JULIA LAB PROGRAMS - BDSL456D

PROG 1

a. Develop a Julia program to simulate a calculator (for integer and real numbers).

```
# Define function for addition
function add(x, y)
    return x + y
end

# Define function for subtraction
function subtract(x, y)
    return x - y
end

# Define function for multiplication
function multiply(x, y)
    return x * y
end

# Define function for division
function divide(x, y)
    if y != 0
        return x / y
    else
        println("Error: Division by zero!")
        return NaN
    end
end

# Main function to perform calculator operations
function calculator()
    println("Welcome to the calculator simulation!")
    println("Please select an operation:")
    println("1. Addition (+)")
    println("2. Subtraction (-)")
    println("3. Multiplication (*)")
    println("4. Division (/)")
    operation = readline()

    println("Enter first number:")
    num1 = parse(Float64, readline())

    println("Enter second number:")
```

```
    num2 = parse(Float64, readline())

    if operation == "+" || operation == "1"
        result = add(num1, num2)
        println("Result: $result")
    elseif operation == "-" || operation == "2"
        result = subtract(num1, num2)
        println("Result: $result")
    elseif operation == "*" || operation == "3"
        result = multiply(num1, num2)
        println("Result: $result")
    elseif operation == "/" || operation == "4"
        result = divide(num1, num2)
        println("Result: $result")
    else
        println("Invalid operation selected!")
    end
end

# Call the calculator function to start the program calculator()
```

```
julia> calculator()
Welcome to the calculator simulation!
Please select an operation:
1. Addition (+)
2. Subtraction (-)
3. Multiplication (*)
4. Division (/)
4
Enter first number:
4
Enter second number:
0
Error: Division by zero!
Result: NaN
```

b. Develop a Julia program to add, subtract, multiply and divide complex numbers.

```
# Define function for complex number addition
function complex_add(z1::Complex, z2::Complex)
    return z1 + z2
end

# Define function for complex number subtraction
function complex_subtract(z1::Complex, z2::Complex)
    return z1 - z2
```

```
end

# Define function for complex number multiplication
function complex_multiply(z1::Complex, z2::Complex)
    return z1 * z2
end

# Define function for complex number division
function complex_divide(z1::Complex, z2::Complex)
    return z1 / z2
end

# Main function to perform complex number operations
function complex_calculator()
    println("Welcome to the complex number calculator!")
    println("Please select an operation:")
    println("1. Addition (+)")
    println("2. Subtraction (-)")
    println("3. Multiplication (*)")
    println("4. Division (/)")
    operation = readline()

    println("Enter the real part of the first complex number:")
    real1 = parse(Float64, readline())

    println("Enter the imaginary part of the first complex number:")
    imag1 = parse(Float64, readline())

    println("Enter the real part of the second complex number:")
    real2 = parse(Float64, readline())

    println("Enter the imaginary part of the second complex number:")
    imag2 = parse(Float64, readline())

    z1 = complex(real1, imag1)
    z2 = complex(real2, imag2)

    if operation == "+" || operation == "1"
        result = complex_add(z1, z2)
        println("Result: $result")
    elseif operation == "-" || operation == "2"
        result = complex_subtract(z1, z2)
        println("Result: $result")
    elseif operation == "*" || operation == "3"
        result = complex_multiply(z1, z2)
        println("Result: $result")
    elseif operation == "/" || operation == "4"
        result = complex_divide(z1, z2)
```

```
        println("Result: $result")
    else
        println("Invalid operation selected!")
    end
end

# Call the complex_calculator function to start the program complex calculator ()
```

```
julia> # Call the calculator function to start the program

julia> calculator()
Welcome to the calculator simulation!
Please select an operation:
1. Addition (+)
2. Subtraction (-)
3. Multiplication (*)
4. Division (/)
4
Enter first number:
4
Enter second number:
0
Error: Division by zero!
Result: NaN
```

c. Develop a Julia program to evaluate expressions having mixed data types (integer, real, floating-point number and complex).

```
# Define function to evaluate mixed data type expressions
function evaluate_expression(expr::String)
    # Use the Meta.parse function to parse the input expression
    parsed_expr = Meta.parse(expr)

    # Evaluate the parsed expression
    result = eval(parsed_expr)

    return result
end

# Main function to input expression and display result
function main()
    println("Welcome to the mixed data type expression evaluator!")
    println("Please enter the expression to evaluate:")
    expr = readline()

    try
        result = evaluate_expression(expr)
```

```
        println("Result: $result")
    catch e
        println("Error: $e")
    end
end

# Call the main function to start the program main()
```

```
julia> main()
Welcome to the mixed data type expression evaluator!
Please enter the expression to evaluate:
1+3.45
Result: 4.45
```