CS 158/159 Lab 01

In this lab session you will complete the following tasks in the specified order. **Do not begin this lab until your lab sessions starts**. Individuals failing to remain on task during the lab will be assigned a zero for the lab, lab quiz, and will be considered absent for the lab. Web browsing, personal e-mail, social networking, and/or text messaging is not permitted during lab.

1. Record Attendance

Log on to your UNIX account on guru on a physical PC in the lab room (not on a wireless device) and enter the command **`attend`** to officially record your lab attendance. A confirmation e-mail will then be sent to you to verify that your attendance was correctly recorded. If you are late, you should still run the `attend` command and continue to participate in the remaining portion of the lab, but you will not be eligible to receive credit for the lab programming assignment. If you are present in the lab, but fail to run the `attend` command as expected, you will be marked as absent and likewise will not be eligible to receive credit for the lab programming assignment.

2. Review Material

The first task is to improve your understanding of the material from the book and the course programming standards (available in the course packet and in Blackboard) by working through a series of problems with the assigned lab partners of your group. You must show your lab instructor that you have successfully completed these tasks BEFORE you leave lab today. You will not receive points for the programming assignment, unless this task has been completed to the satisfaction of your lab instructor.

Answer the following questions related to material found in Chapter 2.

Statement	True or False
A C program begins with a section for preprocessor directives.	
The preprocessor is a part of the compiling process and prepares your code for the remainder of that process.	
Every program must have exactly one function named `main`.	
The `main` function is the starting point for execution of the program.	
Within each function the local declarations and executable statements must NOT be permitted to overlap.	
Variable declarations will NEVER be permitted in the global section this semester.	
The files `stdio.h` and `math.h` are libraries that contain standard functions for our use.	
The `return 0;` statement will be the final statement in the `main` function.	
The `return` statement in `main` will return control back to the first statement in main.	
Comments are added to a program to improve its level of documentation intended for other programmers.	
The course program headers are multi-line comments.	

Answer the following questions related to material found in the course C programming standards.

Statement	True or False
You should place a single space between all operators and operands.	
All variables should be commented to the right of each declaration.	
You should declare multiple variables on one line.	
Select meaningful identifiers (names) for all variables in your program.	
Rarely are single character identifiers considered meaningful for variables in a program.	
Do not single (or double) space the entire program, use blank lines when appropriate.	
There is no need to include example output with your submission.	
All code found between `{` and `}` should be indented two additional spaces.	
The file you submit for lab #1 must be named `lab01.c`	

Modify the program 2-2 on page 46 of your C programming text to generate the following programs. Demonstrate each program for your TA and have them verify that they work properly.

Expected Output	TA Initials

`Welcome. This program adds`
`three numbers. Enter three numbers`
`in the form: nnn nnn nnn <return>`
`4 5 6`
`The total is: 15`
 	
`Thank you. Have a good day.`	

`Welcome. This program adds three numbers.`
`Enter the first number: 4`
`Enter the second number: 5`
`Enter the third number: 6`
`The total is: 15`	

Revise the program to use only two integer variables. One for input and the other for output.	

Answer the following questions regarding Program 2-7 found on pages 71 and 72 of your textbook:

What is the purpose of the statements on lines 12 through 14?
How are the statements on line 17 and 18 related in this program?
Can the statements on lines 20 and 21 be moved before those on line 17 and 18? Why or why not?
Can the statements on lines 20 and 21 be moved after those on lines 23 through 25? Why or why not?
On line 23, what are the `\n` and `%10.2f` instead of the quotes accomplishing for the print statement?

3. Programming Assignment

The second task is to develop a program as a group which solves the given problem. This assignment is worth 15 points and will be **due 30 minutes prior to the start of your next lab session**. All assignment deadlines are firm and the ability to submit your assignment will be disabled after your deadline elapses. No late work will be accepted!

As you develop the program, you should rotate through the following roles approximately every 30 minutes. Do not allow the same person be the driver the entire time. It is not acceptable to designate a single individual to complete the assignment. Every individual group member should have a full understanding of all work submitted. Assignments are an opportunity to develop and demonstrate your understanding of course material.

Role	Description
Driver	The driver is in charge of the computer which includes entering code, saving, testing, and submitting. This individual should be soliciting the other two members for advice.
Navigator	The navigator's role is to look over the shoulder of the driver for syntactical errors, logical errors, and concerns related to course standards. The most common mistakes include failing to pair quotes, misplaced semi-colons, and improper placement of parentheses.
Manager	The manager may not be as close to the driver as the navigator but still plays an important role ensuring that the algorithm to be implemented is correct and can be tested using a variety of input to verify correctness. The manager is responsible for communicating to the teaching assistant who will be making the group's final lab submission.

This programming assignment does not have a single solution, and each group should collaborate together to develop their own unique solution. Submissions may be processed with comparison software and results will be used to detect unacceptable collaboration between groups. The development of your algorithm and the resulting code should only be discussed among your group members and course staff.

Your program must adhere to the course programming standards (available in the course packet and in Blackboard). Please review this document before submitting your assignment, because failure to adhere to it will result in a reduction in points. You program must include the designated program header (~cs15x/student/hdrProg) at the top of the program (which can be inserted in `vi` using the `hp` shortcut while in command mode). The header must include an appropriate description of your program and must contain the official Purdue career account e-mail addresses of each **contributing** group member. Do not include the e-mail address of anyone who did not actively participate in the program development. Failing to participate in the process to the satisfaction of all partners will result in a zero. Also note that course standards prohibit the use of advanced programming concepts not yet introduced in the course, unless otherwise specified.

Each of the example executions provided below represents a single execution of the program. Your program must accept input and produce output **exactly** as demonstrated in the example executions. Your program will be tested with the data seen in the examples below and an unknown number of additional tests making use of reasonable data. Do not include any example outputs with your submission.

A single program file (with the `.c` extension) must be submitted electronically via the `guru` server. An example submission was conducted during the first week in lab00. If you have a concern regarding how to submit work, please visit course staff prior to the assignment deadline.

Problem: You recognize that being able to calculate where you stand in each of your courses is important to preparing for how you will approach finals week. Many of your classes have common categories of assignments such as homework, labs, quizzes, midterms, and a final. What may not be the same for each class is how much each category of assignment makes up the final letter grade of the course. For each of the previously mentioned categories, request from the user the weight each category carries in determining the course grade and what percentage of the possible points for each category have been earned. The final two inputs are the estimated score you believe you will earn on the final exam and desired course grade you would like to earn in the course. With this information you will calculate the weight carried by the final exam, the resulting course average earned given the estimated final exam score, and the final exam score needed to obtain the course grade you desire.

Example Execution #1:

```
Homework weight (%)-> 20
Homework average -> 100

Lab weight (%)-> 20
Lab average -> 100

Quiz weight (%)-> 20
Quiz average -> 100

Midterm weight (%)-> 20
Midterm average -> 100

Estimated final exam score -> 75
Desired course grade (%)-> 90

Final exam course weight: 20.00%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Estimated final exam score: 75.00%
Resulting course average: 95.00%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Desired course average: 90.00%
Accomplished with a final exam score of: 50.00%
```

Example Execution #1 Explained:

- Each of the five categories are worth 20% of the course grade. In this example the student has earned all possible points (100%) of each category.
- The student estimates that they could achieve a 75% on the final exam and would like to earn a 90% in the course.
- With a 75% on the final the total course grade would be 95%.
- To achieve a 90% in the course the student needs only a 50% on the final exam.

Example Execution #2:

```
Homework weight (%)-> 6.2
Homework average -> 90

Lab weight (%)-> 10.6
Lab average -> 92

Quiz weight (%)-> 21.3
Quiz average -> 87.5

Midterm weight (%)-> 35.4
Midterm average -> 81

Estimated final exam score -> 72
Desired course grade (%)-> 85

Final exam course weight: 26.50%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Estimated final exam score: 72.00%
Resulting course average: 81.72%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Desired course average: 85.00%
Accomplished with a final exam score of: 84.36%
```

Example Execution #3:

```
Homework weight (%)-> 10
Homework average -> 75

Lab weight (%)-> 0
Lab average -> 0

Quiz weight (%)-> 0
Quiz average -> 0

Midterm weight (%)-> 45
Midterm average -> 70

Estimated final exam score -> 60
Desired course grade (%)-> 70

Final exam course weight: 45.00%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Estimated final exam score: 60.00%
Resulting course average: 66.00%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Desired course average: 70.00%
Accomplished with a final exam score of: 68.89%
```

Example Execution #4:

```
Homework weight (%)-> 50
Homework average -> 55

Lab weight (%)-> 10
Lab average -> 60

Quiz weight (%)-> 10
Quiz average -> 60

Midterm weight (%)-> 15
Midterm average -> 55

Estimated final exam score -> 75
Desired course grade (%)-> 70

Final exam course weight: 15.00%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Estimated final exam score: 75.00%
Resulting course average: 59.00%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Desired course average: 70.00%
Accomplished with a final exam score of: 148.33%
```

Example Execution #5:

```
Homework weight (%)-> 50
Homework average -> 95

Lab weight (%)-> 10
Lab average -> 95

Quiz weight (%)-> 10
Quiz average -> 100

Midterm weight (%)-> 20
Midterm average -> 95

Estimated final exam score -> 95
Desired course grade (%)-> 80

Final exam course weight: 10.00%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Estimated final exam score: 95.00%
Resulting course average: 95.50%
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Desired course average: 80.00%
Accomplished with a final exam score of: -60.00%
```

Additional Notes:

- All floating-point variables must be of the `double` type.
- No negative values will be used to test your program.
- The sum of the weights for lab, homework, quiz, and midterm assignments will not exceed 100.
- Course standards prohibit the use of programming concepts not yet introduced in lecture. For this assignment, you can only consider material in the first 3 chapters of the book, notes, and lectures. Use of advanced programming constructs beyond this material would result in a loss of points.

4. Group Coordination

The next task is to collaborate as a group to determine who will submit your program assignment for grading. **Only one person per group will make submissions for the entire group**. Lab instructors are not required to grade submissions from multiple members of the same group to determine which submission you actually want graded. Also, the group member should not always be making the submission each week. Record the names and official Purdue career account e-mail addresses of all three lab partners here and put a checkmark in the Submitter column for the person responsible for making the submission. Your group should turn in this page of information to their lab instructor before you leave lab today.

Name	Purdue career account e-mail address	Submitter

If possible, it is a good idea to submit your work for grading prior to leaving lab today, even if it is not completely finished. This will allow each lab partner to **verify their contact information in the assignment header is correct** as each would then receive an e-mail verifying the submission. You may make multiple submissions before the deadline, but only the last attempt is retained and graded. Any previous submissions will be over-written and cannot be recovered. The submission script will reject the submission of any file that does not compile. A program must compile to be considered for partial credit.

If your group is unable to complete your assignment during the lab session, then it is expected that your group will meet outside of class to finish and submit the programming assignment. That is why you should exchange contact information during lab today! Before you leave, you should discuss when and where the group will meet next and when you plan for final submission to be made. If a group member you have entrusted to do the submission cannot be contacted and he/she is the only one who has a copy of the program, then the rest of the group would have to essentially start over in order to complete the programming assignment. Thus, it is a good idea for each person to have a copy of the program.

Consider the members of your group to be resources to assist you as you learn the material in this course. As a group, you may want to arrange a time to visit the TA office hours together or attend a particular Supplemental Instruction session together.

5. Lab Quiz

The final task will be to take the lab quiz which be made available in Blackboard during the last part of your lab today. **Lab quizzes are individual efforts and you may not use any resources while completing the quiz.** The quiz is worth 5 points and you must be present in lab session in order to credit for the quiz. All problems on lab quizzes will be multiple-choice or true-false. The quiz will emphasize material from book and the appropriate course programming standards. You will be given 10 minutes to complete the quiz and questions will be presented one at a time and cannot be revisited. Be sure to save your answers to each question and to finish your quiz to ensure it is submitted for grading. Any quiz that is taken when it is not being proctored by the lab instructor will receive zero points.