Homework 3
Assigned: Wednesday, 3/29/23
Due: Wednesday, 4/5/23, 5:00 PM

Homework Assignment #3 — A better Cash register

Abstract

You've decided to improve your cash register program to allow for command-line arguments. You also want to be able to determine the number of times a particular item was sold and save that information to a text file.

Outcomes

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

- Develop a complex program in *C*.
- Read and write from text files
- Use command line arguments

Expected length: 100 lines of code

The Command Line

The user should be able to run your program using the following command line options:

- To get the overall revenue or register balance:

  ```
  ./sales.exe task inputFile
  ```

  Where `task` is the task number (1 or 2) and `inputFile` is the file to be analyzed

- To get the number of times an item was sold:

  ```
  ./sales.exe task inputFile outputFile type item
  ```

  Where `task` is the task number, `inputFile` is the file to be analyzed, `outputFile` is the output file, `type` is 'a' for append or 'o' for overwrite and `item` is the item code to be analyzed

The Assignment:

- Modify your program from Homework 1 to take command line arguments as described above.
- Add an additional function called `tally` that reads the input file and tallies the number of times a particular item was sold and returns the value as an int

  ```
  int tally(FILE *f1, int item);
  ```

- Modify `main` to print the results of `tally()` to a file The user should be able to specify if the output file should be appended to or overwritten.

If the user does not give enough command line arguments, the program should print the following error message and exit the program:

```
Not enough command line arguements
```

If the option is invalid, print the following error message and exit the program:

```
This option is not valid.
```

Example runs and outputs are found on Canvas.

Before Starting

Review the following topics:

- Input and output using fprintf and fscanf in C
- Command line arguements

Step 1: Read all instructions

Step 2: Prep work

Begin by writing pseudocode that outlines the algorithm you will develop. In addition to submitting your working program, you **must** submit pseudocode outlining `tally()`.

Step 3: Write your code

Using the pseudocode you developed in step 2, begin writing your algorithm in stages.

1. Write a function stub for `tally()` that returns 0.
2. Modify `main()` to take command line arguments and call tally.
3. Write each function, compiling and testing as you go.

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 research 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 detailing any problems that you had. Make sure you include examples of the command line arguments
   If you used any outside resources for this assignment, be sure to cite your sources *and* explain in detail how the code works.
2. Your pseudocode for `tally`, submitted as a .txt file or a picture of hand-written pseudocode.
3. `Coffee_shop.c` plus 1 additional .c file and a .h file.

4. `makefile` which will be used to compile your program. The executable should be called `sales.exe`

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

Implementation Notes

- You may not use arrays.
- For this assignment, you **<u>must</u>** partition your program into multiple files.

Grading:

-	Correct execution with autograder test cases	50 pts
-	Use of multiple source files and a makefile	10 pts
-	Satisfactory README file with all required parts	10 pts
-	Pseudocode for tally()	10 pts
-	Header comments for each function	10 pts
-	Proper indentation	10 pts

Deductions:

1.	Compiles with warnings	- 5 pts
2.	Use of arrays	- 10 pts and return to student to correct

Example input and output can be found on the next page and on Canvas under the Homework 3 folder.

Example input and output. User input is highlighted.

Option 1:

```
./sales.exe 1 input.txt
```

```
83.14
```

Option 2:

```
./sales.exe 2 input.txt
```

```
208.14
```

Option 3 with overwrite:

```
./sales.exe 3 input.txt counts.txt o 3
```

Resulting contents of counts.txt:

```
3       2
```

Option 3 with append (assume the line above has already been run) :

```
./sales.exe 3 input.txt counts.txt a 5
```

Resulting contents of counts.txt:

```
3       5
5       4
```