Exercise 1: Name Compatibility (50 marks)

Michelle and Gary are suffering from cold feet just before their wedding. They wanted to confirm if they are really suitable for each other. Michelle remembered a game she played when she was young. The game is as follows: strike off common letters found in both names (each letter can only be struck off once). If both names have an even number of letters remaining or both names have an odd number of letters remaining, then they are said to be compatible. (Zero is considered to be an even number in this exercise.)

Eager to try this out, Michelle and Gary put their full names down on a piece of paper:

```
Michelle  April    Tan
Gary Anand Tham
```

Ignoring case and spaces, they proceed to strike off the common letters:

~~M~~ic~~h~~elle ~~A~~p~~r~~il   ~~Tan~~
G~~ar~~y ~~An~~and ~~Th~~a~~m~~

The remaining letters after the removal are as follows:

```
icellepil
Gyanda
```

The numbers of letters remaining in the two names are 9 and 6 respectively. In other words, Michelle and Gary are not compatible. Feeling discouraged, they wonder if they should really trust this game.

Write a program **compatible.c** to read in two names and check whether they are compatible. You may assume that each name consists of at most 35 characters. A name may contain only spaces or letters.

Your program should contain at least these functions: **readName()** to read one name, and **match()** to perform the above task. You may write additional functions.

Two sample runs are shown below, with user input shown in bold.

Sample run #1:

```
Enter 2 names below:
Michelle  April    Tan
Gary Anand Tham
Number of letters remaining in 1st name = 9
Number of letters remaining in 2nd name = 6
The names are not compatible.
```

Sample run #2:

```
Enter 2 names below:
Tarzan
Jane
Number of letters remaining in 1st name = 4
Number of letters remaining in 2nd name = 2
The names are compatible.
```

Skeleton Program:

A skeleton program **compatible.c** is available in your plab account and is shown below. Do not modify or remove the statements given in the skeleton program.

```
// CS1010 AY2013/4 Semester 1
// PE2 Ex1: compatible.c
#include <stdio.h>

// Name has at most 35 characters
#define MAXLEN 36

// Fill in your function prototypes below


int main(void) {
   char name1[MAXLEN], name2[MAXLEN];
   int remain1,   // number of remaining characters in name1
       remain2;   // and name2 after matching

   printf("Enter 2 names below:\n"); // do not remove/change this line
   readName(name1);      // do not remove/change this line
   readName(name2);      // do not remove/change this line

   // Call the match() function here


   printf("Number of letters remaining in 1st name = %d\n",
          remain1);
   printf("Number of letters remaining in 2nd name = %d\n",
          remain2);

   printf("The names are compatible.\n");
   printf("The names are not compatible.\n");

   return 0;
}

// You must write the function's description here. [1 mark]
// readName() function


// You must write the function's description here. [1 mark]
// match() function

```