COMP281 Assignment 1: Programming Solutions Report

Introduction

This report details my approach and thought process behind the solutions to the five distinct programming challenges presented in the COMP281 coursework. The assignment tasked creating C programs capable of handling various computational and data manipulation tasks, emphasising the importance of efficient, readable code and accurate output.

Problem 1: Area and Circumference of Circles

Task: Calculate the cumulative area and circumference of circles with radii incrementing from r1 to r2.
Strategy: I used a for-loop to iterate through each radius value within the specified range. I computed the area and circumference using the respective mathematical formulas within each iteration, combining these values. I ensured the final sums were displayed with three decimal places, adhering to the output precision requirement.

Problem 2: Count Characters in a String

Task: Analyse a string to count the occurrences of English letters, digits, spaces, and other characters.
Strategy: My approach involved reading the input character by character, categorising each based on ASCII value ranges, and tallying them in respective counters. I could also distinguish between the character types, ensuring an accurate count.

Problem 3: Reverse String

Task: Reverse the characters of an input string.
Strategy: After capturing the string, I determined its length to facilitate a two-pointer approach, where characters at symmetric positions from the string's ends are swapped until the centre is reached. This method was chosen for its simplicity and efficiency.

Problem 4: Precise Division

Task: Determine the n-th digit following the decimal in the quotient of a divided by b.
Strategy: This problem was more complex than initially thought, as I had to discard the integer part of the quotient and iteratively extract each subsequent decimal digit. By repeatedly multiplying the remainder by 10, I could isolate and print the desired digit.

Problem 5: Swap Array

Task: Identify the array element with the smallest absolute value and swap it with the final element in the array.
Strategy: I initiated a traversal of the integer array to locate the element's index with the minimum absolute value. A simple swap operation was then performed to reposition this element at the array's end.

Conclusion

Tackling these problems reinforced fundamental programming concepts and C syntax, honing my problem-solving and code optimisation skills. Each solution focused on clarity, efficiency, and adherence to the specified requirements, ensuring functionality across various test cases. This report includes the methodologies and rationale behind my solutions, reflecting the analytical and technical skills developed through this assignment.