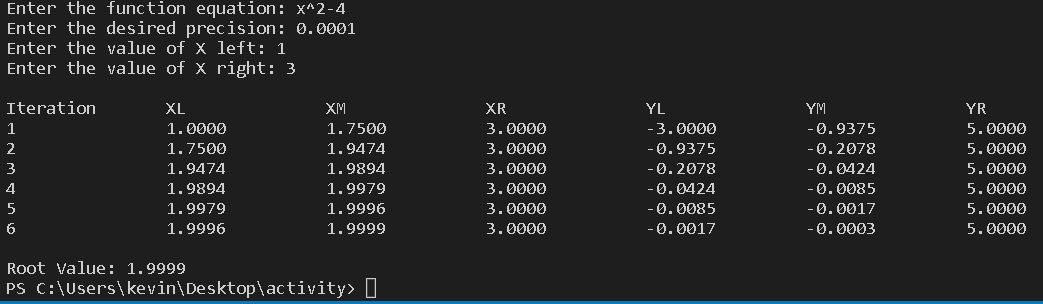
**Falsi**

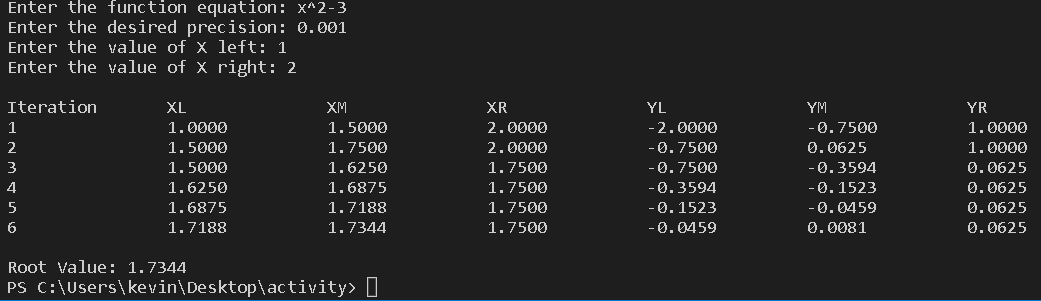
Code:

Output:

Appendix: Figure di ko alam Shows a class named FalsePositionMethod it contains the main method responsible for obtaining user input and calling the method named “*falsePositionMethod*” that performs the calculation of falsi method, and “*evaluateFunction*” method that evaluate the equation and retrieves the value of Y left and Y Right.

**Bisection**

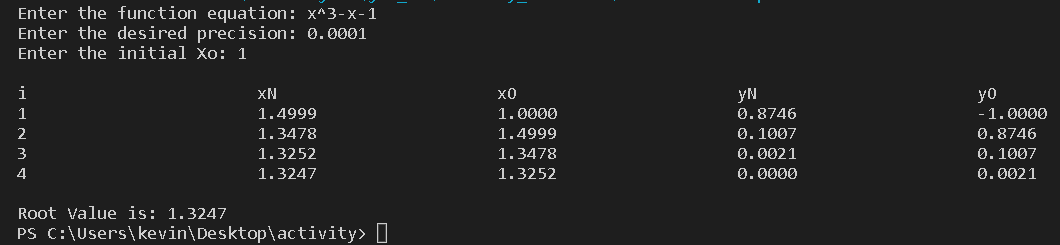
Code:

Output:

Appendix: Figure di ko alam BisectionMethod class represents a class that contains the main method, which is responsible for getting input from the user. It then calls two methods: the “*bisectionMethod”* and “*evaluateFunction”*. The “*bisectionMethod”* calculates the solution using the bisection method, while the “*evaluateFunction”* method evaluates the equation and retrieves the values of Y left and Y right.

**Newton Rhapson**

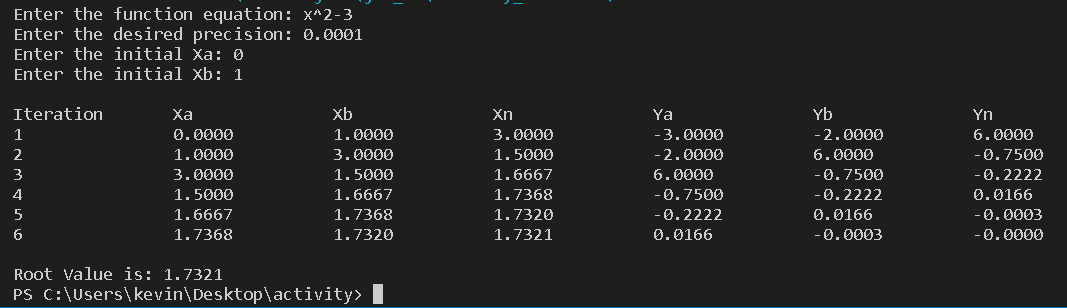
Code:

Output:

Appendix: Figure di ko alam The NewtonRaphsonMethod class serves as the starting point of the program, where the main method acts as the entry point for execution. It requires the user to input the function equation, desired precision, and initial guess for the root (Xo). Upon receiving the inputs, the program calls the newtonRaphsonMethod a method that perform the Newton-Raphson and it determine the root of the equation.

**Secant**

Code:

Output:

Appendix: Figure di ko alam SecantMethod a class that starts with the main method, which acts as the starting point. It asks the user to provide the equation, desired precision, and initial values for Xa and Xb. These values are then used in the secantMethod method, which handles the actual calculations