Fall 2022 -Programming Assignment 1:

Selection Statements

Objectives:

- Write a program that uses conditional statements and I/O.
- Compile, test, and debug your program.

Tasks:

1. Create a source code file, using Template 1 (main only).
2. Get income data from the user, and calculate taxable income based on filing status.
3. Use the debugger to step through the program.
4. Calculate tax and effective tax rate.

During problem sessions, you may work with other students, but every student will be graded individually, based on his/her own submitted work. Every student must submit the solution.

These instructions will assume that you are using the CLion integrated development environment. If you are using another platform, the steps will be slightly different, but you should be able to do everything described in this assignment. The C source code file should compile and run in any environment.

Background: Income Tax

This program will calculate a simplified version of federal income tax, based on a user's total income and filing status. For this program, filing status is either "single" or "married."

First, we calculate taxable income. The tax code allows a deduction of some income before taxes are applied. The user can either specify an amount for the deduction1 or can take the *standard* deduction.

Standard deduction for 202:
Single: $12,400
Married: $24,800

Subtract the deduction from total income to calculate taxable income. If the user enters a deduction that is smaller than the standard deduction, use the standard deduction.

Next, we calculate the amount of tax. Income is divided into chunks called "tax brackets," and any income that falls with each bracket is taxed at a certain rate. The brackets and rates for 2021 are in the table below.

1 In reality, the user-specified deduction must be *itemized* and documented. For this program, we just have the user enter an amount.

Fall 2022 -Programming Assignment 1:

Rate	Single		Married	
	From	To	From	To
10%	$0	$9,950	$0	$19,900
12%	$9,950	$40,525	$19,901	$81,050
22%	$40,526	$86,375	$81,051	$172,750
24%	$86,376	$164,925	$172,751	$329,850
32%	$164,926	$209,425	$329,851	$418,850
35%	$209,426	$523,600	$418,851	$628,030
37%	$523,601	--	$628,031	--

Each rate is applied only to that part of the income that falls within the bracket. For example, if the user files as single and the taxable income is **$50,000**, the tax is:

- 10% on the first `$9,950 = $995.00`
- 12% on the next amount up to `$40,525 = (0.12)($40,525 - $9,950) = (0.12)($30,575) = $3,669.00`
- 22% on the amount higher than `$40,525 = (0.22)($50,000 - $40,525) = (0.22)($9,475)` = $2,084.50
- TOTAL = **`$6,748.50`**

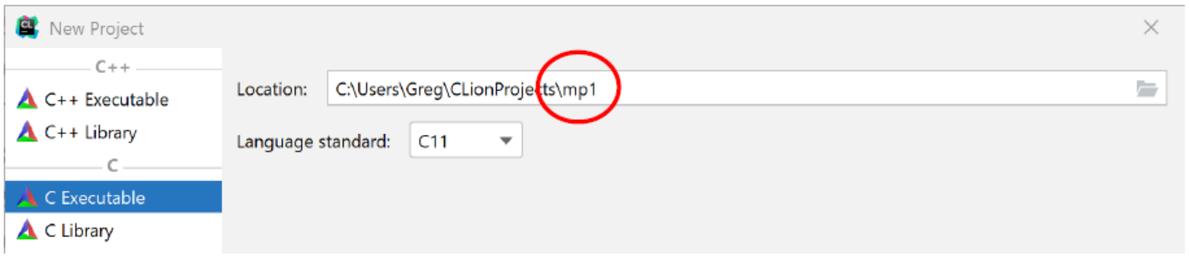
NOTE: To simply your calculations in the program, you can pre-compute the largest amount of tax

Finally, the effective tax rate is the tax divided by total income. This represents the percentage of the user's income that goes to taxes. In the example above, assuming the user took the standard deduction for a single filer, total income is $64,200 and the effective tax rate is **($6,790.00 / $64,200) = 10.57%**. (We are truncating, not rounding, to simplify the program.)

Task 1: Create a C Source Code File

In CLion, create a New Project (under the File menu). The type is "C Executable" and the Language Standard is "C11". Replace the "untitled" in the project name with a name of your choosing, such as ps1. The project name is the directory (folder) where all of your project files will be stored. It will be easier to locate the files later if you give your projects meaningful names.

Fall 2022 -Programming Assignment 1:



Location: C:\Users\Greg\CLionProjects\mp1

Language standard: C11

Note: Pay attention to the full pathname of your project. This tells you where the project files are located, which will be needed later when you submit files.

CLion will automatically create a file named main.c, and will put a “Hello, World!” print statement in the main function. If this is your first time using CLion, build and run the project to make sure your environment is set up properly. (Click on the green triangle at the top right of the window.) You should see “Hello, World!” printed in the output area.

Hint: If you submit this file as is, you’ll get 50 points!!

Note: Add a comment at the top of your file that includes your name. Ideally, this program header would also contain information about what the program does and how to run it, but for problem sessions, just your name will be enough.

Task 2: Get income and filing status from the user

Inside the main function, delete the printf statement.

Write code to prompt the user (printf) to enter total income, and read a decimal integer value (scanf). (Hint: Don't forget to declare a variable to hold the value, and don't forget the ampersand in the scanf call.) For this program, the user should only enter a positive number. Your code does not have to check for this -- just assume the number will be positive. The prompt must look like this, with a single space after the colon:

```
Total income:
```

NOTE: The user will not enter a dollar sign or commas. Just a decimal integer.

If your prompt does not end in a linefeed, then follow the printf statement with the following statement: `fflush(stdout);` This will force the prompt to be printed before waiting for the user to input a number.

Write code to ask the user for filing status. This will be an integer, with 1 meaning “single” and 2 meaning “married”. The prompt must look like this:

Fall 2022 -Programming Assignment 1:

```
Filing status (1 = single, 2 = married):
```

Write code to ask the user for the deduction. The prompt must look like this:

Deduction:

If the specified deduction is less than the standard deduction (for the filing status), print a statement that says:

```
Using standard deduction: xxxxx
```

where xxxxx is replaced with the appropriate deduction (either 14200 or 28400).

Finally, print the taxable income, which is total income minus the deduction. **NOTE: Taxable income cannot be less than zero!**

Taxable income = xxxxx

Compile (build) your program without running it. (The build icon is the hammer in the top right corner of the window.) If you get errors or warnings, try to understand what they are telling you, and make corrections appropriately. If you click on the error, CLion will take you to the line where it found an error. (Keep in mind that the real problem might come earlier in the program, but this is where the compiler figured out something is wrong.)

Once it compiles, run the program. (Click on the green triangle: Run.)

Task 3: Use the debugger

When you write a program, it is unlikely that you'll get it completely right the first time. It is critical that you know how to test and debug your program to make sure that it performs the desired function.

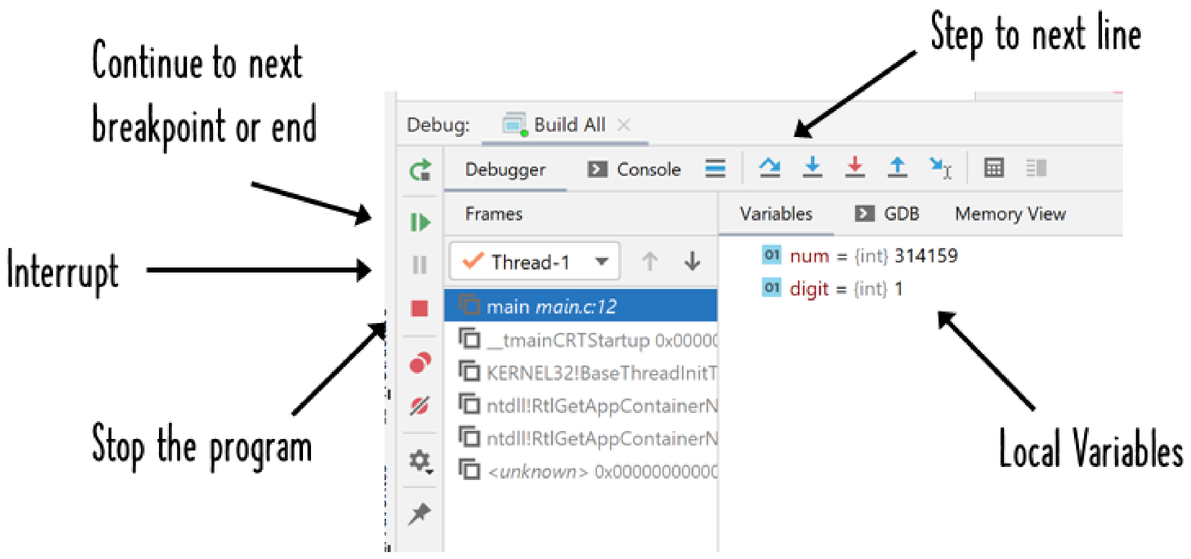
In CLion, as with most integrated development environments (IDEs), the debugger is integrated with the editor. (That's what the "I" in "IDE" means!) This makes it easy to set breakpoints, step through your code, and make changes to the program.

First, set a breakpoint. We'll stop the program after the numbers are read, but before the number is printed. To set a breakpoint, click in the gray lefthand margin right next to your printf statement. You should see a red circle appear, and the statement will be highlighted in pink.

Now, click the green arrow (Run). What happens? Nothing – the program runs normally and completely ignores the breakpoint. That's because you are executing in "run" mode and not in "debug" mode.

Click on the Debug icon – it looks like a green bug. Again the program runs, and waits for you to enter a number. Then, something very different happens. The printf statement is highlighted in blue, to show that the program has stopped executing at this point. Also, the display at the bottom changes.

Fall 2022 -Programming Assignment 1:



The window on the right shows all the local variables and their values. If you were really debugging the program, seeing the values in local variables will be very helpful. You will often find that a variable's value is not what you expected, so your job is then to figure out why.

When you're ready to execute again, you have several options:

- To continue, click on the green triangle (Resume) at the bottom left. This will keep going from where the program stopped.
- To rebuild and start the program over from the beginning in Debug mode, click on the Rerun icon just to the left of the Debugger label. (This is different from clicking the Run icon at the top right, because that will rebuild and run without entering debug mode.)
- Above the debug window, there are a number of different arrows that allow you to step through the program, statement by statement. On the left is Step Over – this will execute the current statement and stop. The next one is Step Into – if the current statement includes a function call, this one will stop at the first statement within that called function. You'll have time to try this out, and the other options, as you write more complex programs.
- If you want to quit running the program, without continuing, just click on the red box (Stop).

You can switch from the debugger view to the console view (where your program reads and prints data) by clicking on the Console and Debugger tabs.

Use the debugger to answer this question: If the user types a non-number (like abc) or a non-integer (like 123.45), what happens to the variable in the scanf statement? Try a few different inputs (e.g., ab33, 123xyz) and confirm that scanf behaves the way we discussed in class.

Task 4: Calculate the tax

Now that we have computed the taxable income, use the table above to calculate the amount of tax owed by the user. Print this amount, along with the effective tax rate (tax divided by total income, as explained above).

The values being printed in this section are floating-point numbers. Since they are dollar amounts, show only two digits after the decimal place. We won't worry about rounding up or down – just truncate the result to two decimal places. The printf format code for this is **`"%.2f"`.**

The results should look like this, with bogus numbers plugged in just to illustrate:

```
Tax = 1234.56 Effective tax rate: 12.34%
```

Note that the effective tax rate is a percentage, while the number you get from the division is not. How to you convert a fraction to a percentage? To print the percent symbol, use "%%" in the printf call.

Compile and execute your program. Try different values to make sure that it behaves correctly in all scenarios. This is the testing phase of program development. What if total income is less than the standard deduction? What if the total income is very large? Test income values in each bracket, so that you can check the math.

Submit your program to ZyLab for grading

Once you've got your program running and you think it's ready, go to **ZyLab 15.1 Programming assignment**. Upload your main.c file by dragging it to the gray box, or click Choose on hard drive to browse for your file.

Where is your main.c file? It is in the directory that you specified as the "location" when you created the new project. For example, all of my CLion projects are in this directory on my Windows laptop:

C:\Users\Dawit\CLionProjects\

So the location of the main.c for this project is here, where XXX is the name of the project:

C:\Users\Dawit\CLionProjects\XXX\main.c

If you don't know where the project directory is, look in the Project window (left of the editor window). CLion shows you the location next to the project name.

If your program fails some tests, feel free to keep working in CLion. You can upload and test as many times as you want. Each upload will overwrite the previous one, and only the highest grade will be used.

Fall 2022 -Programming Assignment 1:

GRADING

Submit: 50 pts

If you submit any code that compiles and runs, you get 50 points. (If you submit after Task 1, you'll get these points with almost zero effort!)

Compute taxable income (Task 2): 30 pts

Compute tax and effective tax rate (Task 4): 20 pts

ON TIME!!! Unless you have made prior arrangements with me, you must make your final submission before the end of due date. Late submissions will not be graded and will receive a grade of zero.

Here is an example of a run of the complete program in CLion. You must match the spaces and characters exactly for the zyBook tests to pass. User input is shown in bold. (When zyBook shows the output of your program, you will not see the input data or the linefeeds after the input data. This is normal.)

Sample Run 1

```
Total income: 62400
 Filing status (1 = single, 2 = married): 1
 Deduction: 0
 Using standard deduction: 12400
Taxable income = 50000
Tax = 6748.50
Effective tax rate = 10.81%
```

Sample Run 2

```
Total income: 100000
 Filing status (1 = single, 2 = married): 2
 Deduction: 30000
 Taxable income = 70000
Tax = 8002.00
Effective tax rate = 8.00%
```