Assignment 4 - Modularity, Looping, Selection, File and Interactive Input, Concatenation

Assignment Specifications

Overview

You are the owner of a small but growing IT network consulting firm, consisting of several staff members, some of which are employees and others are contractors. The company is always working on several client projects, each of which might have anywhere from 1 to 15 (or more) team members.

You are developing a weekly Staff Member Earnings Report Generator application. The user will interactively enter information for each staff member, including their first name and last name, hourly wage, and number of hours worked. The program should create a report as this information is entered. The report will be stored in a sequential disk file in text format.

The report generated by the application will include the information entered by the user, plus will show both the regular and overtime hourly wage, regular and overtime pay for the period, and a total pay for the period. The overtime rate is 1.5 times the regular rate, although the legislature is considering changing this within the next year or two. Overtime is currently paid for any time over 40 hours during the period; however, this might change too.

User Interface Specifications

Here is a sample dialog. Your spacing and wording should be the same as it is here. Please note that the yes/no answer in response to "Process another employee?" is a single character, and can be Y, y, N or n. No validation is necessary in this program, so assume that the user types a valid response (one of the ones just mentioned).

```
NetworkHaus Information Technology, LLC
Staff Earnings Report Generator

Please enter the project name: MFS Server

Enter staff member #1's first name: Monique
Enter staff member #1's last name: Fabre
Enter the hourly wage of Monique Fabre: 37
Enter total number of hours: 60

Thank you. Process another employee? Y

Enter staff member #2's first name: Steve
Enter staff member #2's last name: Kim
Enter the hourly wage of Steve Kim: 32
Enter total number of hours: 57.5
```

```
Thank you. Process another employee? y

Enter staff member #3's first name: Juan Carlos
Enter staff member #3's last name: Jimenez
Enter the hourly wage of Juan Carlos Jimenez: 34.50
Enter total number of hours: 40

Thank you. Process another employee? Y

Enter staff member #4's first name: Claire
Enter staff member #4's last name: St. Marie
Enter the hourly wage of Claire St. Marie: 9.75
Enter total number of hours: 21

Thank you. Process another employee? y

Enter staff member #5's first name: Jack
Enter staff member #5's last name: Hadjian
Enter the hourly wage of Jack Hadjian: 28.30
Enter total number of hours: 62

Thank you. Process another employee? n

End of processing.
```

Report File Specifications

Please follow these rules precisely for full credit!

Your output should appear identical to that shown below (same horizontal spacing and everything) given the exact same sample input. If lines are missing or values are off or spacing is different, your score will take a hit.

Assume that the project name is one or more words, 20 characters long maximum, with possible embedded spaces. Each of the two names requested is limited to a logical length of 14 characters, and either name may or may not have spaces. Think about what these variables need to use for the declared size, as well as the string variables that will contain the combined names and any extra space / punctuation. You must use the guideline I've been discussing all term, which is the logical length plus 1 (for declaration). Note that you will have to somehow create both a combined display name and a combined name for the report (in a different format).

You'll have to do a little thinking to figure out how to get the report name lined up so that no matter what length, your report will be properly aligned. The first hint is that it involves joining strings together, something called concatenation. Give it a go, and if you can't figure it out, look at the next document in this folder; it contains additional hints. Part of the purpose of this assignment is to implicitly teach you concatenation. Do NOT use tabs (\t) and make sure your Dev C++ Editor Options specifies to NOT use Tab characters.

In essence, all numbers appearing in the report should be right-justified and decimal-aligned. All numbers appearing in the summary should appear "conversationally" without leading spaces (other than the one which normally separates the number from the previous word). Hourly wage amounts CAN be less than 10.00, so be very careful with your formatting. You may assume that there are no regular wages for a staff member that exceeds 60.00, and that no gross is ever over $9999.99. You may also assume that Total hours per week for an employee never exceeds 80. The sample output can appear correct, but you can still be docked up to a point and a half if things aren't aligned properly (the more misaligned, the higher the deduction).

Remember everything I've taught you about formatting items in a report. Your report should remain aligned without the need for any adjustments as long as the input values are within the value range/length.

Your calculated amounts in the report should NOT be off by one cent. Since we are all using Dev C++, you should end up with the exact same results as below.

Here is the report that would be developed from the sample input above. The output should be written to the file similar to your previous assignments (**c:\class** folder and with the filename having your first initial and last name with a **-er** at the end). For instance, if I were submitting the assignment as a student, my file would be **c:\class\kjefferies-er.txt** on the hard disk. .

Your output file should START with the company name line; there is no blank line before it.

```
NetworkHaus Information Technology, LLC
Weekly Staff Earnings Report

Project: MFS Server

Staff Member                      Reg Hrs        Overtime Hrs
Gross
-------------------------------------------------------------------------
--
Fabre, Monique                 40.0 ($37.00)   20.0 ($55.50)
$2590.00
Kim, Steve                     40.0 ($32.00)   17.5 ($48.00)
$2120.00
Jimenez, Juan Carlos           40.0 ($34.50)    0.0 ($51.75)
$1380.00
St. Marie, Claire              21.0 ($ 9.75)    0.0 ($14.63)    $
204.75
Hadjian, Jack                  40.0 ($28.30)   22.0 ($42.45)
$2065.90
-------------------------------------------------------------------------
--

Total Regular Hours Worked: 181.0
Total Overtime Hours Worked: 59.5
```

```
Percentage of Total Hours That Are Overtime: 24.7%

Total Regular Wages: $5476.75
Total Overtime Wages: $2883.90

Total Wages This Period: $8360.65
```

Compiler bug workaround (CRITICAL and MANDATORY!!!)

All compilers have bugs and quirks. Dev C++ happens to have a problem with any **gets()** that occurs after a **scanf()**. For your program to work, you MUST include the following function definition in your code:

```
int clear()  {
    while ((getchar())^'\n');    // yes, this is a
loop with no body!
}
```

Don't forget to also include a function prototype for this function.

This function should be called immediately following the **scanf()** that reads the answer to the "Process another staff member?
question. Please use a char variable (not a string) for that response, similar to how we did it in Module 5. Such as:

```
printf("\nThank you.  Process another staff member?
");
scanf("%s", &answer);
clear();  //  this is the line you must include
```

If you do not place the **clear()** there, your program won't read the next name properly and you will be VERY frustrated.

What You Are Expected To Submit

For this assignment, I want you to develop (but not submit) a full top-down design including a module structure chart, and a Nassi-Shneiderman chart for each module box. And then produce your C program.

Submit to the submission area the source code file (the file should be named with your first initial and last name with
–er.c appended to it [for example, mine would be called ***kjefferies-er.c***]) and the output file (the **.txt** file) from the sample run. Although you should do the design, you won't be required to hand it in. However, you are welcome to submit it (non-graded basis) either in class or scan it and submit it with your code. I'll comment on it.

Don't worry about "submitting" the interactive portion of the output; I'll just look at your code for that (and possibly compile and execute your code). Be sure to

place your name in the source code file somewhere near the top in a comment. Use the sample input shown above to test your program. Submit the output from that run. Check your answers carefully, and be sure try the program with alternate data also to make sure things always look good!.

Be sure to include appropriate comments and indentation in your source code. Think about the style tips we've discussed in previous lectures and include what you can to make your program easy to read and flexible.

I may test your code to make sure it produces the correct output. For this reason, you should make sure your source code file has the extension of .c instead of .cpp or some other extension. I will score and comment only the last submitted version of this program that is submitted prior to the due time and date..

START ON THIS PROGRAM IMMEDIATELY. It is not one that can be understood and written the same day as the due date! I'm giving you about two weeks to do this, and you will need the time. START EARLY!