle.
6. Directory may have subdirectories.

4 Submission Requirements

In this assignment, you will have to turnin two parts.

1. If you are using your designed linklist, you will turnin wordcount.c, dllist.c, dllist.h and Makefile. wordcount.c is the main program. If you are using STL link list, you only need to turn in wordcount.cpp and Makefile. (70%)
2. WordcountDesignForC.txt: more about this file are described below. Please use the text file, instead of pdf format this time. (30%)

For WordcountDesignForC.txt, you will have to include the following things:

1. **Design Description:** This part describes how you implement your program, such as the steps to resolve this problem. You **should** describe how to use the linklist in your program. How to find out files inside a directory or a file?

2. **Performance Comparison:** This part will get the performance of your script on a middle size input, which is provided at `http://www.cs.utsa.edu/~tongpingliu/teaching/cs3423/middleinput`. Since you already got the performance of SHELL script and perl script on the same input. To get the performance data, you simply run

```
time ./wordcount middleinput 0 OUTPUT
```

3. **Advantage of Using C language:** You simply lists some advantage of using C language, comparing to SHELL and Perl script, based on your observations. For example, performance, easy to program, maintainability, readablity, or other reasons. The answer can be open-minded.

4. Changing the Assignment3: If you are using your own link list, how can you change it to make it suitable for this homework? Is there a way to make it an universal doubly link list? If yes, how you will change your code to do that or how you did it?

I expecting that the writeup is about two pages.