

Faculty of Science

**Course**: CSCI 1030U: Introduction to Computer Science with C++

**Lab:** #3

Topic: Iteration and Numerical Methods

***Note****: This lab assignment assumes that you have completed your reading assignment from last week, particularly chapter 3 of your textbook.*

**Part 1**

For the first part of this lab, you will use loops and conditionals to find an approximation for the square root of a positive number. Below is pseudocode which estimates the square root of a positive number:

epsilon = 0.01

guess = num / 2.0

while (abs(guess^2 - num) >= epsilon) do

guess = guess - (guess^2 - num) / (2 \* guess)

endwhile

print guess

Write this program in C++. Explore the effects of using different values of epsilon (e.g. 0.1, 0.000001). Does it take significant extra time to gain the extra precision?

**Part 2**

For the second part of this lab, you write your own program to approximate the value of Pi (π), using the method proposed by Archimedes (circa 250 BCE). Archimedes started by imagining a circle with circumference equal to π (i.e. a circle with diameter 1, radius ½). He then imagined successive polygons within the circle, calculating the sum of the polygon’s edges as an approximation to the circumference (and thus π). As the number of sides (n) in the polygon increases, the sum of the polygon’s sides becomes a better and better estimate for π.

*Note: A similar technique, where the polygons are outside of the circle, can also be used.*

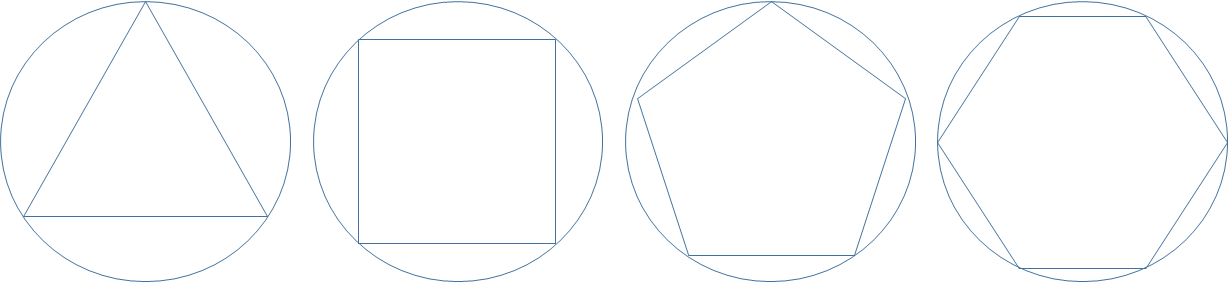


Figure 3.1 – Successive Polygons within a Circle

Each side in the polygon can be assumed to be the same length. To calculate the sides’ length, we’ll divide the polygon into right triangles. An example has been shown below for the hexagon:

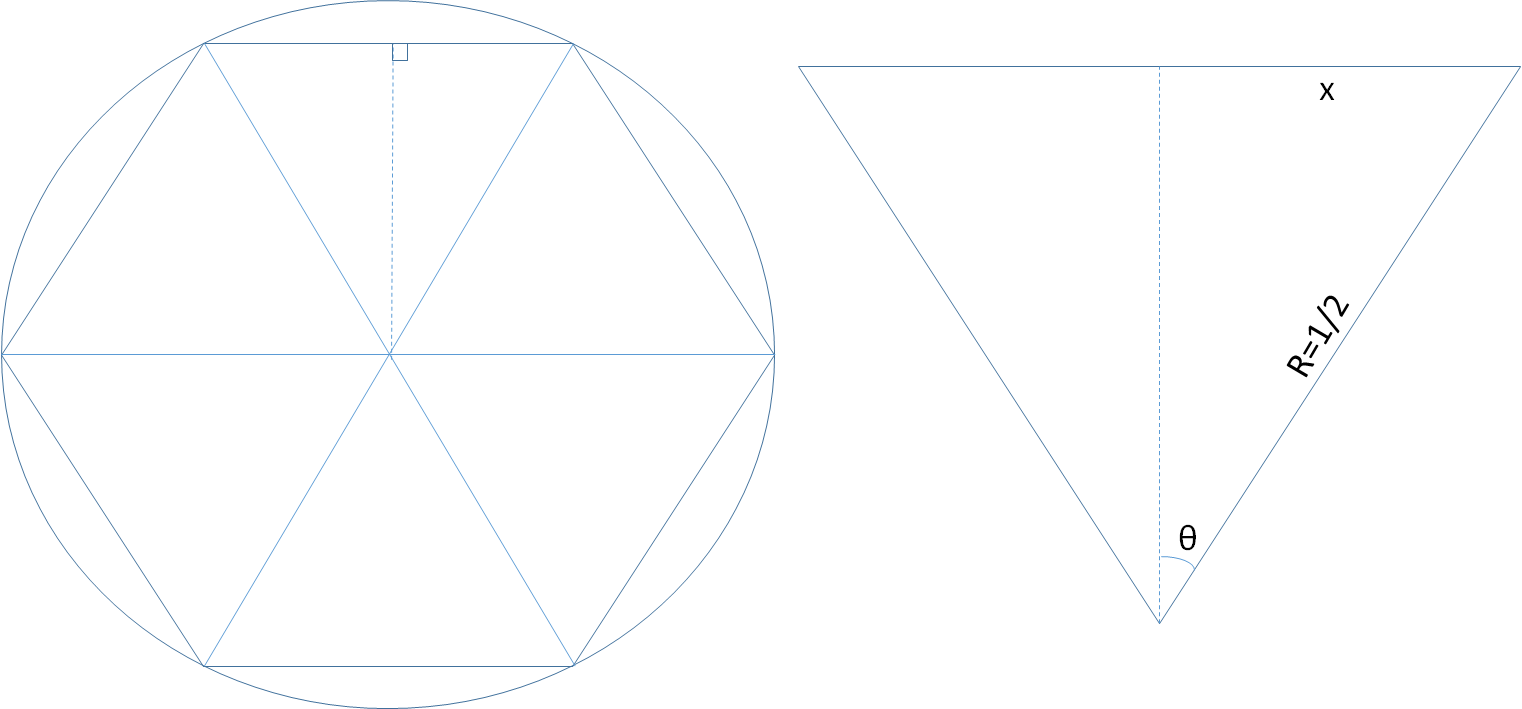


Figure 3.2 – Viewing the Polygon as a Triangle, to Calculate Side Length

From figure 3.2, we could use our knowledge of trigonometry, to figure out that sin θ = OPP/HYP = 2x, thus x = (sin θ)/2. x is the length of half of one side of the polygon, so the sum of all the sides is 2n\*x. Similarly, the total of all of the angles around the centre of such a polygon should be 360 degrees. We can then figure out that θ = 360°/2n, giving us the following formula for the estimate of π.

***Note****: For more details about Achimedes’ method, check out the following video:* https://www.youtube.com/watch?v=\_rJdkhlWZVQ

|  |  |  |
| --- | --- | --- |
| **Function** | **Description** | **Example Use** |
| abs | Calculates the absolute value | abs(n) |
| sin | Given an angle in radians, calculates the sin of the angle | sin(theta) |

*Table 3.1 – A list of useful functions for this lab assignment*

***Note****: To convert degrees to radians, use the following formula:*

double radians = degrees \* 0.0174533;

For the second part of this lab, write a loop that tries successive approximations for π, starting with n = 3 and continuing until the estimate is close enough to π. At the top of your program, include the following two lines of code to allow the use of math functions (sin() and abs()), and define a reasonable approximation of π, for comparison.

#include <math.h>

#define PI 3.14159265358979323846

The value PI, defined above, can be used as a value, in place of π. Use an epsilon of 0.000000001. When you find the result, print the estimate (with a label), and also print the polygon size (n). How many iterations does it take? Can you think of any ways to improve this algorithm?

**How to Submit**

If you finish this lab assignment before the end of the lab period, present the code for both parts of this lab assignment to your TA before you leave, and submit the C++ file (containing both parts, including comments) to the drop box corresponding with this lab assignment on Blackboard. If you are unable to complete the lab assignment before the end of the lab period, you can merely submit the Python file (containing all parts of the lab assignment) to the corresponding drop box on Blackboard.

***Note****: Comments are mandatory. Even though this lab is not worth any marks, assignments are. Be sure to get into the habit of properly documenting your work, since it also will help you understand your own program after a break between programming sessions.*