*Owen Lindsey*

Professor Demland, David

CST-201 *Exercise 1*  
9/9/2024

**Exercise 1.1 - 5**

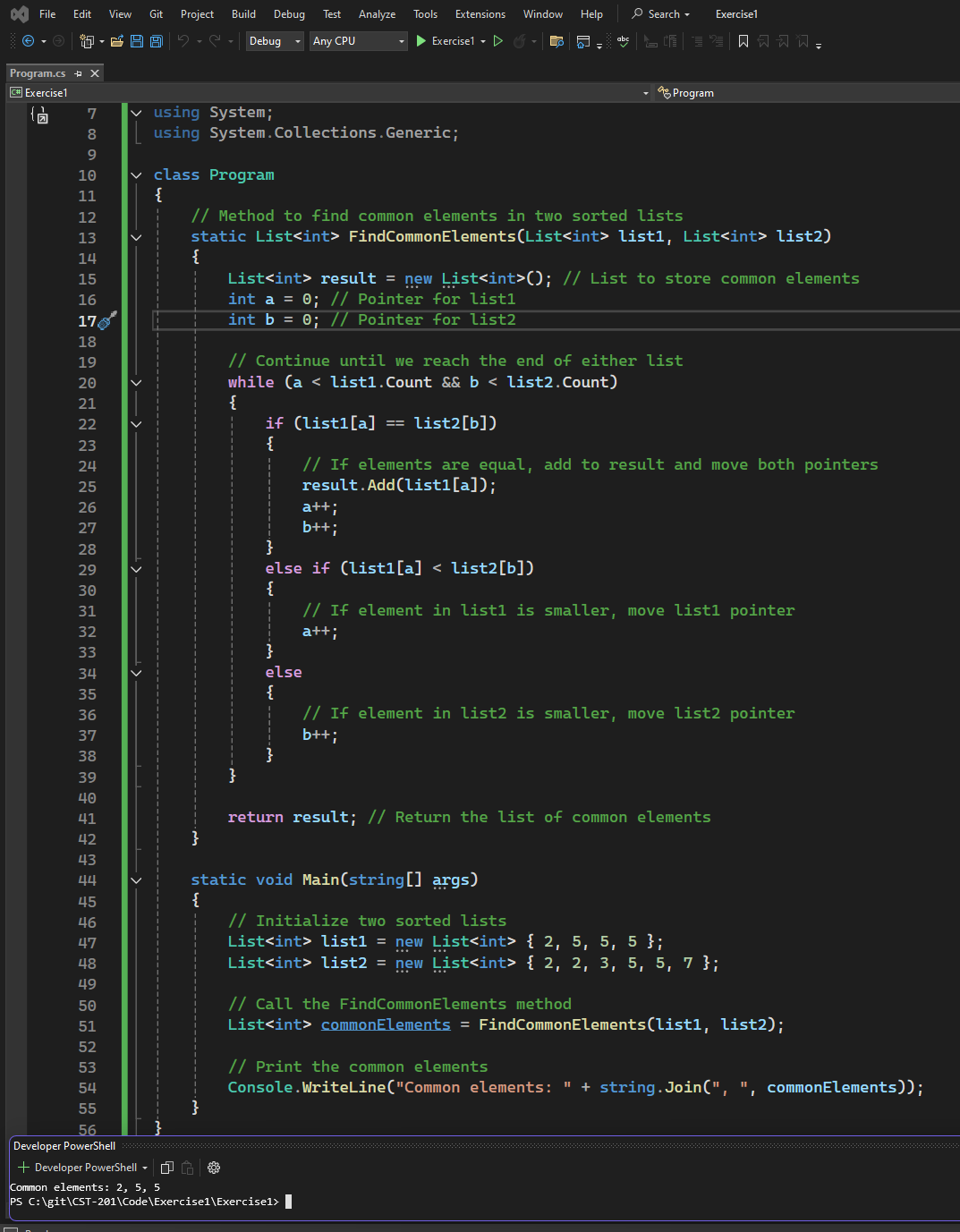
*Algorithm Explanation*

1.) We define a FindCommonElements method that takes two List<int> parameters and returns a List<int> of common elements.

2.) Inside the method, we use the same two-pointer approach to iterate through both lists simultaneously.

3.) We use a while loop to continue until we reach the end of either list.

4.) In each iteration, we compare the elements and update the pointers.   
  
5.) When we find a matching, we add them to the result list.   
  
6.) In the main method, we create two example lists, call out FindCommonElements method, and print the result.

*For code see attached screenshot with output*

**Exercise 1.2 - 4**

*Algorithm Explanation*

1.) Check if a is Zero. If it is, the equation is not quadratic, so we output a message and exit.

2.) We calculate the discriminant (b² - 4ac), which determines the nature of the roots.

3.) Based on the discriminant, there are 3 cases:

A. If positive, there are two distinct real roots. Using the quadratic formula: (-b ± √(discriminant)) / (2a)

B. If the discriminant is zero, there is one repeated real root. Calculated: -b / (2a)

C. If the discriminant is negative, there are no real roots.

Output the roots based on the case:

For case A: Output both real roots

For case B: Output the single real root

For case C: Output a message stating no real roots exist

**Exercise 1.2 - 4**  
*Pseduocode Explanation*

ALGORITHM FindRealQuadraticRoots(a, b, c)

// Check if it's a quadratic equation

IF a = 0 THEN

OUTPUT "This is not a quadratic equation"

RETURN

END IF

// Calculate the discriminant

discriminant = b^2 - 4\*a\*c

// Check the nature of roots based on the discriminant

IF discriminant >= 0 THEN

// At least one real root exists

root1 = (-b + sqrt(discriminant)) / (2\*a)

root2 = (-b - sqrt(discriminant)) / (2\*a)

IF discriminant > 0 THEN

OUTPUT "Two distinct real roots:"

OUTPUT "Root 1 =", root1

OUTPUT "Root 2 =", root2

ELSE // discriminant = 0

OUTPUT "One repeated real root:"

OUTPUT "Root =", root1 // or root2, they're the same

END IF ELSE // No real roots OUTPUT "No real roots exist"

END IF END ALGORITHM

**Exercise 1.2 -5**  
*Algorithim Explanation*

1.) Start with the decimal number that needs to be converted.

2. Divide the number by 2.

3. Keep the rightmost digit of the remainder (which will always be either 0 or 1) and discard the others.

4. Take the result of the division (ignoring any remainder) and repeat steps 2-4 until the result becomes 0.

5. The kept digits, read from bottom to top (or right to left), give the binary representation of the original decimal number.

**Exercise 1.2 -5**  
*Psuedocode Explanation*

// Handle the special case of input being 0  
ALGORITHM DecimalToBinary(decimal\_num)

IF decimal\_num = 0 THEN // 0 in decimal is 0 in binary

RETURN "0"

END IF

// Initialize an empty string to store the binary representation

binary\_string = ""

// Continue the process until the decimal number becomes 0

WHILE decimal\_num > 0 DO

// Get the remainder when divided by 2 (will be either 0 or 1)

remainder = decimal\_num MOD 2

// Add the remainder to the left side of the binary string

// This builds the binary number from right to left

binary\_string = CONCATENATE(CONVERT\_TO\_STRING(remainder), binary\_string)

// Integer divide the decimal number by 2 for the next iteration

decimal\_num = decimal\_num DIV 2

END WHILE

// Return the complete binary representation

RETURN binary\_string

END ALGORITHM