Test 01

Programming Paradigms

Naveen

Q 1 . **Mapping, Filtering through Lists**

Code :

```fsharp
let salaries = [75_000; 48_000; 120_000; 190_000; 300_113; 92_000; 36_000]

let calculateTax salary =
    if salary <= 49020 then float salary * 0.15
    elif salary <= 98040 then
        49020.0 * 0.15 + float (salary - 49020) * 0.205
    elif salary <= 151978 then
        49020.0 * 0.15 + 49020.0 * 0.205 + float (salary - 98040) * 0.26
    elif salary <= 216511 then
        49020.0 * 0.15 + 49020.0 * 0.205 + 53838.0 * 0.26 + float (salary - 151978) * 0.2932
    else
        49020.0 * 0.15 + 49020.0 * 0.205 + 53838.0 * 0.26 + 64533.0 * 0.2932 + float (salary - 216511) * 0.33

let highIncome = salaries |> List.filter (fun s -> s > 100000)
let taxAmounts = salaries |> List.map calculateTax
let boostedLow = salaries |> List.map (fun s -> if s < 49020 then s + 20000 else s)

// Fix: provide an initial accumulator value for the fold
let midRangeSum =
    salaries
    |> List.filter (fun s -> s >= 50000 && s <= 100000)
    |> List.fold (fun acc x -> acc + x) 0

printfn "High Income Salaries: %A" highIncome
printfn "Tax Amounts: %A" taxAmounts
printfn "Boosted Low Salaries: %A" boostedLow
printfn "Mid Range Sum: %d" midRangeSum
```

Output :

```
kuhar@fedora:~/Documents/Learn/F#/Test 01

High Income Salaries: [120000; 190000; 300113]
Tax Amounts: [12678.9; 7200.0; 23111.7; 42548.0304; 77909.7156; 16163.9; 5400.0]
Boosted Low Salaries: [75000; 68000; 120000; 190000; 300113; 92000; 56000]
Mid Range Sum: 167000
kuhar@fedora:~/Documents/Learn/F#/Test 01$
```

Q2 : Tail Recursion

Code :

```fsharp
let sumMultiplesOf3 n =
    let rec loop current acc =
        if current > n then acc
        else loop (current + 3) (acc + current)
    loop 3 0
printf "Enter a number: "
let userInput = System.Console.ReadLine() |> int
printfn "%d" (sumMultiplesOf3 userInput)
```

Output :