Small Program 1

COP3223C Introduction to Programming with C Fall 2022
Dr. Andrew Steinberg

Due Date

The assignment is due on September 9th 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!

This assignment contains a set of problems that are to be completed in **one C file with one main function** only. We have not learned user-defined functions yet. This will start in Small Program 2 and beyond. This means the main function in your file will execute each problem in a single run. You do not write separate main functions for each problem. **If separate main functions are created, then no credit will be given for the assignment! If you create separate program files for each question, then no credit will be given for the assignment!** The file must be named *smallprogram1_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 *smallprogram1_Steinberg_Andrew.c*. Make sure to include the underscore character _. If your file is not named properly, points will be deducted. The script the graders use will pull your name from the file name properly. It is imperative you follow these steps so you can get your points!

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!!

The Python Script File! Read Carefully!

A python script has been provided for you to test your code with a sample output of Dr. Steinberg's solution. This script will check to make sure your output matches exactly with Dr. Steinberg's solution file as the graders are using this to grade your assignments. The script removes leading and trailing white space, but not white space in the actual text. If there is anything off with the output (including the expected answer), the script will say your output is not correct. This includes your output producing the correct answer, however there is something off with the output display. The script does not point directly where your mistake(s) are in the code. It will only produce a success or unsuccessful output message as a whole. If you get an unsuccessful output my suggestion is to look at the sample solution text file provided to see what is different from your answer and Dr. Steinberg's when comparing. If you have extra white space or new lines or even just missing a space/new line, you will lose points that won't be changed!

Make sure you place the python script in the same directory as your C file. You can use the ls command to check to see if the following items are in the directory.

```
ls
```

You should these three files in your directory.

1. Your C File.
2. The Python Script File
3. Sample Solution Text File

If you have these three files. You are ready to run the script. Use the following command to test your code with Dr. Steinberg's provided solution sample.

```
python3 sp1test.py
```

If the script says your output is incorrect, checkout the sample text file that was generated (a new text file will be created from the script that contains YOUR output). If your numbers are off or different from Dr. Steinberg's, then that means there is something not right with code's logic and calculating the answer. However if your numbers match with Dr. Steinberg's solution, then that means there is extra/missing white space detected. Compare the text file generated by the script with solution text file line by line to find the missing/extra white space or newlines. Once you believe you found the error, rerun the script to see if the output matches.

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
//Small Program 1
```

Missing a comment header will result in point deductions!

The Solution Text File

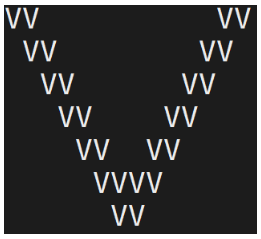
You are provided a solution file that was created by Dr. Steinberg's Python Script. Now you may notice some strange things about the file. In this assignment, you are going to write statements that involve interacting with the user. Now you are probably wondering from the solution text file where the interaction is happening? The fact is that the Python script handles the interaction. The script creates an input stream that feeds it input. That is why you don't see the input directly in the text file. For example, the text file on the last line says `Enter the radius: Enter the height:...` Here this looks like we should be typing input, however the python script has already fed it input. That is why you don't see the values except for the results. In each problem, a screenshot of the C file being executed manually without the Python script shows how it looks on a normal run. Carefully look at the output in the pictures provided.

samplesolutionsp1.txt

```
VV          VV
 VV        VV
  VV      VV
   VV    VV
    VV  VV
     VVVV
      VV
Mileage Reimbursement Calculator
Enter beginning odometer reading=> Enter ending odometer reading=> You traveled 7.0 miles.
At $2.61 per mile, your reimbursement is $18.27
Enter the weight in pounds: Enter the total height in inches: BMI = 3.649
Enter the radius: Enter the height: The volume of the cone is 78.5397
```

Problem 1

Write a set of statements in the main function that displays to the monitor a large letter ‘V’ made up of the character ‘V’. Here is a sample output of how it should look when the program runs. Make sure to follow this output to receive full points. Any differences will cause points to be deducted. You do not need to use loops for this problem. Use the text file output to assist with the number white spaces.

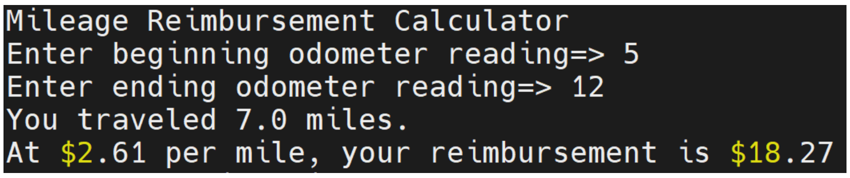


```
VV          VV
 VV        VV
  VV      VV
   VV    VV
    VV  VV
     VVVV
      VV
```

Figure 1: Output for question 1. The output must match exactly to receive credit. This includes white space and new lines.

Problem 2

Write some code that calculates mileage reimbursement for an employee a rate of $2.61 per mile. This code needs to interact with the user. Hint: Use scanf statements. Make sure to follow this output to receive full points. Any differences will cause points to be deducted. Think about the type of data we are working with. Make sure you display up to two decimal places when displaying dollar amount and one decimal place when displaying distance traveled. Make all variables of type double. You do not need to worry about negative values. We have not discussed conditions yet.



```
Mileage Reimbursement Calculator
Enter beginning odometer reading=> 5
Enter ending odometer reading=> 12
You traveled 7.0 miles.
At $2.61 per mile, your reimbursement is $18.27
```

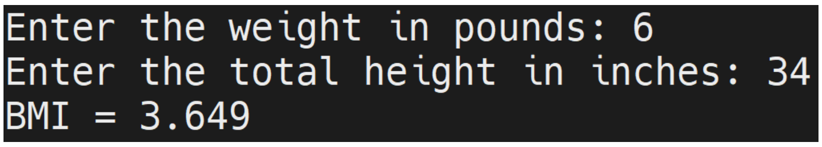
Figure 2: Output for question 2. The output must match exactly to receive credit. This includes white space and new lines.

Problem 3

Write some code that calculates the body mass index (BMI). The BMI can be calculated as follows using the formula

$$BMI = \frac{weightInPounds \times 703}{heightInInches \times heightInInches}$$

You are going to ask the user for some input. You will ask for $weightInPounds$ and $heightInInches$. Both variables are float type. Make sure to follow this output to receive full points. Any differences will cause points to be deducted. Your output result should only display up to three decimal places. Note: Assume the total height is input. Example, 5 feet and 6 inches would result in 66 inches as the input. You do not need to worry about the denominator being 0 or negative values. We have not discussed conditions.



```
Enter the weight in pounds: 6
Enter the total height in inches: 34
BMI = 3.649
```

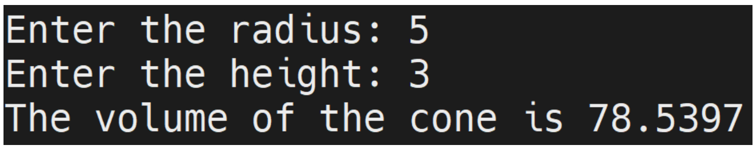
Figure 3: Output for question 3. The output must match exactly to receive credit. This includes white space and new lines.

Problem 4

Write some code inside the main function that calculates the volume of a cone. The formula for calculating the volume of a cone is as follows

$$V = \frac{1}{3} \times \pi \times r^2 \times h$$

The value of $\pi = 3.14159$. Make sure to declare pi as a <u>constant variable</u> or else points will be deducted. The graders will be checking for this and not the script! You should have the following exact output in your code for the script to give you credit. Display the result up to four decimal places. Be very careful with this problem. Some students may run into small errors do to loss of information. Make sure 1/3 is $0.\overline{3}$ repeating and not just 0. All variables are type double. You do not need to worry about negative values. We have not discussed conditions yet.



```
Enter the radius: 5
Enter the height: 3
The volume of the cone is 78.5397
```

Figure 4: Output for question 4. The output must match exactly to receive credit. This includes white space and new lines.