```java
import java.math.BigInteger;

public class newMethods {
    static int x,y,d;
    //Returns the factorial of the number given (n)
    static int fac(int n)
    {
        if(n==0)return 1;
        if(n==1)return 1;
        else return n*fac(n-1);
    }

    //Returns the bumber n-choose-k for given n and k
    static long choose(int n, int k)
    {
        long temp = 1;
        if(n<k)temp=0;
        else {
            for(int i = 0; i<k ; i++){
                temp*=(n-i);
            }
            for(int i = 0; i<k ; i++){
                temp/=(i+1);
            }
        }
        return temp;
    }

    //Returns the sum of first n squares
    static long sq(int n)
    {
        return n*(n+1)*(2*n+1)/6;
    }

    //Returns the sum of first n cubes
    static long cu(int n)
    {
        return n*(n+1)*(n+1)*n/4;
    }

    //Returns the nth Fibonacci Sequence term
    static long fib(long n)
    {
        long[] fib = new long[(int)n+1];
        long fibo;
        if(n==0)
            return 0;
        if(n==1)
            return 1;
        if(fib[(int)n]!=0)
            return fib[(int)n];
        else
            fibo = fib(n-1) + fib(n-2);
        fib[(int)n] = fibo;
        return fibo;
    }
    static long nfib(long s)
    {
        double phi = (1.0+Math.sqrt(5))/2;
        double _phi = (1.0-Math.sqrt(5))/2;
        double n = (double)s+1;
        return (long)((Math.pow(phi, n)-Math.pow(_phi, n))/Math.sqrt(5));
    }

    //The Seive of Eratosthenes (Number of Primes less than n)
    static long seiveN(int n)
    {
        long tot = 0;
        long[] temp = new long[n+1];
        //mark all the numbers as prime
        for (int x = 0; x<n+1; x++)
        {
            temp[x] = x;
        }
        //remove all the non-primes from the list isPrime
        for(int i = 2; i<n+1; i++){
            if(temp[i]!=0){
                for(int j = 2; j*i<n+1; j++){
```

```java
//                  System.out.println(temp[j*i]);
                    temp[j*i]=0;
                }
            }
        }
        for (int i= 2 ; i<n+1; i++){
//          System.out.print(temp[i] + " ");
            if(temp[i]!=0){
                tot+=1;
//              System.out.print(temp[i] + " ");
            }
        }
        return tot;
    }

    //The Seive of Eratosthenes (List of All the primes)
        static long[] seiveP(int n){
            long[] temp = new long[n+1];
            long[] P = new long[n+1];
            for (int x = 0; x<n+1; x++){
                temp[x] = x;
            }
            //remove all the non-primes from the list isPrime
            for (int i = 2; i<n+1; i++){
                if(temp[i]!=0){
                    for(int j = 2; j*i<n+1; j++){
                        temp[j*i]=0;
                    }
                }
            }
            //Unneccesary but returns the final array eliminating zeroes
            for (int i= 2 ; i<n+1; i++){
                if(temp[i]!=0){
                    P[i]=temp[i];
                }
            }
            return P;
        }

```

```java
	static BigInteger palinBig(String s)
	{
		StringBuilder sb = new StringBuilder(2*s.length());
		sb.append(s);
		sb.reverse();
		String temp = sb.toString();
		sb.delete(0, sb.length());
		sb.append(s);
		sb.append(temp);
		return new BigInteger(sb.toString());
	}

	static int palin(int n)
	{
		Integer t = new Integer(n);
		int length = (int)Math.log10(n);
		StringBuilder sb = new StringBuilder(2*length);
		String s = t.toString();
		sb.append(s);
		sb.reverse();
		String temp = sb.toString();
		sb.delete(0, sb.length());
		sb.append(s);
		sb.append(temp);

		return Integer.parseInt(sb.toString());
	}
	// Finds the Greatest Common Divisor of two numbers : Very Useful
	static long gcd(long a, long b)
	{
		long in = Math.min(a, b);
		long ax = Math.max(a, b);
		return in==0 ? ax : gcd(in, ax%in);
	}
	//Finds the Least Common Multiple of two numbers, uses gcd(a,b) statically
	static long lcm(long a, long b){
		return a*b/gcd(a,b);
```

```java
        }
        // store x, y, and d as global variables
        static void extendedEuclid(int a, int b){
            if(b == 0){x = 1; y = 0; d = a; return;}
            extendedEuclid(b, a%b);
            int x1 = y;
            int y1 = x - (a/b)*y;
            x = x1;
            y = y1;
        }

        //Good Seive of Eratosthenes
        //The Seive of Eratosthenes (List of All the primes)
        static Long[] seiveE(int n){
//          long tot = 0;
            long[] temp = new long[n];
            ArrayList<Long> P = new ArrayList<Long>();
            //mark all the numbers as prime
            for (int x = 2; x<n+1; x++){
                temp[x-1] = x;
            }
//          System.out.println(Arrays.toString(temp));
            //remove all the non-primes from the list isPrime
            for (int i = 2; i<n; i++){
                if(temp[i-1]!=0){
                    for(int j = 2; j*i<n+1; j++){
//                      System.out.println(i);
                        temp[j*i-1]=0;
//                      System.out.println(Arrays.toString(temp));
                    }
                }
            }
            for (int i= 0 ; i<n; i++){
//              System.out.print(temp[i] + " ");
                if(temp[i]!=0){
//                  tot+=1;
                    P.add(temp[i]);
//                  System.out.print(temp[i] + " ");
                }
            }
            Long Primes[] = P.toArray(new Long[P.size()]);
            return Primes;
        }
}
```