Certainly! Here are some coding questions based on strings that you can use to practice your programming skills:

1. Reverse a String: Write a function that reverses a given string without using any built-in reverse functions or methods.

2. Anagram Detection: Create a function to check if two strings are anagrams of each other (i.e., they contain the same characters, but not necessarily in the same order).

3. Palindrome Checker: Write a program that checks if a given string is a palindrome (reads the same forwards and backward).

4. String Compression: Implement a function that performs basic string compression using the counts of repeated characters. For example, "aabcccccaaa" would become "a2b1c5a3."

5. String Rotation: Given two strings, write a function to check if one is a rotation of the other. For example, "waterbottle" is a rotation of "erbottlewat."

6. Longest Substring Without Repeating Characters: Find the length of the longest substring without repeating characters in a given string.

7. String to Integer (atoi): Implement a function that converts a string to an integer. Handle cases where the string may have leading whitespace, a plus or minus sign, and other non-digit characters.

8. String Matching: Write a function that finds all occurrences of a substring within a larger string.

9. String Permutations: Generate all permutations of a given string.

10. String Edit Distance (Levenshtein Distance): Write a function to calculate the minimum number of edit operations required to transform one string into another. Allowed operations are insertion, deletion, and substitution.

11. First Non-Repeating Character: Find the first non-repeating character in a string.

12. Longest Common Prefix: Find the longest common prefix among an array of strings.

13. Valid Parentheses: Check if a given string containing various parentheses characters (e.g., '()', '[]', '{}') is valid, i.e., all opening brackets are closed in the correct order.

14. String to ZigZag Conversion: Convert a given string to a zigzag pattern with a specified number of rows.

15. Word Break: Given a string and a dictionary of words, determine if the string can be segmented into a space-separated sequence of dictionary words.

16. Regular Expression Matching: Implement regular expression matching with support for '.' (matches any character) and '*' (matches zero or more of the preceding element).

These questions cover a range of string manipulation and text processing tasks and can help you improve your programming skills.