CS 159 – Spring 2021 – Lab Exam #1

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 February 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 #1 – Programming Assignment
Due: Friday February 19, 2021 at 11:00pm (time local to West Lafayette, IN)
20 Points Possible

Problem: The user will provide a positive integer value as the **first input** representing the options described below.

- 1 – A circle with an equilateral [T]riangle inscribed.
- 2 – A circle with a [S]quare inscribed.
- 3 or larger – A regular polygon with a [C]ircle inscribed. The number of sides of the polygon is equal to the integer input.

The second input is the area of the figure (circle for options 1 and 2, polygon for 3 or larger). Your program will calculate the radius or apothem and area of the inscribed figure.

Example Execution #1 (circle with [T]riangle inscribed):

```
Enter desired option -> 1
Enter area of figure -> 150

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Radius/Apothem of figure with area given:       6.9
Area of inscribed figure [T]: 62.0
```

Example Execution #2 (circle with [S]quare inscribed):

```
Enter desired option -> 2
Enter area of figure -> 255.45

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Radius/Apothem of figure with area given:       9.0
Area of inscribed figure [S]: 162.6
```

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

Example Execution #3 (Pentagon with [C]ircle inscribed):

```
Enter desired option -> 5
Enter area of figure -> 5500

-=================================================-
Radius/Apothem of figure with area given:      38.9
Area of inscribed figure [C]: 4756.4
```

Example Execution #4:

```
Enter desired option -> 10
Enter area of figure -> 52000

-=================================================-
Radius/Apothem of figure with area given:     126.5
Area of inscribed figure [C]: 50277.9
```

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.

Include the Purdue University e-mail addresses of **each contributing group member** 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, do not add any "bonus" features not 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.**
   - The input of the option will always be a positive integer value and all floating-point variables must be of the `double` type.
   - Use the constant value `M_PI` where needed for the constant pi.
   - The number of sides of the regular polygon will always be a positive **integer** value.
3. Course standards **prohibit** the use of programming concepts beyond the material found in the first three chapters of the book, notes, and lectures to be acceptable for use.
   - Any use of logical operators, relational operators, `bool` variables, or selection constructs is **prohibited** and would violate requirements of this assignment **resulting in no credit being awarded for your effort.**
   - **All code for this assignment will be placed inside of the `main` function.** User-defined functions will be a requirement beginning with lab #4.
4. 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: `exam01.c`

Course Programming and Documentation Standards Reminders:

- Maximize your use of symbolic/defined constants and minimize your use of literal constants.
- Indent all code found within the `main` function **exactly** two spaces.
- Place a **single space** between all operators and operands.
- Comment **all** variables to the right of each declaration. Declare only one variable per line.
- Notice that several programs (see program 2-9 on pages 74-75) in the programming text use a single line comment to indicate the start of the local declaration and executable statement sections of the `main` function.
  - At no point during the semester should these two sections ever overlap. You might consider adopting this habit of commenting the start of each section to help you avoid this mistake.
- 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 `exam01.c`), to the correct assignment (`exam01`), 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.