CS 100 Final Exam – Coding – Fall 2018

Create a directory called **final** using **mkdir final** and move into that directory to complete the programs below.
Templates for programs **two.c** and **three.c** can be downloaded from Blackboard.
Do NOT use the Internet while coding your programs. You can use the **cs-intro** server to test your programs if desired.
When finished, compress your **final** directory into a single (compressed) file and submit it to Blackboard.

1. Name this program **one.c** – This program will read integers from a file specified on the command line. Each integer will then be either discarded or written into one of the three files. If the integer is divisible by three (3), it will be written into a file named `three.dat`, and if it is divisible by five (5), it will be written into a file named `five.dat`. However, if the integer is divisible by both three (3) and five (5), it must be written into a different file named `both.dat` only. All other integers will be discarded. At the end, the program will also print out three lines indicating how many numbers out of all the numbers are divisible by three only, by five only and by both. The following illustrate the contents of three output files and the screen printout after executing **./a.out test.dat** with `test.dat` also shown below.

```
Input file: test.dat
33 15 42 16 57 25 38 20 41 10 60 27 12 35
```

```
Output file: three.dat
33 42 57 27 12
```

```
Output file: five.dat
25 20 10 35
```

```
Output file: both.dat
15 60
```

```
Screen printout:
5 out of 14 integers are divisible by three only.
4 out of 14 integers are divisible by five only.
2 out of 14 integers are divisible by both three and five.
```

You can assume the user will prepare the correct input file and enter the correct command line. You can also assume all the files can be opened for either reading or writing. You can use any white space to separate the integers in the output files.

2. Name this program **two.c** – This program reads a matrix of integers from a file specified on the command line. It then finds the maximum of all the integers in the matrix and counts its occurrences. Your job is to complete the `findMax` function that finds the maximum and counts its occurrences. **Please do not change anything else.** The `findMax` function has the following signature.

```
void findMax(int **x, int numRows, int numCols, int *pMax, int *pCount);
```

As an example, consider the file named **data** shown at the right. The execution of **./a.out data** will produce the following output.

```
The maximum is 59, and it occurs 4 times.
```

data

```
4 5
13  19  21  17  27
44  59  34  22  59
15  25  18  22  26
46  59  59  24  34
```

3. Name this program **three.c** – This program will first read integers from the user and use add-at-rear to build a linked list of integers, and then it will build a new list (a sublist) that contains only the integers that are greater than the average of all the integers in the original list. Your job is to complete the following two functions. **Please don't change anything else.**
   - **void printList(Node *head);** Print the linked list. In the printout, two numbers should be connected by an arrow (-->). For example, 7-->9-->5-->8.
   - **Node *sublist(Node *head);** Build and return a sublist that contains only the integers that are ***greater than*** the average of all the integers in the input list. If the input list is empty, the sublist should be empty too.

The following is an example execution of the program. Please note the average of 8 integers in the list is 4.875.

```
Enter a list of positive integers and use -1 to end:
3 7 9 5 1 2 4 8 -1
The original list: 3-->7-->9-->5-->1-->2-->4-->8
The resultant sublist: 7-->9-->5-->8
```