

## Homework Assignment #5 — Sales analysis

After analyzing your coffee shop sales, you've realized it would be useful to analyze a variable number of months. That way you can analyze cold months separate from the warm ones, since ice coffee isn't very popular in February. You want to adapt your existing analysis code to take command line arguments. You've also learned about the heap and want to test that out in your project.

### Outcomes

After successfully completing this assignment, you should be able to:

- Use command line arguments in C
- Allocate memory on the heap

### The Assignment

1. Start with your program for Homework 4, or the solutions for Homework 4 that will be posted on 4/19. Modify main, and any required functions, to take a variable number of command line argument, with the format as follows:

```
./analysis.exe Jan.txt Feb.txt
```

Where the command line arguments are the name of the input data files.

2. Next, write a two new functions:
  - A. One called `readCosts` that takes reads the Costs column of the the Costs.txt file into an array. The function prototype should be:

```
float* readCosts(int numItems);
```

Where `numItems` is the number of items to be read and the function should use `malloc` or `calloc` to allocate the memory. (Make sure to use `free()` to deallocate the memory at the end of main).

- B. One called `readIncome` that takes reads the Income column of the the Costs.txt file into an array. The function prototype should be:

```
float* readIncome(int numItems);
```

Where `numItems` is the number of items to be read and the function should use `malloc` or `calloc` to allocate the memory. (Make sure to use `free()` to deallocate the memory at the end of main).

As a result of these two functions, you will have two 1D arrays instead of the current money array in main. (We are essentially splitting `money[51][2]` into two 1D arrays, one for each column).

3. Finally, modify your functions from Homework 4 as needed to deal with a variable number of months being analyzed. You will need to change the function calls (and prototypes) to the following in order to manage a variable size 2D array. Make sure that month\_data is created with the appropriate number of columns.

```
void get_totals(int rows, int col, int array[][col]);  
void get_profits(int rows, int col, int array[][col], float income[], float  
cost[]);  
void top_earner(int rows, int col, int array[][col], float income[], float  
cost[]);
```

### Using your resources:

- If you consult web resources, or other students or staff when developing your program, *you must cite your source*. If you completed the entire program on your own, you should say in your write up file that this is entirely your work.
- If you borrowed some or all of an algorithm from someone else or from somewhere else, *do not copy it*. Write it out in your own words and your own coding style. Also, please explain enough about how the algorithm works that the graders can conclude that you understand it.

### Deliverables:

1. Write a document called **README.txt** that summarizes your program, how to run it, and cites your sources. If you used any outside resources for this assignment, be sure to cite your sources *and* explain in detail how the code works.
2. The .c and .h files needed for your program (at least 2 .c files)
3. makefile which will be used to compile your program. The executable should be called analysis.exe

Programs submitted after 5:00pm on due date will be late and subject to the **Late Homework Policy**.

### Grading:

- |  |        |
|--|--------|
| - Correct execution with autograder test cases               | 50 pts |
| - Use of readCosts and readIncome to read the Costs.txt file | 10 pts |
| - Use of malloc and free                                     | 10 pts |
| - Use of multiple source files and a makefile                | 10 pts |
| - Satisfactory README file with all required parts           | 10 pts |
| - Header comments for each function                          | 5 pts  |
| - Proper indentation   | 5 pts  |

### Deductions:

1. Compiles with warnings - 5 pts
2. No comments within functions -10 pts