1. FACTORIAL

1. Inside the fact function:
2. Check if n is equal to 0:
3. If true, return 1 (the factorial of 0 is 1).
4. If false, continue to the next step.
5. Calculate the factorial recursively:
6. Calculate the factorial by calling the fact function with n as an argument and store the result in a variable f.
7. Print the result: "Factorial of " + n + ": " + f.

```
package demo4;

public class Fact {
	public static int fact(int n) {
		if(n==0)
			return 1;
			return n*fact(n-1);

	}


	public static void main(String[] args) {
		int n=5;
		int f=fact(n);
		System.out.println("Factorial of "+ n+":"+f);

	}
}
```

Output:

Factorial of 5:120

2. PRIME NUMBER

Algorithm:

1. If n is less than or equal to 2, it returns true if n is exactly 2 (as 2 is a prime number) or false otherwise.
2. If i is less than or equal to 1, it means no divisor other than 1 has been found, so it returns true, indicating that the number is prime.
3. If the current divisor i divides the number n evenly, it returns false, indicating that the number is not prime.
4. The main method calls isPrime with the number n and n/2 as the initial divisor since you don't need to check divisors greater than half of the number.

```
package demo4;

public class Prime {
        public static boolean isPrime(int n, int i) {
            if (n <= 2) {
                return (n == 2) ? true : false;
            }
            if (i <= 1) {
                return true;
            }
            if (n % i == 0) {
                return false;
            }
            return isPrime(n, i - 1);
         }
```

```
    public static void main(String[] args) {
        int n = 7;

        if (isPrime(n, n / 2)) {
            System.out.println(n + " is a prime number.");
        } else {
            System.out.println(n + " is not a prime number.");
        }
    }
}
```

Output:
7 is a prime number.

3. FIBONACCI SERIES

Alogithm:

1. Declare two static integer variables: firstT and secondT, both initialized to 0. These variables represent the first and second terms of the Fibonacci sequence.
2. Define a static method m1 that takes an integer n as an argument.
3. Inside the m1 method:
4. Check if n is greater than 0:
5. If true, calculate the next term in the Fibonacci sequence by adding firstT and secondT, and store it in a variable next.
6. Update firstT to be the value of secondT.
7. Update secondT to be the value of next.
8. Print the value of next followed by a space.
9. Recursively call m1(n - 1) to calculate and print the next term.

```
package demo4;
```

```
public class Fibonnacci {
   static int firstT = 0, secondT = 1;

   public static void m1(int n) {
      if (n > 0) {
         int next = firstT + secondT;
         firstT = secondT;
         secondT = next;
         System.out.print(next + " ");
         m1(n - 1);
      }
   }

   public static void main(String[] args) {
      int n = 10;
 System.out.print("Factorial of 10 terms:");

 System.out.print(firstT + " " + secondT + " ");
      m1(n - 2);
   }
}
```

Output:

Factorial of 10 terms:
0 1 1 2 3 5 8 13 21 34