## Problem 1:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

//Structure that contains variables for each part of the IP address
typedef struct
address_t{

	int one;
	int two;
	int three;
	int four;
	char name[15];

}address;

//Checks to see if two addresses are on the same network
int
localnet(address addr1, address addr2){

	int i;
	if (addr1.one == addr2.one && addr1.two == addr2.two && strcmp(addr1.name, addr2.name) != 0)
		i = 1;
	else
		i = 0;

	return (i);
}

int
main(void)
{
	int i, j, l, k, t;
	int  array_size;
	char same_net[15][15];

	//Reads all addresses and names from the File
	FILE *in;

	in = fopen("L8_ip.txt", "r");

	address ad[300];

	i = 0;
	while(!feof(in)){

		fscanf(in, "%d%*c", &ad[i].one); // <- note usage of "*c" to skip the "." in the file
		fscanf(in, "%d%*c", &ad[i].two);
		fscanf(in, "%d%*c", &ad[i].three);
		fscanf(in, "%d%*c", &ad[i].four);
		fscanf(in, "%s", ad[i].name);

		//Defines the array size as one less than the location of the sentinel
		if(ad[i].one == 0 && ad[i].two == 0 && ad[i].three == 0 && ad[i].four == 0 && strcmp(ad[i].name, "sentinel")
== 0)
			array_size = i;
```

```
		i++;
	}
	fclose(in);

	int local[array_size];

	//Passes each address one at a time into function "localnet"
	//Records the addresses where the ip's are the same
	for(i = 0; i < array_size; i++){
		for (j = 0; j < array_size; j++){
			t = localnet(ad[i], ad[j]);
			if (t == 1){
				local[l] = j;
				l++;
			}
		}
	}

	//Copies the names of the computers depending on which address they are located at
	for (k = 0; k < (array_size/2); k++) // <- Note: loop stops at array_size/2 since each result is repeated twice
(always an even number of pairings)
		strcpy(same_net[k], ad[local[k]].name);

	//Prints which names are the on the same net
	for (i = 0; i < (array_size/2); i++)
		printf("Servers %s and %s are on the same local network\n", ad[i].name, same_net[i]);

	//Prints all ip's and names
	for (i = 0; i < array_size; i++)
		printf("\n%d.%d.%d.%d %s", ad[i].one, ad[i].two, ad[i].three, ad[i].four, ad[i].name);

	return 0;
}
```

```
Servers dumbledore and hermione are on the same local network
Servers spiderman and wonderwoman are on the same local network
Servers gandalf and mirkwood are on the same local network
Servers zeus and aprhodite are on the same local network

121.211.171.34 dumbledore
21.67.43.78 spiderman
56.89.12.122 gandalf
67.109.113.215 zeus
21.67.43.79 wonderwoman
67.109.124.3 aprhodite
121.211.104.23 hermione
56.89.177.171 mirkwood
100.200.100.100 einstein
```

**Problem 2:**
**C program:**

```
#include <stdio.h>
#include "mylibrary.h"

int main(void)
{
	double ls[10][10], calc[5], results[5];
	int i = 0, k;

	//Read values from file
	FILE *in;
	in = fopen("L8_real.txt", "r");

	while (!feof(in)){
		for (k = 0; k < 10; k++)
			fscanf(in, "%lf", &ls[i][k]);

		i++;
	}
	fclose(in);

	//Pass array to each function to get calculations
	calc[0] = sumdiag(ls);
	calc[1]= sumall(ls);
	calc[2] = avright(ls);
	calc[3] = corners(ls);
	calc[4] = largeanti(ls);


	//Write calculations to Binary File
	FILE *out;
	out = fopen("results.bin", "wb");

	for	(i = 0; i < 10; i++)
		fwrite(&calc[i], sizeof(double), 1, out);
	fclose(out);

	//Read results from Binary File
	FILE *read;
	read = fopen("results.bin", "rb");

	i = 0;
	while (!feof(read)){
		fread(&results[i], sizeof(double), 1, read);
		i++;
	}

	//Print Results
	printf("The sum of the diagonal numbers is %.1lf.\n", results[0]);
	printf("The sum of the all numbers is %.1lf.\n", results[1]);
	printf("The average of the rightmost numbers is %.1lf.\n", results[2]);
	printf("The sum of the four corners is %.1lf.\n", results[3]);
	printf("The largest number on the antidiagonal is %.1lf.\n", results[4]);

	return 0;
}
```

## C library:

```
//Sums all values along the main diagonal
double
sumdiag(double array[10][10]){

	int i;
	double sum = 0;

	for (i = 0; i < 10; i++)
		sum = sum + array[i][i];

	return(sum);
}

//Sums all of the values
double
sumall(double array[10][10]){

	int i, k;
	double sum = 0;

	for (i = 0; i < 10; i++){
		for (k = 0; k < 10; k++)
			sum = sum + array[i][k];
	}

	return(sum);
}

//Finds the average of all of the rightmost values
double
avright(double array[10][10]){

	int i, count = 0;
	double avr = 0;

	for (i = 0; i < 10; i++){
		avr = avr + array[i][9];
		count = count + 1;
	}

	avr = avr/count;

	return(avr);
}

//Sums all of the values found in the four corners of the array
double
corners(double array[10][10]){

	double sum;

	sum = array[0][0] + array [9][9] + array[0][9] + array [9][0];

	return(sum);
}
```

```
//Finds the largest number along the antidiagonal
double
largeanti(double array[10][10]){

	int i, k = 9;
	double numbers[10], largest;

	for (i = 0; i < 10; i++){
		numbers[i] = array[i][k];
		k--;
	}

	largest = numbers[0];
	for (i = 0; i < 10; i++){
		if (numbers[i] > largest)
			largest = numbers[i];
	}

	return(largest);
}
```

```
The sum of the diagonal numbers is 7038.7.
The sum of the all numbers is 54410.4.
The average of the rightmost numbers is 511.2.
The sum of the four corners is 3478.9.
The largest number on the antidiagonal is 980.8.
```