For learning programming basics and understanding time complexity, here are 50 practice questions that range from basic to intermediate level. These will help you understand loops, conditional statements, functions, and complexity analysis.

**Basics of Programming**

1. Write a program to print "Hello, World!".
2. Write a program to add two numbers entered by the user.
3. Write a program to check if a number is even or odd.
4. Write a program to find the largest of three numbers.
5. Write a program to calculate the factorial of a number using loops.
6. Write a program to find the sum of all numbers up to N.
7. Write a program to reverse a given number.
8. Write a program to check if a number is prime.
9. Write a program to generate the Fibonacci series up to N terms.
10. Write a program to check if a given year is a leap year.

**Functions**

1. Write a function to check if a number is palindrome.
2. Write a function to calculate the power of a number using recursion.
3. Write a function to find the GCD of two numbers.
4. Write a function to check if a string is a palindrome.
5. Write a function to calculate the sum of digits of a number.
6. Write a function to find the factorial of a number using recursion.
7. Write a function to count the number of vowels in a string.
8. Write a function to find the maximum and minimum elements in an array.
9. Write a function to find the second largest number in an array.
10. Write a function to calculate the n-th Fibonacci number using recursion.

**Conditional Statements & Loops**

1. Write a program to print all even numbers between 1 and 100.
2. Write a program to print a pattern like a right-angle triangle using '\*'.
3. Write a program to find the sum of all even numbers up to N.
4. Write a program to count the number of digits in a number.
5. Write a program to print the multiplication table of a given number.
6. Write a program to check if a given number is an Armstrong number.
7. Write a program to find the sum of squares of the first N natural numbers.
8. Write a program to swap two numbers without using a temporary variable.
9. Write a program to find the largest number in an array.
10. Write a program to print all prime numbers between 1 and 100.

**Arrays & Strings**

1. Write a program to reverse an array.
2. Write a program to find the frequency of each character in a string.
3. Write a program to remove duplicates from an array.
4. Write a program to sort an array in ascending order.
5. Write a program to merge two sorted arrays.
6. Write a program to find the common elements in two arrays.
7. Write a program to check if a string is a rotation of another string.
8. Write a program to find the longest common prefix in an array of strings.
9. Write a program to find the intersection of two arrays.
10. Write a program to rotate an array by k positions.

**Time Complexity Practice**

1. Write a program to find the time complexity of linear search.
2. Write a program to analyze the time complexity of bubble sort.
3. Write a program to analyze the time complexity of binary search.
4. Write a program to calculate the time complexity of finding the maximum element in an array.
5. Write a program to find the time complexity of merging two sorted arrays.
6. Write a program to analyze the time complexity of the Fibonacci sequence.
7. Write a program to calculate the space complexity of a recursive function.
8. Write a program to analyze the time complexity of quicksort.
9. Write a program to calculate the time complexity of a nested loop.
10. Write a program to determine the space complexity of a given array-based problem.

**Next Steps**

1. Try implementing each problem in the language you're learning (C++, Java, or Python).
2. Analyze the time complexity of each problem.
3. Focus on best, worst, and average-case scenarios for sorting and searching algorithms.

This practice should help you move from basic to intermediate understanding of programming and complexity analysis. Let me know if you'd like any specific solutions or further clarification on any topic!