Lab Partner 1 Name
Lab Partner 2 Name

CS 115 Fall 2019 Lab #1

Due: **Tuesday, September 3rd, 5:00 PM**
Points: **20**

Instructions:

1. Use this document template to report your answers. Enter all lab partner names at the top of first page.
2. You don't need to finish your lab work during the corresponding lab session.
3. Name the complete document as follows:

```
LastName_FirstName_CS115_Lab1_Report.doc
```

4. Submit the final document to Blackboard Assignments section before the due date. No late submissions will be accepted.
5. ALL lab partners need to submit a report, even if it is the same document.

Objectives:

1. (10 points) Demonstrate the ability to break a basic problem down into inputs, process and outputs.
2. (5 points) Demonstrate the ability to design test cases for your problem,
3. (5 points) Demonstrate the ability to select appropriate Java data type for the problem at hand.

Problem 1:

Break a basic problem down into **inputs**, **process** and **outputs** and write **pseudocode** (step by step sequence of necessary actions) to solve the problem. **NO Java CODE IS NEEDED**

Answer the following questions for the problems listed below. Populate provided tables (enter as many rows as you find necessary) and pseudocode boxes with your answers. Feel free to add extra tables, boxes, comments, etc. if needed.

INPUTS: What are the inputs?

- What format / data type are they? (integer, real number, single character, string - a sequence of characters)
- Any valid / invalid / illegal / special values? (positive, negative, valid range, etc.)
- How do you get them? (enter manually, ask user, read from file, etc.)

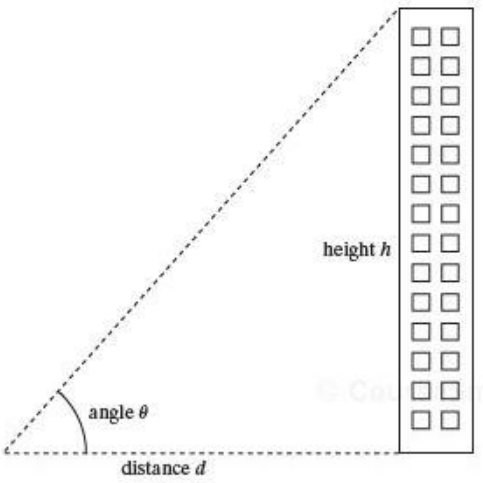
PROCESS: How do you get from inputs to the outputs you want?

- What are the calculation steps?
- To follow these steps, what else do you need? (formulas, etc.)
- Other variables, constants, conversions (besides input and output variables)

OUTPUTS: What are the outputs?

- What format / data type are they in? (integer, floating-point, character, or string)
- Any valid / invalid / illegal / special values? (positive, negative, valid range, etc.)
- How do you output them? (display on screen, save to a file, plot, tabularize, etc.)

1. You can use trigonometry to find the height of a building as shown on the figure below.



Suppose you can measure the angle theta (+/- 3 degrees) between the line of sight to the top of the building and the ground, and you can measure the distance d to the building. Calculate upper and lower estimate for the height of the building. **[3 points]**

Inputs and outputs (use "N/A", "undefined", "none", etc. If necessary)					
Variable name	Input or Output ?	Data type / format	Constraints	Special cases	Comments

Pseudocode:

What are the inputs? DISTANCE ANGLE
What format are they in? floating point
Any valid or invalid values? DISTANCE>0 0<ANGLE<90
you would just need to limit the angle from tolerance to 90, instead of 0 to 90
How do you get them? prompt user
How do you get from inputs to the outputs you want (process)?
What are the calculation steps?
HIGH_HEIGHT=DISTANCE * tan(ANGLE+3)
LOW_HEIGHT=DISTANCE * tan(ANGLE-3)

To follow these steps, what else do you need?
Other variables, constants, conversion? NONE
What are the outputs? HIGH_HEIGHT LOW HEIGHT
What format are they in? floating point
How do you output them? SCREEN

2. An outdoor mall is installing raised flower beds. The installer needs to know how many cubic yards of soil to order to fill the flower beds. Each bed is a rectangular box measured in inches (length by width by depth) and there are "count" number of beds. Calculate the total amount of soil to order. **[3 points]**

Inputs and outputs (use "N/A", "undefined", "none", etc. If necessary)					
Variable name	Input or Output ?	Data type / format	Constraints	Special cases	Comments

Pseudocode:

Inputs: length, width, depth each bed, count beds to fill.
Format: length/width/depth floating point, all greater than zero
count integer greater than zero
Prompt User

Process: volume=length*width*depth (inches cubed)
totalVolume=volume*count (inches cubed)
cubicYards = totalVolume/(36*36*36) CONVERT CUBIc INCHES TO CUBIC YARDS

Output: cubic yards of soil
Format: Floating point
Display the result to screen

3. If an ice cream cone is 6 inches tall, and its rim has a diameter of 2 inches, write pseudocode to determine the weight of the ice cream that can fit in the cone, assuming that the ice cream above the cone is a perfect hemisphere. You may neglect the thickness of the cone material. Assume that a gallon of ice cream weighs 8 lb and occupies 7.5 cubic feet. **[4 points]**

Inputs and outputs (use "N/A", "undefined", "none", etc. If necessary)					
Variable name	Input or Output ?	Data type / format	Constraints	Special cases	Comments

Pseudocode:

1.What are the inputs? Height (h) and diameter of cone
What format are they in? integer inches
How do you get them? constants given PI
2.How do you get from inputs to the outputs you want (process)?
What are the calculation steps?
r = diameter / 2
VolumeCone = 1/3 * pi * r^2 * h
VolumeHemisphere = 1/2 * (4/3 * pi * r^3)
TotalVolume = VolumeCone+ VolumeHemisphere (in inches cubed)
TotalVolumeFeetCubed = TotalVolume / 1728
densityIceCream = 8 / 7.5 (in lbs per cubic feet)
TotalWeight = TotalVolumeFeetCubed * densityIceCream

To follow these steps, what else do you need?
Other variables, constants, conversion? (besides input and output variables)
none

3.What are the outputs? TotalWeight
What format are they in? real
How do you output them? screen

Problem 2 (5 points):

Develop a test plan (a set of test cases) for the following problem. The goal of testing is to determine if the solution (a computer program) to your problem:

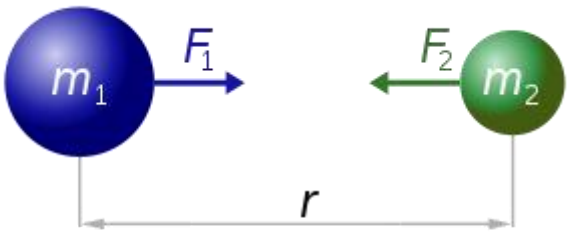
- Behaves correctly / produces correct results when given legal input values,
- Handles illegal input values correctly (for example: preventing division by zero),
- Behaves as planned when inputs are assigned "boundary" values.

In simple words, your test plan, at the very least, should:

- Consider "what could go wrong",
- Check if your problem solution behaves as expected.

NO Java CODE IS NEEDED.

1. Calculating the gravitational force F_1 (equal to F_2) between two bodies at a distance (see figure below [source: Wikipedia]) **[5 points]**:



$$F_1 = F_2 = G\frac{m_1 \times m_2}{r^2}$$

Not all test cases need to be included (division by zero is an absolute must, though). Ability to recognize them matters most. Some may be combined too.

Test case name	Input data set for this test	Explain why you chose this test

(ex. "negative height", "typical conditions", etc.)	case	case
Legal values	m1 > 0, m2 > 0, r > 0	To test if you get the expected results
Negative m1	m1 < 0, m2 > 0, r > 0	Check program response to illegal input
Negative m2	m1 > 0, m2 < 0, r > 0	Check program response to illegal input
Negative r	m1 > 0, m2 > 0, r < 0	Check program response to illegal input
r = 0	m1 > 0, m2 > 0, r == 0	Test division by zero
m1= 0	m1 == 0, m2 > 0, r > 0	Test program behavior at boundary / special case value
m2 = 0	m1 > 0, m2 == 0, r > 0	Test program behavior at boundary / special case value
m1 close to boundary value 0	m1 == 0.0001, m2 > 0, r > 0 Or m1 == -0.0001, m2 > 0, r > 0	Test program behavior when boundary value is being approached from either side
m2 close to boundary value 0	m2 == 0.0001, m1 > 0, r > 0 Or m2 == -0.0001, m1 > 0, r > 0	Test program behavior when boundary value is being approached from either side
r close to boundary value 0	r == 0.0001, m1 > 0, m2 > 0 Or r == -0.0001, m1 > 0, m2 > 0	Test program behavior when boundary value is being approached from either side
Combinations of the above		

Problem 3 (5 points):

Which Java data type and why would you use to store following data (if more than one can be chosen, explain your choice). **NO Java CODE IS NEEDED**:

2. US ZIP postal code (basic 5-digit one and NOT the ZIP+4 type) **[1 point]**:

Java data type(s)	Explanation

int	Integer values. 5 digits within allowed range. Can be long too.

3. United Kingdom postal code (see: https://en.wikipedia.org/wiki/Postal_code#United_Kingdom) **[1 point]**:

Java data type(s)	**Explanation**
String	It is a string, a sequence of alphanumerical characters.

4. Human age **[1 point]**:

Java data type(s)	**Explanation**
byte	Can be other integer data types too

5. Final CS 116 grade (A, B, C, D, E or F) **[1 point]**:

Java data type(s)	**Explanation**
char	Single character value.

6. Earth-Moon distance in yards (420388232.721 yards) **[1 point]**:

Java data type(s)	**Explanation**
double	Floating-point values.