Introduction to Computers and Programming
Homework 4 (Week 5)
Due date: 2020/10/29(Thu.)

1. The Fibonacci sequence is a sequence where the next term is the sum of the previous two terms. The first two terms of the Fibonacci sequence are 0 followed by 1.

   ```
   The Fibonacci sequence: 0, 1, 1, 2, 3, 5, 8, 13, 21…
   ```

   Write a **recursive program** that the user can input an integer as the number of terms and show the Fibonacci Series.

   Input：
   ```
   Enter a number：6
   ```

   Output：
   ```
   Fibonacci Series: 0, 1, 1, 2, 3, 5
   ```

2. Modify Programming Project 16 from Chapter 8 so that it includes the following functions:
   ```
   void read_word(int counts[26]);
   bool equal_array(int counts1[26], int counts2[26]);
   ```

   `main` will call `read_word` twice, once for each of the two words entered by the user. As it reads a word, `read_word` will use the letters in the word to update the `counts` array, as described in the original project. (main will declare two arrays, one for each word. These arrays are used to track how many times each letter occurs in the words.) `main` will then call `equal_array`, passing it the two arrays. `equal_array` will return `true` if the elements in the two arrays are identical (indicating that the words are anagrams) and `false` otherwise.

   (Reviewing 8-16)
   Write a program that tests whether two words are anagrams (permutations of the same letters):
   ```
   Enter first word: smartest
   Enter second word: mattress
   The words are anagrams.

   Enter first word: dumbest
   Enter second word: stumble
   The words are not anagrams.
   ```

   Write a loop that reads the first word, character by character, using an array of 26 integers

to keep track of how many times each letter has been seen. (For example, after the word *smartest* has been read, the array should contain the values 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1 2 2 0 0 0 0 0 0, reflecting the fact that *smartest* contains one *a*, one *e*, one *m*, one *r*, two *s*'s and two *t*'s.) Use another loop to read the second word, except this time decrementing the corresponding array element as each letter is read. Both loops should ignore any characters that aren't letters, and both should treat upper-case letters in the same way as lower-case letters. After the second word has been read, use a third loop to check whether all the elements in the array are zero. If so, the words are anagrams.
Hint: You may wish to use functions from `<ctype.h>`, such as `isalpha` and `tolower`.

```
#include <ctype.h>
#include <stdbool.h>   /* C99 only */
#include <stdio.h>

#define NUM_LETTERS 26

void read_word(int counts[NUM_LETTERS]);
bool equal_array(int counts1[NUM_LETTERS], int counts2[NUM_LETTERS]);

int main(void){
      int counts1[NUM_LETTERS] = {0}, counts2[NUM_LETTERS] = {0};

      printf("Enter first word: ");
      read_word(counts1);
      printf("Enter second word: ");
      read_word(counts2);

      if (equal_array(counts1, counts2))
            printf("The words are anagrams.\n");
      else
            printf("The words are not anagrams.\n");

      return 0;
}

void read_word(int counts[NUM_LETTERS]){

      /*remember to check if it's an English character
      *
      *
      */
}

bool equal_array(int counts1[NUM_LETTERS], int counts2[NUM_LETTERS]){

      /* detect if those two words are anagrams.
      *
      *
      */
}
```
