```
public class 0_ Basics{
// Count Digits
class Solution {
    static int evenlyDivides(int N) {
        // code here
        int x = 0;
        int k = N;
        while (N != 0) {
            int d = N % 10;
            if (d!=0&&k % d == 0) {

                x++;
            }
            N = N / 10;
        }
        return x;
    }
}
//  Reverse a Number
class Solution {
    public int reverse(int x) {
        int rev = 0;
        while (x != 0) {
            int d = x % 10;
            if (rev >= Integer.MAX_VALUE / 10 || rev <= Integer.MIN_VALUE / 10) {
                return 0;
            }
            rev = rev * 10 + d;
            x = x / 10;
        }
        return rev;
    }
}
//  Check Palindrome
class Solution {
    public boolean isPalindrome(int y) {
        int rev = 0;
        int x = y;
        boolean sign = y >= 0 ? true : false;
        if (sign == false) {
            x = -x;
        }
        while (x != 0) {
            int d = x % 10;
            rev = rev * 10 + d;
            x = x / 10;
        }
        if (rev == y && sign == true) {
            return true;
        } else if (sign == false) {
            return false;
        }
        return false;
    }
}
//  GCD Or HCF
//  Armstrong Numbers
//  Print all Divisors
//  Check for Prime
// Understand recursion by print something
class Solution {

    public void printNos(int N) {
        // Your code here
        if (N == 0) {
            return;
        }
        printNos(N - 1);
        System.out.print(N + " ");
    }
}

// Print name N times using recursion
class Solution {

    void printGfg(int N) {
        // code here
        if (N == 0) {
            return;
        }
        System.out.print("GFG ");
        printGfg(N - 1);
    }
}

// Print 1 to N using recursion
class Solution {

    public void printNos(int N) {
        // Your code here
        if (N == 0) {
            return;
        }
        printNos(N - 1);
        System.out.print(N + " ");
    }
}
```

```
// Print N to 1 using recursion
class Solution {

    void printNos(int N) {
        // code here
        if (N == 0) {
            return;
        }
        System.out.print(N + " ");
        printNos(N - 1);
    }
}
// Sum of first N numbers

class Solution {
    long sumOfSeries(long N) {
        // code here
        if (N == 0) {
            return 0;
        }
        return N * N * N + sumOfSeries(N - 1);
    }
}

// Factorial of N numbers
class Solution {
    static ArrayList<Long> factorialNumbers(long N) {
        // code here
        ArrayList<Long> ls = new ArrayList<>();
        long ind = 1;
        while (true) {
            long fact = f(ind);
            if (fact <= N) {
                ls.add(fact);
                ind++;
            } else {
                break;
            }
        }
        return ls;
    }

    static long f(long a) {
        if (a == 1) {
            return 1;
        }
        return a * f(a - 1);
    }
}
```

```
// Reverse an array
// Check if a string is palindrome or …
class Solution {
    public boolean isPalindrome(String s) {
        String n = s.toLowerCase();
        int l = n.length();
        return isPal(n, 0, l - 1);
    }

    public boolean isPal(String st, int s, int e) {
        if (s >= e) {
            return true;
        }
        boolean start = checkIfAlphanumeric(st.charAt(s));
        boolean end = checkIfAlphanumeric(st.charAt(e));
        if (start == true && end == true) {
            return st.charAt(s) == st.charAt(e) && isPal(st, s + 1, e - 1);
        } else if (start == false && end == false) {
            return isPal(st, s + 1, e - 1);
        } else if (start == false && end == true) {
            return isPal(st, s + 1, e);
        } else {
            return isPal(st, s, e - 1);
        }
    }

    public boolean checkIfAlphanumeric(char character) {
        if ((character >= '0' & character <= '9') || (character >= 'a' && character <= 'z')
                || (character >= 'A' && character <= 'Z')) {
            return true;
        } else {
            return false;
        }
    }

}

// Fibonacci Number
class Solution {
    public int fib(int n) {
        if (n == 0 || n == 1) {
            return n;
        } else {
            return fib(n - 1) + fib(n - 2);
        }
    }
}}
```