

Homework 7

Rishabh Gosain

Work Done

Conditional statements, loops, vectors, recursion/iteration, and user-defined functions were among the programming concepts that were to be implemented in many tiny C++ programs for this project. To enable independent testing of each component, I coded all functions in an easy-to-read way.

- **Question 1: Switch Statement** I converted a MATLAB `switch` statement to a C++ `switch` block. After reading an integer, the software prints the category (Negative One, Zero, Positive One, or Other Value).
- **Question 2: Vector Printing** I wrote a `print_vector` function that outputs an integer vector's contents on a single line. In order to cope with both named vectors and transient return values, the function takes a vector by value.
- **Question 3: Fibonacci Values Below 4,000,000** I used a `while` loop to create the Fibonacci sequence. I prolonged the loop until the amount surpassed 4,000,000, which satisfies the assignment requirement, rather than publishing a set number of words.
- **Question 4.1: Checking for Primality** I implemented `isprime(int n)` by counting the number of divisors of n . If exactly two divisors exist, the number is prime. A small test function prints the results for three inputs as required.
- **Question 4.2: Factorization** I created `factorize(int n)`, which yields all of an integer's factors. I made sure that every factor pair was accurately gathered by looping up to \sqrt{n} .
- **Question 4.3: Prime Factorization** By constantly dividing the input by 2 and then by odd numbers up to \sqrt{n} , I developed an effective prime factorization method. All prime factors, even those that are repeated (e.g., $72 \rightarrow 2, 2, 2, 3, 3$), are returned by the function.

- **Question 5: Pascal’s Triangle** I generated the first n rows of Pascal’s triangle using iteration. Each row is built from the previous row using the well-known rule:

$$\text{row}[j] = \text{prev}[j - 1] + \text{prev}[j].$$

The function prints the triangle without requiring any combinatorial formulas.

Results

All functions were tested inside the `main()` program:

- The switch statement correctly printed the matching category for each tested input.
- The vector printing function output the expected sequence: 10 12 14 16 18.
- The Fibonacci loop printed all terms not exceeding 4,000,000.
- The primality tests returned accurate values for 2, 10, and 17.
- The factorization of 2, 72, and 196 matched the expected outputs.
- The prime factorization of those same values also matched the expected repeated-prime structure.
- The Pascal’s triangle function printed clean, correct rows for any positive input.

Discussion of Results

Overall, every element of the program operates as anticipated. Debugging was made easy by the clear code style, and clarity was guaranteed by breaking each query up into its own test function. The repetitive building of Pascal’s triangle showed that recursion was not required to meet the criteria, and the factorization and prime factorization reasoning worked correctly for all tested values. Concepts like vector use, loops, function design, and transferring ideas from MATLAB to C++ were all reinforced by the assignment.