SOFTWARE ENGINEERING CS 487
Homework #3

Name: Anirudha Kapileshwari
Email: akapileshwari@hawk.iit.edu

Design a component which validates input values.

1. **Show pseudo-code for a module which must assess an integer against a valid range. It must return a valid value as close as possible to the input and an indicator of the returned value's reliability. (2 pts)**

=>
Intiger Validation module:

```
Function validate_integer(input_value, min_value, max_value)
    If input_value is not an integer
        Return None, "invalid data type"
    End if

    valid_value= Max(min_v, min(input_value, max_value))

    If valid_value equals input_value
        reliability ="High"
    Esle

        reliability ="Low"
    End If

    Return valid_value,realibility
End function
```

Validates an intiger with specific range and returns the closest valid value and its reliability

2. Make it reusable – identify the code segments that can remain as is and those which need to be modified such that the component could be used to process other data types (e.g., date and string values) (1 pt)
=>
Making it reusable

```
Function process_data(input_value, validation_function, extra_parameters)
    valid_value, reliability = validate_input(input_value, validator_function, extra_parameters)

    if reliability equals"High"
        process_automatically(valid_value)
    else
        alert_humans(input_value)
    end if
End Function

Function validate_integer(input_value, min_value, max_value)
    //integer specific logic
End Function

Function validate_date(input_value, min_value, max_value)
    //date specific logic
End Function
```

```
Function validate_string(input_value, min_value, max_value)
        //string specific logic
End Function
```

Unchanged -> the validate_input function is generic and each function takes an input value and validation parameters, returning a valid value and reliability.

Changed-> different data type like integers, date, string can be validated also the parameter of each specific validator function depends on the data type and validation criteria

3. Use pseudo-code to show how an automated mission-critical system would manage the exception of a "less-than-perfectly-reliable" value. Alert the humans if automated processing is too risky. (2 pts)

=>

Handling less then perfectly reliable values

```
Function process_data(input_value, validation_function, extra_parameters)
        valid_value, reliability = validate_input(input_value,validator_function, extra_parameters)

        if reliability equals"High"
                process_automitacally(valid_value)
        else
                alert_humans(input_value)
        end if
End Function

Function process_automitacally(valid_value)
        //automated processing logic
End Function

Function alert_humans(input_value)
        //Logic to alert humans for interaction
End Function
```

Uses reliability information to decide between automated processing or alerting human for manual validation