

newMethods.java

```
1 import java.math.BigInteger;
2
3
4 public class newMethods {
5     static int x,y,d;
6     //Returns the factorial of the number given (n)
7     static int fac(int n)
8     {
9         if(n==0) return 1;
10        if(n==1) return 1;
11        else return n*fac(n-1);
12    }
13
14    //Returns the bumber n-choose-k for given n and k
15    static long choose(int n, int k)
16    {
17        long temp = 1;
18        if(n<k) temp=0;
19        else {
20            for(int i = 0; i<k ; i++){
21                temp*=(n-i);
22            }
23            for(int i = 0; i<k ; i++){
24                temp/=(i+1);
25            }
26        }
27        return temp;
28    }
29
30    //Returns the sum of first n squares
31    static long sq(int n)
32    {
33        return n*(n+1)*(2*n+1)/6;
34    }
35
36    //Returns the sum of first n cubes
37    static long cu(int n)
38    {
39        return n*(n+1)*(n+1)*n/4;
40    }
41
```

```

42  //Returns the nth Fibonacci Sequence term
43  static long fib(long n)
44  {
45      long[] fib = new long[(int)n+1];
46      long fibo;
47      if(n==0)
48          return 0;
49      if(n==1)
50          return 1;
51      if(fib[(int)n] != 0)
52          return fib[(int)n];
53      else
54          fibo = fib(n-1) + fib(n-2);
55      fib[(int)n] = fibo;
56      return fibo;
57  }
58  static long nfib(long s)
59  {
60      double phi = (1.0+Math.sqrt(5))/2;
61      double _phi = (1.0-Math.sqrt(5))/2;
62      double n = (double)s+1;
63      return (long) ((Math.pow(phi, n)-Math.pow(_phi,
64  n))/Math.sqrt(5));
65
66  //The Seive of Eratosthenes (Number of Primes less than
67  n)
68  static long seiveN(int n)
69  {
70      long tot = 0;
71      long[] temp = new long[n+1];
72      //mark all the numbers as prime
73      for (int x = 0; x<n+1; x++)
74      {
75          temp[x] = x;
76      }
77      //remove all the non-primes from the list isPrime
78      for(int i = 2; i<n+1; i++){
79          if(temp[i] != 0){
80              for(int j = 2; j*i<n+1; j++){

```

newMethods.java

```
80 //                System.out.println(temp[j*i]);
81                temp[j*i]=0;
82            }
83        }
84    }
85    for (int i= 2 ; i<n+1; i++){
86 //        System.out.print(temp[i] + " ");
87        if(temp[i]!=0){
88            tot+=1;
89 //        System.out.print(temp[i] + " ");
90        }
91    }
92    return tot;
93 }
94
95 //The Seive of Eratosthenes (List of All the primes)
96 static long[] seiveP(int n){
97     long[] temp = new long[n+1];
98     long[] P = new long[n+1];
99     for (int x = 0; x<n+1; x++){
100         temp[x] = x;
101     }
102     //remove all the non-primes from the list isPrime
103     for (int i = 2; i<n+1; i++){
104         if(temp[i]!=0){
105             for(int j = 2; j*i<n+1; j++){
106                 temp[j*i]=0;
107             }
108         }
109     }
110     //Unneccesary but returns the final array
    eliminating zeroes
111     for (int i= 2 ; i<n+1; i++){
112         if(temp[i]!=0){
113             P[i]=temp[i];
114         }
115     }
116     return P;
117 }
118
```

```

119         static BigInteger palinBig(String s)
120         {
121             StringBuilder sb = new
122             StringBuilder(2*s.length());
123             sb.append(s);
124             sb.reverse();
125             String temp = sb.toString();
126             sb.delete(0, sb.length());
127             sb.append(s);
128             sb.append(temp);
129             return new BigInteger(sb.toString());
130         }
131
132         static int palin(int n)
133         {
134             Integer t = new Integer(n);
135             int length = (int)Math.log10(n);
136             StringBuilder sb = new StringBuilder(2*length);
137             String s = t.toString();
138             sb.append(s);
139             sb.reverse();
140             String temp = sb.toString();
141             sb.delete(0, sb.length());
142             sb.append(s);
143             sb.append(temp);
144
145             return Integer.parseInt(sb.toString());
146         }
147         // Finds the Greatest Common Divisor of two numbers :
148         Very Useful
149         static long gcd(long a, long b)
150         {
151             long in = Math.min(a, b);
152             long ax = Math.max(a, b);
153             return in==0 ? ax : gcd(in, ax%in);
154         }
155         //Finds the Least Common Multiple of two numbers,
156         uses gcd(a,b) statically
157         static long lcm(long a, long b){
158             return a*b/gcd(a,b);

```

```

156     }
157     // store x, y, and d as global variables
158     static void extendedEuclid(int a, int b){
159         if(b == 0){x = 1; y = 0; d = a; return;}
160         extendedEuclid(b, a%b);
161         int x1 = y;
162         int y1 = x - (a/b)*y;
163         x = x1;
164         y = y1;
165     }
166
167     //Good Seive of Eratosthenes
168     //The Seive of Eratosthenes (List of All the primes)
169     static Long[] seiveE(int n){
170 //         long tot = 0;
171         long[] temp = new long[n];
172         ArrayList<Long> P = new ArrayList<Long>();
173         //mark all the numbers as prime
174         for (int x = 2; x<n+1; x++){
175             temp[x-1] = x;
176         }
177 //         System.out.println(Arrays.toString(temp));
178         //remove all the non-primes from the list isPrime
179         for (int i = 2; i<n; i++){
180             if(temp[i-1]!=0){
181                 for(int j = 2; j*i<n+1; j++){
182 //                     System.out.println(i);
183                     temp[j*i-1]=0;
184 //                     System.out.println(Arrays.toString(temp));
185                 }
186             }
187         }
188         for (int i= 0 ; i<n; i++){
189 //             System.out.print(temp[i] + " ");
190             if(temp[i]!=0){
191 //                 tot+=1;
192                 P.add(temp[i]);
193 //                 System.out.print(temp[i] + " ");
194             }

```

newMethods.java

```
195         }
196         Long Primes[] = P.toArray(new Long[P.size()]);
197         return Primes;
198     }
199 }
200
```