COP 3223 Program #9: More Pizza and Weather

Problem A: Online Ordering Version 2 for Pizza Shack (pizza2.c)

Your previous online ordering program for Pizza Shack was a great hit! Of course, it only worked for three specific pizzas but Pizza Shack offers many different pizzas and the choice of pizzas and prices changes over time, so your boss would like for your program to take file input of the current pizza prices so that you don't have to rewrite your program every time there is a change in price or pizzas offered.

Also, it turns out that the time calculations provided by your previous program didn't help customer loyalty, so your boss will ask you to no longer include this feature.

To simplify this problem (as many restaurants do), each different pizza order will be given a number (starting from 0) and each of these orders will have a price. Your program will read in this information from a file named "pizza2.txt" and then use it to determine the prices of different orders of pizza and then add 6.5% sales tax to the final total. The information for these orders will ALSO be in the file. The specifications for the file format are given below.

Input Specification (pizza2.txt)

1. The first line of the input file will contain a single positive integer, *n (n < 100),* specifying the number of different pizza orders for Pizza Shack.
2. The next *n* lines will each contain one positive real number specifying the price for that corresponding pizza. (The first line will have the price for pizza 0, the next line for pizza 1, etc.)
3. The following line will contain a single positive integer *k* (*k* < 50) representing the number of orders you have to evaluate.
4. The next *k* lines will contain information about each of the *k* orders with one order per line.
5. Each of these lines will contain *n* non-negative integers, representing how many of those pizzas, respectively are in the order. (For example, if n=5 and the line contains the values 0 0 6 0 9, then the order contains 6 slices of pizza #2 and 9 slices of pizza #4.)

Output Specification

For each of the *k* test cases, output a line with the following format:

```
Order #X: cost = $YY.YY.
```

where `X` is the test case number (starting with 1), and YY.YY is the price of the pizza displayed with exactly 2 digits after the decimal. (Note: The price may exceed 100 dollars, so YY simply represents a dollar amount and not the exact number of digits in that dollar amount.

Sample Input File (pizza.txt)

```
5
3.00
3.50
4.50
5.00
6.00
3
1 1 1 1 1
0 0 2 1 6
1 0 3 2 3
```

Sample Output (Corresponding to Sample Input File)

```
Order #1: cost = $23.43.
Order #2: cost = $53.25.
Order #3: cost = $47.39.
```

Problem B: More Weather Predictions and Pizza (weather2.c)

Now that several Pizza Shacks have opened, the owner of Pizza Shack has made the process so efficient that it's no longer necessary for any restaurant to stay open year round. Rather, as long as there is a four month stretch (or longer) within a calendar year that is temperate, a restaurant can just stay open for that stretch of time.

Your boss has asked you to edit your last weather program so that it calculates the percentage of temperate days (in between a specified range of temperatures designated by the user) for each month in a locale. If four consecutive months (or more) have a percentage of temperate days higher than the user entered threshold, then your program should recommend opening a Pizza Shack.

The input file format for the weather file will be the exact same as your previous program.

Input Specification

1. The lower bound temperature in Fahrenheit will be an integer in between 0 and 80.
2. The upper bound temperature in Fahrenheit will be an integer greater than the lower bound and less than 90.
3. The desired percentage will be an integer in between 0 and 100, inclusive.
4. The name of the weather file will be a string less than 20 characters in length and the corresponding file will be in the format of files posted at the website http://academic.udayton.edu/kissock/http/Weather/default.htm, **with a -1 added to the very last line, to make detecting the end of data easier.**

Output Specification
For each month, output a single line with following format:

```
Month m: X percent of days in range.
```

where m is the month number, ranging from 1 to 12, inclusive and X is the percentage of days with valid readings where the temperature was within the range entered by the user, rounded to one decimal place

If there are four or more consecutive months with a percentage of days above the user-entered threshold, output the following line as the last line of output:

```
Great, I recommend opening a Pizza Shack at your location!
```

Otherwise, output the following last line:

```
Sorry, your weather isn’t temperate enough for Pizza Shack.
```

Note: In order for four months to be in a row in a calendar year, the starting month MUST be September or earlier. (Thus, for the purposes of this program, do NOT count October, November, December and January as four consecutive months, or any other streak that “crosses” the boundary of a year.)

Sample Program Run (User Input in Bold and Italics)

```
Please enter the lower temperature bound in Fahrenheit.
60
Please enter the upper temperature bound in Fahrenheit.
80
Please enter the percentage of days needed in the range.
60
Please enter the name of the weather data file for your
city.
FLORLAND.txt

Month 1: 54.4 percent of days in range.
Month 2: 65.7 percent of days in range.
Month 3: 83.7 percent of days in range.
Month 4: 97.2 percent of days in range.
Month 5: 84.2 percent of days in range.
Month 6: 57.3 percent of days in range.
Month 7: 38.1 percent of days in range.
Month 8: 35.1 percent of days in range.
Month 9: 57.0 percent of days in range.
Month 10: 91.7 percent of days in range.
Month 11: 86.5 percent of days in range.
Month 12: 64.0 percent of days in range.
Great, I recommend opening a Pizza Shack at your location!
```

Sample Program Run (User Input in Bold and Italics)

```
Please enter the lower temperature bound in Fahrenheit.
50
Please enter the upper temperature bound in Fahrenheit.
70
Please enter the percentage of days needed in the range.
35
Please enter the name of the weather data file for your
city.
MABOSTON.txt

Month 1: 2.3 percent of days in range.
Month 2: 0.7 percent of days in range.
Month 3: 7.9 percent of days in range.
Month 4: 35.6 percent of days in range.
Month 5: 80.0 percent of days in range.
Month 6: 63.1 percent of days in range.
Month 7: 28.1 percent of days in range.
Month 8: 37.1 percent of days in range.
Month 9: 76.8 percent of days in range.
Month 10: 74.2 percent of days in range.
Month 11: 24.9 percent of days in range.
Month 12: 5.6 percent of days in range.
Sorry, your weather isn't temperate enough for Pizza Shack.
```

Note: These two files are posted in WebCourses under program 8. If you download them from the website, you may get a slightly different version with more days added. Also, both of these files have the -1 added as the last token to indicate the end of file. Make sure to add this to any new file you download from the Weather Website.

Deliverables

Please submit two separate **.c files** for your solutions to these problems via WebCourses by the designated due date:

Program A: **pizza2.c**
Program B: **weather2.c**