# ***LAB\_1 F#:***

# ***Moses Shaji Varghese***

*A white text with black text

AI-generated content may be incorrect.*

*CODE:*

*open System*

*// 1. Partial Application*

*let power exponent value =*

*value \*\* float exponent*

*let square = power 2.0*

*let cube = power 3.0*

*printfn "Square of 4: %f" (square 4.0)*

*printfn "Cube of 3: %f" (cube 3.0)*

*A screen shot of a computer program

AI-generated content may be incorrect.*

*OUTPUT  
A black background with white text

AI-generated content may be incorrect.*

*A black text on a white background

AI-generated content may be incorrect.*

*CODE:*

*open System*

*// 2. Tail Recursion 1: Product of all elements in a list*

*let rec productTailRec (lst: int list) (acc: int) =*

*match lst with*

*| [] -> acc*

*| h::t -> productTailRec t (acc \* h)*

*let product lst = productTailRec lst 1*

*printfn "Product of list: %d" (product [1; 2; 3; 4; 5])*

*A screen shot of a computer program

AI-generated content may be incorrect.*

*OUTPUT:*

*A black background with white text

AI-generated content may be incorrect.*

*A white background with black text

AI-generated content may be incorrect.*

*CODE:*

*open System*

*// 3. Tail Recursion 2: Product of odd numbers down to 1*

*let rec productOddTailRec n acc =*

*if n <= 1 then acc*

*else productOddTailRec (n - 2) (acc \* n)*

*let productOdd n = productOddTailRec n 1*

*printfn "Product of odd numbers from 11 to 1: %d" (productOdd 11)*

*A screen shot of a computer program

AI-generated content may be incorrect.*

*OUTPUT:*

*A black background with white text

AI-generated content may be incorrect.*

*A screenshot of a computer

AI-generated content may be incorrect.  
CODE:*

*open System*

*// 4. Using Map Function with a Collection*

*let names = [" Charles"; "Babbage  "; "  Von Neumann  "; "  Dennis Ritchie  "]*

*let trimmedNames = List.map (fun (name: string) -> name.Trim()) names*

*printfn "Trimmed Names: %A" trimmedNames*

*A black background with text

AI-generated content may be incorrect.*

*OUTPUT:*

*A black background with white text

AI-generated content may be incorrect.*

*A white background with black text

AI-generated content may be incorrect.*

*CODE:*

*open System*

*// 5. Using Filter and Reduce with a Collection*

*let numbers = [1..700]*

*let filteredNumbers = List.filter (fun n -> n % 7 = 0 && n % 5 = 0) numbers*

*let sumFilteredNumbers = List.fold (+) 0 filteredNumbers*

*printfn "Sum of multiples of 7 and 5: %d" sumFilteredNumbers*

*A screen shot of a computer code

AI-generated content may be incorrect.*

*OUTPUT:*

*A black background with white text

AI-generated content may be incorrect.*

*A white text with black text

AI-generated content may be incorrect.*

*CODE:*

*// List of names*

*let names = ["James"; "Robert"; "John"; "William"; "Michael"; "David"; "Richard"]*

*// Filter names containing the letter "I" and concatenate them using fold*

*let concatenatedNamesWithI =*

*names*

*|> List.filter (fun name -> name.Contains("I"))  // Filter names that contain the letter "I"*

*|> List.fold (fun acc name -> acc + name) ""     // Concatenate filtered names using fold*

*// Print the concatenated names*

*printfn "Concatenated names with 'I': %s" concatenatedNamesWithI*

*[<EntryPoint>]*

*let main argv =*

*0 // Return an integer exit code*

*A computer screen with colorful text

AI-generated content may be incorrect.*

*OUTPUT:*

*A black background with white text

AI-generated content may be incorrect.*

*GITHUB PART*

*A screenshot of a computer

AI-generated content may be incorrect.*

*https://github.com/Lycanroc0409/Lab1f-.git*