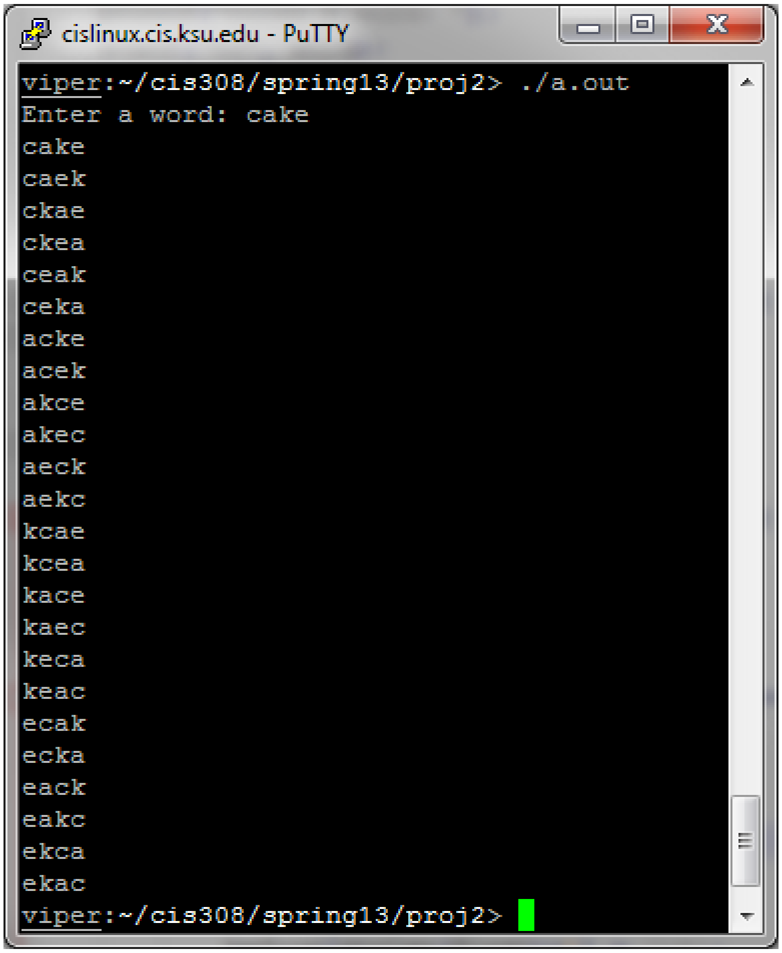
Programming Project 2 (40 points)
Due: Thursday, February 14 by midnight

Assignment Description:

You are to write a program in C that asks the user to enter an input word and then prints all possible permutations of the characters in that word. Here is a sample run of the program:



```
viper:~/cis308/spring13/proj2> ./a.out
Enter a word: cake
cake
caek
ckae
ckea
ceak
ceka
acke
acek
akce
akec
aeck
aekc
kcae
kcea
kace
kaec
keca
keac
ecak
ecka
eack
eakc
ekca
ekac
viper:~/cis308/spring13/proj2>
```

Implementation Requirements:

Your assignment must meet the following requirements:

- Your program should contain at least two separate functions: one for finding permutations of a word and the main function.
- The permutations function must be **recursive**. We will discuss an algorithm you can use for finding the permutations in class.
- Your prompt and output format should exactly match the example above.
- You should not declare any global variables or any static variables

- If the input word has multiple occurrences of a letter, then your permutations algorithm will likely print the same permutations multiple times. (For example, if your input word is “ball”, then you will see “labl” printed twice. This is because each letter ‘l’ is taking its turn as the first character in a permutation.) If this is happening to you, don’t worry about it. You are welcome to optimize your program to not repeat duplicate permutations, but you don’t have to.

Documentation:
Your program must include a comment block at the top of every file, as well as at the top of each function. The function comments should include a brief description of what the function does, and explain any function arguments and return values. You may use the comment block below as a template:

```
/********************************************
* Name: (YOUR NAME)                          *
* Date: (THE DUE DATE)                       *
* Assignment: Project 2: Permutations        *
*********************************************
* (WRITE A DESCRIPTION OF THE PROGRAM)       *
*********************************************/
```

Submission:
First, you will need to create a zip file of your project. To create a zip file in Unix, put all your code for this project (probably just one .c file) in a directory called “proj2”. Change directories to one back from the proj2 directory. To create a zip file of your project called “proj2.zip”:

```
        zip proj2.zip proj2/*
```

It should list all the files that it included in the zip file.

To create a zip file in Windows, again put all your code for the project in a directory called “proj2”. Then, right-click on the proj2 folder and right-click, select “Send To”, and then select “Compressed (zipped) file”. This will create a zip file with your code called proj2.zip.

To submit your project, find the `proj2.zip` file that was created above. Then, go to “`Files and Content->Modules->File Dropbox`” on K-State Online, and upload the `proj2.zip` file. **Put your name and Project 2 in the description box.**

Extra Credit (10 points):
The basic version of the project prints all permutations of the input letters, whether they form an English word or not. To fix this, download the dictionary file “words.txt” from K-State Online. This file contains a number of words, one per line. For 10 points of

extra credit, you should ONLY print those permutations which appear in the dictionary file. These permutations that are also words are called *anagrams*.

Grading:

Programs that do not compile will receive a grade of 0. A grading breakdown for programs that do compile appears below:

User input	2
Correct permutations and formatting in output	28
Documentation and submission	2
Program design (contains recursive function for finding permutations plus a main function, code is as short and simplified as possible)	8
Extra credit (only prints permutations from dictionary file)	10
Total	**40**