Large Program 2
The Unlimited Vending Machine and Wallet

COP3223C Introduction to Programming with C — Fall 2022

Dr. Andrew Steinberg

Background Story of this Assignment

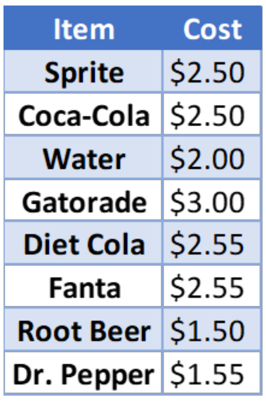
You are about to implement your second large program in this course. In this assignment, the primary objective is to apply your skills in user-defined functions and pointers!! Pointers are an extremely imperative topic to understand in this class and your future as a CS student. In this assignment, you are going to implement the following scenario.

Scenario: You have a magical wallet that contains an unlimited amount of cash and you can pull any amounts of $1, $5, and $10 bills. You find a vending a machine that contains (you guessed it from the title of the assignment) an unlimited amount of drinks. You decide that you are going to purchase a lot of drinks from the unlimited vending machine with your unlimited wallet. In this large programming assignment, you will simulate this scenario that utilizes pointers heavily.

The program will have the following setup.

- The program will first welcome the user with a friendly message. It will provide the user with instructions on how the program works.
- The program will be menu driven. This means the program will display a list of options of what the user would like to do. From the list of options, the user will select one of the following options.
  - User will be able to view the current amount in their hand. **When the program starts for the first time, the user will have $0.00.**
  - User will be able to order a drink from the vending machine. *Note: In order for the user to make a purchase, the program must make sure the amount of money in the user's hand will be able to handle the transaction. If not, the user will have to make sure to pull money from the unlimited wallet until enough money is added to make the purchase.*
  - User will be able to view the menu drinks along with its respective prices.
  - User will be able to pull money from the unlimited wallet.
  - User will be able to exit the program and see the amount of money in their hand before the program terminates.

- The following table shows the vending machine items you will be using along with their respective price. Make sure to use these items and prices or else points will be deducted.



Item	Cost
Sprite	$2.50
Coca-Cola	$2.50
Water	$2.00
Gatorade	$3.00
Diet Cola	$2.55
Fanta	$2.55
Root Beer	$1.50
Dr. Pepper	$1.55

Due Date

The assignment is due on October 30th at 11:59pm EST via Webcourses. **Do not email the professor or TAs your submissions as they will not be accepted!** This assignment is accepted late up to 24 hours with a penalty. Please see the syllabus for more information on this. Make sure to submit on time to get potential full credit. Make sure to also take into consideration the uploading time. In the past, students who are working last minute on the assignment sometimes run into uploading issues where their Internet may run slow, resulting in late submissions. The timestamp Webcourses uses for your submission will be applied and will be the final say. Please do not email the instructor or TAs saying your Internet was running slow. If the time is off by a second of the due date, then the assignment is considered late. Plan accordingly!

Important! Read Carefully!

In this assignment, you are going to implement **6 user defined functions**

```
void greeting(); //welcome the user
void order(double *balance); //user will make a purchase
void viewHand(double *balance); //display current amount in hand
void transaction(double *balance, double price); //transaction with user
void pullMoney(double *balance); //grab more money from the unlimited ↩
    wallet
void displayVendingOptions(); //display beverage options and prices
```

You cannot add any additional functions or remove them. Modifying them in any way (name, parameters, etc...) will result in point deductions! Utilize them the way they are provided!

The function prototypes are provided for you here. You will create a file called `largepro-`

`gram2_lastnamefirstname.c` for this assignment and write out the code. There is NO SKELETON PROVIDED FOR THIS LARGE PROGRAM! **Do NOT modify any of the function prototypes that are provided for you!** Any modifications from the provided function prototypes will result in points being deducted! You are allowed to use the same message that is provided in the sample output for your program. You can also make modifications to the text of messages as long as the program follows all directions as stated in this assignment.
The following section will discuss the prototypes more in detail. Make sure that you name your C file *largeprogram2_lastname_firstname.c*, where lastname and firstname is your last and first name (as registered in webcourses). For example Dr. Steinberg's file would be named *largeprogram2_Steinberg_Andrew.c*. Make sure to include the underscore character _. If your file is not named properly, points will be deducted.

The Function Prototypes

This section will discuss the function prototypes.

```
void greeting();
```

This function welcomes the user to the vending machine by printing a friendly message to the user. **Important: The menu driven component should not be placed in the greeting function. That should be placed in your main function after the greeting function is invoked.**

```
void order(double *balance);
```

The order function will handle the collecting of the respective item the user would like to order from the vending machine. Inside the function definition you will ask the user what they would like to order. Once the user makes a selection, the program should display the selection along with the cost of the item. **Important: Now that you know conditions, make sure to handle if the user inputs an invalid selection.** Once a proper selection was made, the function will begin the transaction. The function has one parameter that holds a reference to the address of the balance variable. Make sure to consider invalid input.

```
void viewHand(double *balance);
```

The viewHand function will display the user's account balance to the terminal. The function's parameter is a reference to the address of the variable balance. Note. If the user selects the functionality first during the program run, the account balance should be $0.00.

```
void transaction(double *balance, double price);
```

The transaction function handles the actual transaction based on the item that was purchased from the vending machine. The function has two parameters. One is a reference to the amount of money in the user's hand and the other is the price of the item selected. **Important: The second parameter is passed by value. It is not a typo.** Also, there can be a situation where the amount of money in the user's hand is not enough for the transaction. If such a scenario occurs you must have the user pull money from the unlimited wallet. This is all done in the function definition. Hint: Think about conditional while loop in this scenario. Once the balance can afford the item, then proceed with the actual transaction.

```
void pullMoney(double *balance);
```

The pullMoney function adds money to the user's hand. Three options are given for how much they can pull from the wallet. The user can add $1, $5, or $10. One of these options is selected and the amount stored in their hand will increase based on the respective amount. The function has one parameter that represents a reference to the balance. Make sure to consider invalid input.

```
void displayVendingOptions();
```

The displayVendingOptions function displays the beverage items the vending machine has available along with its respective price. The table of items and prices is on page 2 of the assignment. **Important: The displayVendingOptions function does not display the menu of options the user can perform in the vending machine. That component should be in the main function.**

Testing on Eustis

It is your responsibility to test your code on Eustis. If you submit your assignment without testing on Eustis, you risk points being deducted for things that may behave differently on your operating system. Remember, you cannot dispute grades if your code didn't work properly on Eustis all because it worked on your machine. The Eustis environment gives the final say. Plan accordingly to test on Eustis!!

Why was I not provided a Python Script File?

You are probably wondering why Dr. Steinberg didn't provide a python script to check your code like in the small programs. The reason is that Dr. Steinberg wants his students to enjoy the assignment without being told of what must be displayed to the terminal and matching exactly. For large programs, Dr. Steinberg allows students to change the text output to the terminal, however your program must perform the specific functionalities that is requested at least to receive potential full credit.

The Rubric

Please see the assignment page for the established rubric on webcourses.

Comment Header

Make sure you place a comment header at the top of your C file. You will use single line comments to write your name, professor, course/section, and assignment. For example, Dr. Steinberg's header would be:

```
//Andrew Steinberg
//Dr. Steinberg
//COP3223C Section 1
//Large Program 2
```

Missing a comment header will result in point deductions!

The Solution Text File

I have provided a sample output of a working program of what is expected in webcourses. As I stated before, you are welcome to change the text output to the terminal, or you can keep it exact like in the sample.

Recommendations for Completing the Assignment

Here is the order of steps I would strongly consider when attempting the large program assignment. After each of these steps, see if your program builds/compiles successfully and performs the correct task. See if the output is what you were expecting.

1. Start with the Message Greeting. Try to get the program to welcome the user.
2. After a successful message greeting, try to design menu driven component. Hint: Think of how to use a while loop and switch statement.
3. After getting the menu working. Work on the display menu function to display all drink prices. Note. This is different from menu that shows user options of what they can do.
4. After getting the display menu function working, work on the viewHand function.
5. After finishing the viewHand function, work on the pullMoney function to increase the amount of money in your hand.
6. After getting the pullMoney function to work, begin the ordering function. This is where the user will pick an item to purchase and make a transaction.
7. Test your program with different inputs to see if it works successfully.

Basic Rules for Completing The Assignment

Please make sure to follow these rules when completing the assignment.

- Please use all items that were presented to you in class and labs. Everything that was presented to you can help you compete the assignment.
- DO NOT COPY ANYONE'S CODE! This is cheating as stated in syllabus. This assignment is individual work.
- DO NOT SHOW YOUR CODE to classmates. The only group of people that can see your code are the TAs, ULAs, and Course Instructor. Follow the rules from the syllabus if you talk to other students. Otherwise you may be in cheating violations!
- Make sure to use good practice of writing code. Anything that is considered bad practice will result point deductions. Example GOTO statements should not be seen anyone's code. There is a reason I don't go cover them. They are extremely bad in practice to use and just awful.

Some Advice

Here are some tips and tricks that will help you with this assignment and make the experience enjoyable.

- Do not try to write out all the code and build it at the end to find syntax errors. For each new line of code written (my rule of thumb is 2-3 lines), build it to see if it compiles successfully. **It will go a long way!**
- After any successful build, run the code to see what happens and what current state you are at with the program writing so you know what to do next! If the program performs what you expected, you can then move onto the next step of the code writing. If you try to write everything at once and build it successfully to find out it doesn't work properly, you will get frustrated trying find out the logical error in your code! **Remember, logical errors are the hardest to fix and identify in a program!**
- Start the assignment early! Do not wait last minute (the day of) to begin the assignment.
- Ask questions! It's ok to ask questions. If there are any clarifications needed, please ask TAs and the Instructor! We are here to help!!! Do not wait last minute to seek help as generally you could be waiting in a long line of fellow classmates who may need help. As stated previously start early and ask your questions right away!