Student Name: ________________________________ Student ID: __________________ TA: ________________________

Rust Lab 05

30/7/2025

1. Write a complete Rust program that demonstrates the use of functions, keyboard input, and argument passing by reference.

   Requirements:

   1) Prompt the user to select a temperature conversion mode:
      - Enter 1 to convert from **Celsius to Fahrenheit**
      - Enter 2 to convert from **Fahrenheit to Celsius**
   2) Prompt the user to input a **temperature value** (as a floating-point number) via keyboard.
   3) Based on the user's selection, convert the temperature using one of the following functions:

      fn celsius_to_fahrenheit(c: f64) -> f64

      fn fahrenheit_to_celsius(f: f64) -> f64

      Each function should return the converted temperature.
   4) Display the converted result by calling:

      fn show_temperature(label: &str, value: &f64)

      This function should print a label and the temperature with 2 decimal places.
   5) Next, simulate a calibration or correction by adjusting the **original input value**:

      fn adjust_temperature(value: &mut f64, delta: f64)

      - Pass the original temperature as a mutable reference (&mut f64)
      - Add a small delta (e.g., +0.5) to simulate sensor correction or rounding offset
   6) Use show_temperature() to display the adjusted original input value.

TA Check: 1) Input: ______________ 2) Conversion ________________ 3) Display ______________ 4) Adjust_______________

2. Write a recursive function to calculate the **sum of digits** of a non-negative integer. Use Result<u32, String> to handle input validation.

   fn sum_of_digits_checked(n: i32) -> Result<u32, String>

   Example:

   Input: 1234

   Output: 10

TA Check (result and error handling): ____

3. Create a Rust program that generates and displays Pascal's Triangle using recursive functions.

   Requirements:

   1) Prompt the user to enter a number 'n' between 1 and 9 (inclusive).
   2) Validate the input and re-prompt if the input is invalid.

   Pascal's Triangle Generation:

   1) Implement a recursive function 'pascal(row, col)' that calculates the value at any given position in Pascal's Triangle.
   2) The function should return 1 for the edges of the triangle and use recursion for inner values.

   Triangle Display:

   1) Create a function 'print_pascal_row(n, row)' that prints a single row of the triangle.
   2) Implement proper spacing for alignment of the triangle.
   3) Each number should be displayed with a width of 4 characters for readability.

   Main Program:

1) In the main function, get user input and display the result.

2) Include appropriate comments to explain your logic.

Constraints:

1) Use only the standard Rust library.

2) Do not use any additional data structures to store the triangle values.

Example Output:

For input n = 5, the output should look similar to this:

```
    1
   1 1
  1 2 1
 1 3 3 1
1 4 6 4 1
```

TA Comment: ____Finished Time: __________