Project 0
75 to 150 lines of code

The Cash Register

After years of hard work, you've decided to quit your job and follow your dream alternative career: running a local coffee shop. Instead of purchasing a cash register, you decide to write your own program to calculate change.

Overview:
Your program will prompt for two values: the total of the purchase and the amount paid. You will then 1) determine if the customer has paid enough and 2) print out exactly what change should be given. You may not use arrays or pointers in this assignment.

Learning Objectives covered:

1. I can write and compile simple C programs that communicate with the terminal and contain functions, loops and conditionals.
2. I can write a program that adheres to a style guide.
3. I can write high quality documentation.

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 your algorithm.

Step 3: Write your code
Using the pseudocode you developed in step 2, begin writing your algorithm in stages.

1. Write function stubs for each function that return 0.
2. Write `main()`. Best programming practice is to write and test outer loops and conditional statements before working on inner statements. This allows you to catch bugs early, debugging as you write, instead of dealing with all the errors after you have many lines of code written.
3. Write each function, compiling and testing as you go.

Program structure:
One source code file called `change_calculator.c` with the following functions

- `main`
- `check_amount`
- `print_formatted`

Before you start programming, use the program description below to write pseudocode for each function in the program. You must upload your pseudocode as part of the assignment.

Your program will be compiled using the following command:
```
gcc change_calculator.c -Wall -o getChange
```
Your program will be run using the following command:
```
./getChange
```

In main, you should do the following:

1. Prompt the user to enter the total and amount paid using the following format (user input is in bold). Your output must exactly match what is shown below

```
Input purchase total:
40.50
Input amount paid:
50.00
```

2. Check that the payment is valid, using the `check_amount` function. If it is not enough, print the following and exit the program:

```
Insufficient payment.
```

3. Call the `print_formatted` function. In the function, determine the number of bills ($5, $1) and coins (quarters, dimes, nickels and pennies) needed. Print with the following format (more examples are on Canvas)

```
Your change is $7.65
  1 – 5D
  2 - 1D
  2 - Q
  1 - D
  1 - N
```

Functions:
`int check_amount(float total, float paid);`
Returns 1 if paid >= total. 0 if not.

`void print_formatted(float change);`
Prints the number of $5, $1, quarters, dimes, nickels and pennies needed to give the specified change

Deliverables:

1. Write a document called **README.txt** that describes the function of your program, how to compile and how to run it. If you used any outside resources for this assignment, be sure to cite your sources *and* explain in detail how the code works. An example is on Canvas.
2. Your pseudocode, submitted as a .txt file or a picture of hand-written pseudocode.
3. `change_calculator.c`

Restricted material: You may not use arrays, pointers, or global variables.

File types: Upload your README and pseudocode as .jpg, .pdf or .txt files. Do not upload .doc or .docx files.

Grading:

The full grading rubric is explained in the CS 2303 Project Rubric document. Five additional rubric items are listed below.

Met	Rubric Item
	Program does not contain restricted items (arrays, pointers, global variables)
	Functions are implemented as described in the assignment document
	Pseudocode contains a section for each function
	Pseudocode outlines the logic of each function
	Function order in the file is prototypes, main, then implementations

The grading breakdown is shown in the following table. To earn a particular grade, you must meet all criteria in the row corresponding to that grade. There are 7 core rubric items and 11 supplemental items (6 in the Rubric document and 5 listed above).

		Rubric Items Met	
	Autograder tests	**Core**	**Supplemental**
Excellent	All Passed	7	10+
Meets Expectations	All Passed	7	8+
Needs Revision	All Passed	N/A	N/A
Not Graded	Failed 1 or More test	N/A	N/A

Notes:

Using your resources:

- If you consult web resources, or other students or staff when developing your program, *you must cite your source*. If you complete the entire program on your own, you should say in your write up file that this is entirely your work.
- If you view an online algorithm, *do not copy it.* Write it out in plain text in your own words, then use those notes to write your own code.

Function header comments:

Each function you write (except for main) should have a block comment above it that allows other people (and future-you) quickly know what the function does and what information it needs. Different style guides have different requirements. The expected format for this class is shown below. For this assignment, the function comments will be simple, for future assignments they will become more complex.

```
/***************************************************
* returnType function_name(type1 param1, type2 param2)
* Brief description of the function
* Parameters:
*    param1: explain what param1 is
*    param2: explain what param2 is
* Returns:
*    explain the return value
* Notes: (optional) Explain any output or other notes
* Sources: (optional) cite any websites used for the function
************************************************/
```