Question 1:

A **Prime-Adam** integer is a positive integer (without leading zeros) which is a prime as well as an Adam number.

Prime number: A number which has only two factors, i.e. 1 and the number itself. Example: 2, 3, 5, 7 ... etc.

Adam number: The square of a number and the square of its reverse are reverse to each other.

```
Example: If n = 13 and reverse of 'n' = 31, then, (13)^2 = 169 (31)^2 = 961 which is reverse of 169 thus 13, is an Adam number.
```

Accept two positive integers m and n, where m is less than n as user input. Display all Prime-Adam integers that are in the range between m and n (both inclusive) and output them along with the frequency, in the format given below:

Test your program with the following data and some random data:

Example 1

INPUT:

m = 5 n = 100

OUTPUT:

THE PRIME-ADAM INTEGERS ARE:

11 13 31

FREQUENCY OF PRIME-ADAM INTEGERS IS: 3

Example 2

INPUT:

m = 100

n = 200

OUTPUT:

THE PRIME-ADAM INTEGERS ARE:

101 103 113

FREQUENCY OF PRIME-ADAM INTEGERS IS: 3

Example 3

INPUT:

m = 50

n = 70

OUTPUT:

THE PRIME-ADAM INTEGERS ARE:

NIL

FREQUENCY OF PRIME-ADAM INTEGERS IS: 0

Example 4

INPUT:

m = 700

n = 450

OUTPUT:

INVALID INPUT

Algorithm:

1. Input Validation:

Accept the inputs m and n. If m is greater than n, display
 "INVALID INPUT" and exit the program.

2. Prime Check:

- For every number i in the range from m to n (both inclusive), check if i is a prime number.
- A number is prime if it has only two factors: 1 and itself.
 You can check divisibility from 2 to the square root of i.

3. Reverse and Adam Check:

- o If i is prime, reverse the number i.
- Square both i and its reverse.
- If the square of the reverse of i is the reverse of the square of i, then the number is an Adam number.

4. Print Prime-Adam Integers:

- Print the found Prime-Adam numbers as the loop progresses.
- o Count the frequency of such numbers when one is found.

5. Print NIL case and frequency:

- \circ If there are no Prime-Adam integers (count = 0), output "NIL" and frequency 0.
- Else output the frequency (count).

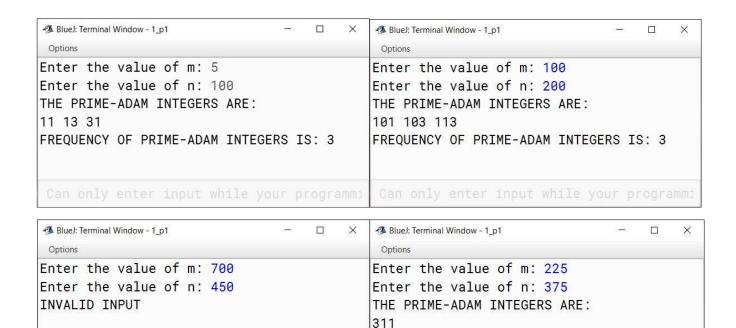
Variable Description Table:

Scope	Name	Data Type	Description
main	m	int	Starting range value input by
			the user
main	n	int	Ending range value input by
			the user
main	i	int	Loop variable to iterate from
			m to n
reverse	rev	int	Stores the reversed value of a
			number
main	count	int	Counts the number of Prime-
			Adam integers found

Solution:

```
import java.util.Scanner;
public class PrimeAdam {
  // Function to reverse a number
  public static int reverse(int num) {
    int rev = 0;
    while (num > 0) {
      rev = rev * 10 + num % 10;
      num /= 10;
    }
    return rev;
  // Function to check if a number is prime
  public static boolean isPrime(int num) {
    if (num < 2) return false;
    for (int i = 2; i <= Math.sqrt(num); i++) {</pre>
      if (num % i == 0) return false;
    }
    return true;
  // Function to check if a number is an Adam number
  public static boolean isAdam(int num) {
    int rev = reverse(num);
    return num * num == reverse(rev * rev);
  }
```

```
public static void main(String[] αrgs) {
    Scanner sc = new Scanner(System.in);
    // Input m and n
    System.out.print("Enter m: ");
    int m = sc.nextInt();
    System.out.print("Enter n: ");
    int n = sc.nextInt();
    sc.close();
   // Check for invalid input
    if (m > n) {
      System.out.println("INVALID INPUT");
      return;
    }
    System.out.println("THE PRIME-ADAM INTEGERS ARE:");
    int count = 0;
   // Loop through the range [m, n]
   for (int i = m; i <= n; i++) {
      // Check if number is prime and Adam and print it
      if (isPrime(i) && isAdam(i)) {
        count++;
        System.out.print(i + " ");
    }
   // Output the results
    if (count == 0) {
      System.out.println("NIL");
    System.out.println("\nFREQUENCY OF PRIME-ADAM INTEGERS IS: "
+ count);
  }
}
```



Can only enter input while your programmi | Can only enter input while your programmi

FREQUENCY OF PRIME-ADAM INTEGERS IS: 1