CS 158/159 Lab 05

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.

Solve the following problems related to material from Chapter 4:

Statement	**True / False**
In downward communication (passing by value) it is only a copy of the data that is sent from the calling function to the called function.	
The C language uses only pass by value and `return` to achieve communication of data between a calling and a called function.	
It is not possible to access a variable in the calling function by its identifier when inside the called function.	
Given the address of a variable the called function can access and manipulate the value of a variable in the calling function.	
To store a memory address that is sent from the calling function, the called function should use a special type of variable known as a pointer.	
To obtain the address of a variable we use the address operator (`&`).	
The asterisk (`*`) in a variable declaration indicates that the variables are not data variables but address variables holding the addresses of other variables in the program.	
The asterisk has two different uses, declaring an address variable (pointer) and indirectly accessing the data (in the memory location to which the variable points).	
When only one data item needs to be returned we should use the standard `return` statement.	

The scope of an object determines the region of the program in which it is visible.	
A variable declared in the local declaration section of a function has a scope that extends until the end of that function.	
Objects with a global scope are visible everywhere in the program.	
It is poor programming style to reuse identifiers within the same scope.	
A structure chart should be created before your program has been written.	
Each rectangle on a structure chart represents only the standard library functions used in a program.	
No code is contained in a structure chart and only demonstrates the function flow through the program.	
A structure chart may show the data that is exchanged between functions.	
Functional cohesion is a measure of how closely the processes in a function are related.	
A function that does one and only one process is functionally cohesive.	
It is a good design practice to limit user-defined functions to only a single task.	
It is a good design practice to not repeat the logic of one function in other functions of the program.	
It is a good design practice to design a user-defined function such that it is testable apart from the rest of the program.	

Solve the following problems related to material from Chapter 4:

Statement	**True / False**
It is possible to determine if any parameters are **passed** to a function **by address** from the declaration statement of the function.	
It is possible to determine if any parameters are **passed** to a function **by address** from an example call to the function.	
It is possible to determine if any parameters are **passed** to a function **by address** based on the first line of the definition of the function (also known as the function header).	
A function that **passes** at least one parameter **by address** must pass them all by address.	
All functions that utilize **pass by address** must be `void` functions.	
Rather than **passing** the only parameter **by address** to a `void` function it is better to make use of the `return` statement in the function to send the needed value to the calling function.	
One benefit of **pass by address** is that it allows multiple changes to be made in a function and to have those changes available in the calling function.	
With the use of **pass by address** it is now permissible for a function to be written to complete several sub-tasks of the program.	

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 are conducting analysis of an aircraft in steady flight. In steady and level flight, lift equals weight and thrust equals drag. You will be performing your analysis at a given (via input) altitude that will not exceed 11,000 meters. Program inputs in addition to altitude will include: flight angle of attack, zero-lift angle of attack, wing chord and span, and aircraft mass. Several calculations will need to account for atmospheric variations with altitude and these (temperature, speed of sound, and density) will be displayed in the first output segment.

Equations

$$T = T_0 + ah$$

T - temperature
h - altitude

$$\frac{\rho}{\rho_0} = \left(\frac{T}{T_0}\right)^{-\left[\frac{g_0}{aR}+1\right]}$$

ρ - Density

Speed of sound $c = \sqrt{\gamma R T}$

Mach number $M = \dfrac{V_\infty}{c}$

V_∞ Velocity

Values

$\rho_0 = 1.225\ \frac{\text{kg}}{\text{m}^3}$

$T_0 = 288.15$ K

$R = 287\ \frac{\text{J}}{\text{kg}\cdot\text{K}}$ (gas constant for air)

$\gamma = 1.4$ (ratio of specific heats for air)

$a = -6.5 \times 10^{-3}\ \frac{\text{K}}{\text{m}}$ (lapse rate)

$g_0 = 9.81\ \frac{\text{m}}{\text{s}^2}$ (consider constant gravity)

The coefficient of lift (c_l) is calculated from thin-airfoil theory. You will calculate and display the required flight speed (V) and display the speed regime (subsonic 'B', transonic 'T', or supersonic 'P') based on the Mach number.

Thin-Airfoil Theory: $c_l = 2\pi(\alpha - \alpha_{zl})$ with angles in **radians**.
α is the flight angle of attack and α_{zl} is the zero-lift angle of attack of the airfoil.

Lift Coefficient Definition: $c_l = \dfrac{L}{\frac{1}{2}\rho V_\infty^2 S}$, where L is lift and S is wing surface area, defined as chord times span. In steady and level flight, lift equals weight.

span times chord

Flight speeds are typically divided into different regimes based on Mach number:

$$\begin{array}{lllll} 0 & \le & M & < & 0.7 \quad \text{su(b)sonic} \\ 0.7 & \le & M & < & 1.2 \quad \text{(t)ransonic} \\ & & M & \ge & 1.2 \quad \text{su(p)ersonic} \end{array}$$

Example Execution #1 (F-18):

```
Enter flight altitude (m) -> 8000
Enter flight angle (degrees) -> 2
Enter zero-lift angle (degrees) -> 1
Enter wing span (meters) -> 12.3
Enter wing chord (meters) -> 3
Enter aircraft mass (kg) -> 20000

Atmospheric details
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Temperature (K):          236.15
Speed of Sound (m/s):     308.03
Density (kg/m^3):           0.52

Flight details
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Flight Speed (m/s):       429.83
Mach Number:                1.40
Flight Regime:                 P
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Example Execution #2 (Boeing 787):

```
Enter flight altitude (m) -> 5000
Enter flight angle (degrees) -> 4
Enter zero-lift angle (degrees) -> -3
Enter wing span (meters) -> 60
Enter wing chord (meters) -> 6.3
Enter aircraft mass (kg) -> 200000

Atmospheric details
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Temperature (K):           255.65
Speed of Sound (m/s):      320.50
Density (kg/m^3):            0.74

Flight details
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Flight Speed (m/s):        135.56
Mach Number:                 0.42
Flight Regime:                  B
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Example Execution #3 (F-22):

```
Enter flight altitude (m) -> 7000
Enter flight angle (degrees) -> 0
Enter zero-lift angle (degrees) -> -1
Enter wing span (meters) -> 13.5
Enter wing chord (meters) -> 5.75
Enter aircraft mass (kg) -> 35000

Atmospheric details
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Temperature (K):           242.65
Speed of Sound (m/s):      312.24
Density (kg/m^3):            0.59

Flight details
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Flight Speed (m/s):        370.01
Mach Number:                 1.19
Flight Regime:                  T
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Example Execution #4 (Airbus A380):

```
Enter flight altitude (m) -> 10000
Enter flight angle (degrees) -> 4
Enter zero-lift angle (degrees) -> -4
Enter wing span (meters) -> 80
Enter wing chord (meters) -> 10.5
Enter aircraft mass (kg) -> 400000

Atmospheric details
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Temperature (K):           223.15
Speed of Sound (m/s):      299.44
Density (kg/m^3):            0.41

Flight details
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Flight Speed (m/s):        160.69
Mach Number:                 0.54
Flight Regime:                  B
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Additional Notes:

- All floating-point variables should be of the double type.
- The use of selection, including logical and relational operators, is prohibited and would result in a loss of technique and output points.
- Course standards prohibit the use of programming concepts not yet introduced in lecture. For this assignment, you can only consider material in the first 4 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.