CS 159 – Spring 2021 – Lab Exam #2

What will you submit? A single C-file will be submitted electronically via the `guru` server. An example submission was conducted as part of the Account Configuration Activity. If you have a concern regarding how to submit work, please contact course staff prior to the deadline for this, and all, assignments. The programming assignment is due on March 19, 2021 at 11:00pm (LOCAL WEST LAFAYETTE, IN TIME). **No late work will be accepted.**

What previous code can be referenced for this assignment? The problem is similar to that of previous labs this semester. You are permitted to access work previously developed by **yourself** for a homework assignment, work that was previously developed **by your lab team** for which you were credited as a contributing member, and any code found in the lectures, course notes packet, and C programming text of the course.

What help can be had from other individuals for this assignment? None. The course staff will respond to requests regarding understanding of the problem but questions regarding your solution and its development are not allowed. **Working with any other individual is prohibited.** All solutions will be compared for similarity. Any code identified on-line relevant to this assignment will be included in the similarity comparison. A violation of course policies as they relate to academic integrity will result in a failing grade for the course and a referral to the Office of the Dean of Students for additional disciplinary action.

Lab Exam #2 – Programming Assignment
Due: Friday March 19, 2021 at 11:00pm (time local to West Lafayette, IN)
20 Points Possible

Problem: Given three points (A, B, C) that will always create a triangle with a positive area, calculate the location of the orthocenter and create a triangle that connects the orthocenter to the end points of the largest side in the triangle ABC. Display the internal angles of this new triangle **from smallest to largest** and display whether it is isosceles, scalene, or **equilateral**, and whether it is acute, obtuse, or right.

How to determine whether two floating-point values are equal?

- The comparison two floating-point values for an EXACT match is prone error, for this assignment if the absolute value of the difference between two floating-point values is less than 0.001 then the values will be considered equivalent.

Example Execution #1:

```
Enter point #1 -> 1 1
Enter point #2 -> 5 2
Enter point #3 -> 3 4

Coordinates of orthocenter: 3.20, 3.20
Internal triangle angles: 31.0, 47.7, 101.3
Triangle Type: Scalene obtuse
```

Example Execution #2:

```
Enter point #1 -> 3 3
Enter point #2 -> -4 -1
Enter point #3 -> 3 5

Coordinates of orthocenter: 6.43, -1.00
Internal triangle angles: 40.6, 60.3, 79.1
Triangle Type: Scalene acute
```

Example Execution #3:

```
Enter point #1 -> 3 1
Enter point #2 -> -3 1
Enter point #3 -> 0 4

Coordinates of orthocenter: 0.00, 4.00
Internal triangle angles: 45.0, 45.0, 90.0
Triangle Type: Isosceles right
```

Example Execution #4 (rounding to one decimal place may result in angles that do not add up to exactly 180):

```
Enter point #1 -> 3 2
Enter point #2 -> -3 2
Enter point #3 -> 0 3

Coordinates of orthocenter: 0.00, 11.00
Internal triangle angles: 36.9, 71.6, 71.6
Triangle Type: Isosceles acute
```

Example Execution #5:

```
Enter point #1 -> -5 0
Enter point #2 -> -2 4
Enter point #3 -> 1 0

Coordinates of orthocenter: -2.00, 2.25
Internal triangle angles: 36.9, 36.9, 106.3
Triangle Type: Isosceles obtuse
```

Example Execution #6:

```
Enter point #1 -> 0 1.7320508
Enter point #2 -> -3 0
Enter point #3 -> 3 0

Coordinates of orthocenter: 0.00, 5.20
Internal triangle angles: 60.0, 60.0, 60.0
Triangle Type: Equilateral acute
```

All course programming and documentation standards are in effect for this and each assignment this semester. Please review this document!

Additional Requirements:

1. Add the **lab assignment header** (`vi` shortcut `hlb` while in command mode) to the top of your program. An appropriate description of your program must be included in the assignment header.
2. **Each of the example executions provided for your reference 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 example executions and an unknown number of additional tests making use of **reasonable data.** All numeric variables must be of the `double` type.
3. For this assignment you will be **required** to implement the user-defined functions (from chapter 4). Failing to follow course standards as they relate to good user-defined function use will result in a **zero for this assignment.**
4. Revisit **course standards as it relates what makes for good use of user-defined functions, what is acceptable to retain in the `main` function, and when passing parameters by address is appropriate.** In many cases user-defined function use should result in a `main` function that only declares variables and makes calls functions.

Additional Requirements (continued):

5. Course standards **prohibit** the use of programming concepts beyond the material found in the first five chapters of the book, notes, and lectures to be acceptable for use.
    - Each of the available selection constructs, logical operators, and relational operators are available for use in this assignment. **Do not** make use of the `bool` data type due to the extra include required.
    - The use of arrays, including character pointers to represent string data, or any technique of repetition would violate requirements of this assignment and **result in no credit being awarded for your effort.**
6. A program **MUST** compile, be submitted through the `guru` server prior to the posted due date to be considered for partial credit. The C-file you submit must be named exactly: `exam02.c`

Selected Course Programming and Documentation Standards Reminders:

- Code found inside the body of a selection construct must be indented two additional spaces.
- Make use of `{` and `}` with all relevant selection constructs.
- See page 258 of your C programming text regarding the proper indentation for a `switch` construct.

- Use the course function header (`head_fx vi` shortcut `hfx` while in command mode) for every user-defined function in your program.
    - List and comment **all parameters** to a function, one per line, in the course function header.
    - **All function declarations** will appear in the global declaration section of your program.
    - **The user-defined function definitions will appear in your program after the `main` function.**

- Maximize your use of symbolic/defined constants and minimize your use of literal constants.
- Place a **single space** between all operators and operands.
- Comment **all** variables to the right of each declaration. Declare only one variable per line.
- Select **meaningful identifiers** (names) for all variables in your program.
- Do not single (or double) space the entire program, **use blank lines when appropriate**.

When you submit... only the final successful submission is kept for grading. All other submissions are over-written and cannot be recovered. You may make multiple submissions but only the last attempt is retained and graded.

- Verify in the confirmation e-mail sent to you by the course that you have submitted the correct file (must be named `exam02.c`), to the correct assignment (`exam02`), and to the correct section.
- Leave time prior to the due date to seek assistance should you experience difficulties completing or submitting this assignment.
- All attempts to submit via a method other than through the `guru` server as set up in the Account Configuration Activity will be denied consideration.

Assignment deadlines... are firm and the electronic submission will disable promptly as advertised. We can only grade what you submit as expected prior to the assignment deadline.

Auto-Grade Tool

- We have implemented what is being referred to as the auto-grade tool. At the time of a successful assignment submission you may receive some feedback on your program in regards to course programming and documentation standards. This feedback may include a potential deduction that you will receive once your assignment is reviewed by your grader.
- It is expected that graders verify those notes identified by this tool to ensure that they are indeed applicable and reasonable to the submission. Graders may make additional deductions for those standards not identified by the new tool.
- We hope that this feedback helps with the enforcement of course standards, consistency in grading across sections, and to encourage students to revise their work when problems are identified before the assignment deadline passes. It is possible to resubmit an assignment for grading up to the advertised deadline. Only the final successful submission is retained and evaluated.