CPE 111 - Programming with Data Structures
International Sections - January 2021
Laboratory Exercise 1

Objective
This lab is intended to give you practice using pointers in creating and manipulating data structures. You will create a program similar to but much simpler than **family.c** demonstrated in class.

Instructions

Create a program called **couples.c**. This program will read a file with the names and genders of people, and create a dynamic array to hold structures representing these people. Then it ask the user for pairs of people who are in a couple, and will store that information using pointers. Finally, the program will print out the names of all people in couples.

To run this program, you will type

./couples inputfile.txt

Thus this program can be run with many different input files. The input files have information like the following sample **people.txt** (which you can download from the website):

13
John M
Max M
Steven M
Bart M
Zeke M
Michael M
Louise F
Cynthia F
Jennifer F
Martha F
Iris F
Penelope F
Heather F

The first line holds the number of people in the file. Then each line that follows has the person's first name and gender.

You should define a structure that can hold a person's name, a person's gender and a pointer to the person's partner, for example:

```
typedef struct _person
  {
  char name[32];               /* person's name */
  char gender[2];              /* person's gender */
  struct _person * pPartner;   /* pointer to another PERSON_T
                                  who is this person's partner.*/
  } PERSON_T;
```

After you read the count in the first line, allocate an array of **PERSON_T*** records that can hold that many people. For example:

```
PERSON_T** people = NULL;
/* open the file, read the count */
    ...
people = (PERSON_T**) calloc(count,sizeof(PERSON_T*));
if (people == NULL)
  {
  printf("Error allocating array of %d people\n", count);
  exit(1);
  }
```

Then, in a loop, read the file line by line using **fgets**, and get the name and gender from the line you read, using **sscanf**. Allocate a new **PERSON_T** structure, store the name and the gender, then save the pointer to the new **PERSON_T** as the next element of the **people** array. When you are done, close the file. Then loop through the array, printing the names and genders of all the people you've read.

Next, go into a loop where you ask the user for the names of couples. Each time you get two names, find them in the array, and set their **pPartner** fields to point to each other. Finally, when the user enters "DONE", print out all the couples you have stored.

A sample run is shown below. (Bold indicates things typed by the user.)

```
Welcome to couples program.
Read 13 people:
  John M
  Max M
  Steven M
  Bart M
  Zeke M
  Michael M
  Louise F
  Cynthia F
  Jennifer F
  Martha F
  Iris F
  Penelope F
  Heather F

Enter couple: Max Iris
Enter couple: Penelope Steven
Enter couple: Joe Martha
     Error: person Joe does not exist
Enter couple: John
     Error: you must enter two names
Enter couple:
     Error: you must enter two names
Enter couple: John Martha
Enter couple: Iris Bart
     Iris is breaking up with Max and is now a couple with Bart
Enter couple: Michael Max
Enter couple: DONE

Couples:
     John is coupled with Martha
     Max is coupled with Michael
     Steven is coupled with Penelope
     Bart is coupled with Iris
     Michael is coupled with Max
     Martha is coupled with John
```

```
     Iris is coupled with Bart
     Penelope is coupled with Steven
Bye!
```

Don't forget to free the memory used by your dynamic data structures before you exit the program.

Test your program with **people.txt** and the input above.

Upload the C file **couples.c**.

Extra challenges

1. Modify your program so that it only allows M/F couples. So if the user entered "Michael Max" as above, or "Louise Jennifer", the program will print an error message and not create the couple links.

2. Modify your program so that it does not print each couple twice (as above). So instead of the output above, your program would print:

```
Couples:
     John is coupled with Martha
     Max is coupled with Michael
     Steven is coupled with Penelope
     Bart is coupled with Iris
Bye!
```