CS 158/159 Lab 04

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**
The function call is an executable statement.	
The function declaration requires the data types and identifiers for each parameter.	
The function call requires the data types and identifiers for each parameter.	
The function definition requires the data types and identifiers for each parameter.	
The function definition contains executable statements that perform the task of the function.	
The first line of the function definition terminates with a semicolon (`;`).	
The value of a local variable may be returned through a `return` statement of a user-defined function.	
Parameters are defined as local variables in the function header and should not be re-defined within the local declaration section of the function.	
Additional local variables can be defined in the local declaration section of a function.	
A local variable cannot be referenced through its identifier outside of the function for which it is defined.	
A variable declared in the local declaration section of a function can have the same	

identifier as one of the parameters of the function.	
Individual tasks in a program must be factored into individual user-defined functions.	
One benefit of user-defined functions is the potential reduction or elimination of duplicate code.	
A function header comment for every user-defined function must be inserted immediately above the definition of the function it is documenting.	
Parameters being received by a function will be commented to the right of where they are defined.	
The `return` statement cannot contain an expression.	
Data sent from the calling function to the function being called will be received in the same order in which it was passed.	
The individual task represented by a function should be testable apart from the rest of the program.	
A user-defined function may be called more than once in a program.	
The control of the program always returns from the calling function to the `main` function.	
A function may return at most one value.	
The role of the `main` function is to coordinate the function calls and to establish the data needs for a program.	

What is the output generated by problem #26 (page 222-223 of your C programming text)?

Given program below, write the function declaration for each user-defined function called from `main`:

```
int main()
{
 float annualRate;
 int term;
 float principal;
 float moPay;

 annualRate = getAnnualRate();
 term = getTerm();
 principal = getPrincipal();
 moPay = calcPayment(annualRate, principal, term);
 printAnalysis(term * 12, annualRate, principal, moPay);
 printAlternatives(term, annualRate, principal);
 return(0);
}
```

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: SpaceX is launching their Falcon Heavy Rocket for its maiden flight with the objective to orbit Mars. A Hohmann transfer is an elliptical transfer between two objects that circle the same body. The Hohmann transfer uses two engine impulses, one to move the spacecraft onto the elliptical transfer orbit and the second to move off of it (see image below moving from circular orbit 1 to circular orbit 3 via elliptical orbit 2). You are a part of a team which needs to calculate the changes in velocity that are needed for a successful Hohmann transfer. Given a starting orbital radius of a planet and a destination orbital radius of a second planet from the Sun, both in Astronomical Units (AU is [$1.496 * 10^{11}$ meters]), use the Hohmann equations to calculate the change in velocity needed to transfer from the starting orbit to the destination orbit.

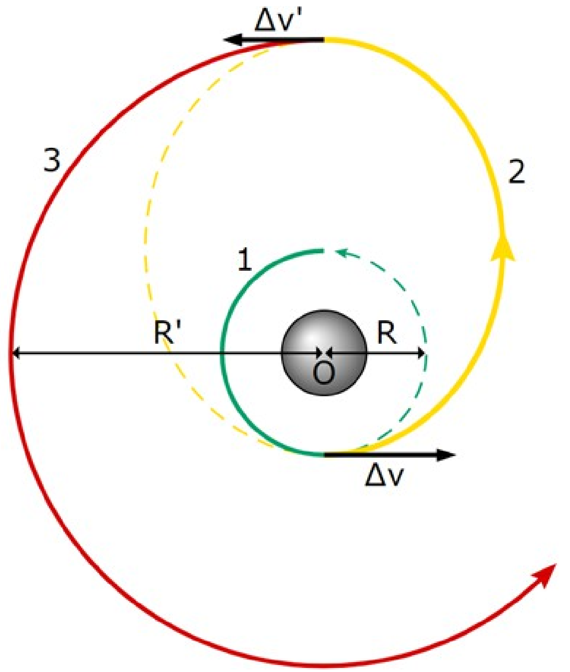

Image Source: https://commons.wikimedia.org/w/index.php?curid=1885233

Semi-major axis of transfer orbit: $a = \frac{R' + R}{2}$

where R and R' represent the orbit radius (in meters) of the first and second planetary bodies.

Velocity (m/s) of circular orbits*: $v = \sqrt{\frac{\mu}{radius}}$

where μ is standard gravitational parameter for the celestial body supporting the orbit (the Sun, $1.327124 * 10^{20}$).

*Planetary orbits are assumed to be circular for this problem.

Velocity (m/s) of elliptical connecting orbits:

$$v^2 = \mu * \left(\frac{2}{radius} - \frac{1}{a}\right)$$

Example Execution #1:

```
Enter starting orbital radius (AU) -> 1
Enter ending orbital radius (AU) -> 1.5

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Orbit Radius of starting planet from the Sun (m):        1.50e+11
Orbit Radius of destination planet from the Sun (m):     2.24e+11
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Insertion burn change in velocity (m/s):      2842.78
Arrival burn change in velocity (m/s):        2567.42
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Change in velocity to complete the Hohmann transfer (m/s):      5410.20
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Example Execution #2:

```
Enter starting orbital radius (AU) -> 0.7
Enter ending orbital radius (AU) -> 1

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Orbit Radius of starting planet from the Sun (m):        1.05e+11
Orbit Radius of destination planet from the Sun (m):     1.50e+11
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Insertion burn change in velocity (m/s):      3013.56
Arrival burn change in velocity (m/s):        2755.50
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Change in velocity to complete the Hohmann transfer (m/s):      5769.06
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Example Execution #3:

```
Enter starting orbital radius (AU) -> 1
Enter ending orbital radius (AU) -> 5.2

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Orbit Radius of starting planet from the Sun (m):        1.50e+11
Orbit Radius of destination planet from the Sun (m):     7.78e+11
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Insertion burn change in velocity (m/s):      8790.95
Arrival burn change in velocity (m/s):        5643.00
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Change in velocity to complete the Hohmann transfer (m/s):     14433.96
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Example Execution #4:

```
Enter starting orbital radius (AU) -> 0.4
Enter ending orbital radius (AU) -> 9.5

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Orbit Radius of starting planet from the Sun (m):        5.98e+10
Orbit Radius of destination planet from the Sun (m):     1.42e+12
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Insertion burn change in velocity (m/s):     18147.39
Arrival burn change in velocity (m/s):        6916.38
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Change in velocity to complete the Hohmann transfer (m/s):     25063.77
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Example Execution #5:

```
Enter starting orbital radius (AU) -> 1.5
Enter ending orbital radius (AU) -> 1

=================================================================
Orbit Radius of starting planet from the Sun (m):        2.24e+11
Orbit Radius of destination planet from the Sun (m):     1.50e+11
=================================================================
Insertion burn change in velocity (m/s):     -2567.42
Arrival burn change in velocity (m/s):       -2842.78
=====================================================
Change in velocity to complete the Hohmann transfer (m/s):      5410.20
=======================================================================
```

Additional Notes:

- All input will be positive values.
- The use of user-designed functions is required in this lab and every other future homework and lab assignment.
- 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.