Programming Practice Questions
Introduction
These programming questions are designed to test understanding of fundamental programming concepts including functions, loops, decision-making structures, and input/output operations. Each question should take approximately 2 hours to complete.

Extra Questions
If your number ends with an even digit, then answer question number 2. If it is divisible by 3, then answer question number 3. Otherwise, answer question number 1.

Question 1: Number Pattern Analysis
Problem Description
Create a program that reads a sequence of numbers from the user and performs various analyses on them. The program should continue reading numbers until the user enters -1.

Required Functions
1. `findLongestIncreasing()`: Find the longest sequence of increasing numbers
2. `countDigitFrequency()`: Count how many times a specific digit appears
3. `printReverse()`: Print the numbers in reverse order
4. `findTargetSum()`: Find pairs of numbers that sum to a target value

Example Interaction
```
Enter numbers (-1 to stop):
15
23
28
14
35
42
-1

Enter a target sum: 51

Results:
Longest increasing sequence length: 3
Most frequent digit: 2 (appears 3 times)
Numbers in reverse: 42 35 14 28 23 15
Pairs that sum to 51:
15 and 35
23 and 28
```

Requirements
- Use functions to organize code
- Implement input validation
- Handle edge cases (empty sequence, invalid inputs)
- Use arrays to store the numbers

Question 2: Word Transformer
Problem Description

Create a program that transforms words based on specific rules. The program should read a word and perform various transformations based on user choice.

Required Functions
1. `reverseWord()`: Reverse the input word
2. `countLetters()`: Count vowels and consonants
3. `replaceVowels()`: Replace all vowels with '*'
4. `checkPalindrome()`: Check if the word is a palindrome

Example Interaction
```
Enter a word: PROGRAMMING

Choose transformation:
1. Reverse word
2. Count vowels and consonants
3. Replace vowels
4. Check palindrome
Enter choice: 2

Results:
Vowels: 3
Consonants: 8

Do another transformation? (y/n): y
Enter choice: 3

Result: PR*GR*MM*NG
```

Requirements
- Implement proper input validation
- Handle both uppercase and lowercase letters
- Allow multiple transformations on the same word
- Display clear error messages for invalid inputs

Question 3: Number Series Generator
Problem Description
Create a program that generates different number series based on user input. The program should allow users to specify the length of the series and the type of series they want.

Required Functions
1. `generateFibonacci()`: Generate Fibonacci series
2. `generatePrimes()`: Generate prime numbers
3. `generateAlternating()`: Generate alternating positive and negative numbers
4. `isPrime()`: Check if a number is prime

Example Interaction
```
Enter series length: 8
Choose series type:
1. Fibonacci
```

2. Prime Numbers
3. Alternating Numbers
Enter choice: 1

Fibonacci Series: 1 1 2 3 5 8 13 21

Generate another series? (y/n): y
Enter series length: 5
Choose series type: 2

Prime Numbers: 2 3 5 7 11
```

### Requirements
- Validate input length (must be positive)
- Handle invalid menu choices
- Display error messages for invalid inputs
- Allow generating multiple series

## Evaluation Criteria
Students will be evaluated on:
1. Proper use of functions
2. Correct implementation of loops
3. Appropriate use of decision-making structures
4. Clear and informative input/output
5. Code organization and readability
6. Handling of edge cases and invalid inputs

## Submission Guidelines
1. Submit well-commented source code
2. Include test cases demonstrating program functionality
3. Document any assumptions made
4. Explain the approach used to solve each problem
```