IMPORTANT NUMERICAL PROBLEMS

1)PRIME NUMBER AND USING RECURSION METHOD

```
import java.util.Scanner;
class Main {
   public static void main(String[] args) {
      Scanner scn=new Scanner(System.in);
      System.out.println("enter a number");
      int n=scn.nextInt();
      boolean flag=true;
      for(int i=2;i<=n/2;i++){
         if(n%i==0){
            flag=false;
            break;
         }
      }
      if(flag && n>=2){
         System.out.println("its a prime number");
      }
      else{
         System.out.println("its a not prime number");
      }
   }
```

ANIL

RECURSION METHOD

```
import java.util.Scanner;
class Prime {
   public static boolean isprime(int n,int i){
      if(i==1)return true;
      if(n<=1||n%i==0)return false;
      return isprime(n,i-1);
   }
```

```
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("enter a number");
        int n=scn.nextInt();
        boolean res=isprime(n,n/2);
        System.out.println(res);
          if(res){
           System.out.println("its prime a number");
        }else{
           System.out.println("its not prime number");
        }



    }
```

ANIL

2)PREFECT NUMBER AND USING RECURSION METHOD

EXAMPLE 6=DIVISIBLE BY 1,2,3

6=1+2+3

28=1+2+4+7+14

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("enter a number");
        int n=scn.nextInt();
```

```
        int sum=0;
        for(int i=1;i<=n/2;i++){
            if(n%i==0){
           sum+=i;
        }
       }
    if(sum==n){
        System.out.println("it s a perfect number");
    }
    else{
        System.out.println("its not a prefect number");
    }
}
 }
```

RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static int sumofD(int n,int i){
        if(i==1)
        return 1;
        if(n%i==0){
            return i+sumofD(n,i-1);
        }
        else{
            return sumofD(n,i-1);

        }

    }
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("enter a number");
```

ANIL

```
        int n=scn.nextInt();
        int res =sumofD(n,n/2);

        if(res==n){
            System.out.println("is a Perfect Number");
        }else{
             System.out.println(" is NOT a Perfect Number");
        }
    }
}
```

3)AUTOMORPHIC NUMBER AND USING RECURSION METHOD

6=36

5=25

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter a Number");
        int n=scn.nextInt();
        int s=n*n;
        boolean flag=true;
        while(n>0){
            if((n%10)!=(s%10)){
                flag=false;
                break;
            }
            n/=10;
            s/=10;

        }
        if(flag){
```

ANIL

```
        System.out.println("AUTOMORPHIC NUMBER");

        }else{
            System.out.println("it is not automorphic number");
        }

    }
```

RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static boolean isAutomorphic(int n,int s){
        if(n==0)return true;
        if(n%10 != s%10){
            return false;

        }
        return isAutomorphic(n/=10,s/=10);

    }
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("enter a number");
        int n=scn.nextInt();
        int s=n*n;
        boolean flag=isAutomorphic(n,s);
        if(flag){
            System.out.println(n + " is an Automorphic Number");
        }else{
            System.out.println(n + " is NOT an Automorphic Number");
        }
```

ANIL

```
    }
}
```

4)STRONG NUMBER AND USING RECURSION METHOD

145=5!+4!+1!=145

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("enter a number");
        int n=scn.nextInt();
       int m=n;
        int sum=0;
        while(n>0){
        int d=n%10;
        int prod=1;
             for(int i=1;i<=d;i++){
            prod*=i;

             }
             sum+=prod;
             n/=10;


        }
        if(sum==m){
            System.out.println("it's strong number");
        }else{
            System.out.println("it's not strong number");

        }
```

ANIL

```
    }
}
```

RECURSION METHOD

```
class Main {
    public static int factorial(int n){
        if(n==0||n==1)return 1;
        return n*factorial(n-1);
    }
    public static int sumofFac(int n){
        if(n==0)return 0;
        int digit=n%10;
        return factorial(digit)+sumofFac(n/10);
    }
    public static void main(String[] args) {
        int n=145;
        int sum=sumofFac(n);
        if(sum==n){
            System.out.println("is a strong number");
        }else{
            System.out.println("is not a strong number");
        }
    }
}
```

ANIL

5)ARM STRONG NUMBER AND USING RECURSION METHOD

153 length=3

1pow(3)+5pow(3)+3pow(3)=1+125+27=153

```
import java.util.Scanner;
class Main {
```

```
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("enter a number");
        int n=scn.nextInt();
        int temp=n;
        int l=0;
        int sum=0;
        while(n>0){
            l++;
            n/=10;

        }
       n=temp;
       while(n>0){
           int p=1;
           int rem=n%10;
           for(int i=1;i<=l;i++){
               p*=rem;
           }
           sum+=p;
           n/=10;

       }
       if(sum==temp){
           System.out.println("its a Arm Strong Number");
       }else{
           System.out.println("its not a Arm Strong Number");
       }

    }
}
```

ANIL

RECURSION METHOD

```
import java.util.Scanner;
class Main {
   public static int countdigits(int n){
      if(n==0)return 0;
      return 1+countdigits(n/10);

   }
   public static int armstrong(int n,int power){
      if(n==0)return 0;
      int digit=n%10;
      return (int)Math.pow(digit,power)+armstrong(n/10,power);
   }
   public static void main(String[] args) {
      Scanner scn=new Scanner(System.in);
      System.out.println("enter a number");
      int n=scn.nextInt();
      int power=countdigits(n);
      int sum=armstrong(n,power);
       if(n==sum){
      System.out.println(n + " is an Armstrong Number");
   }else{
      System.out.println(n+"is NOt an Arm Strong number");
   }

   }

}
```

ANIL

armstrongSum(153, 3)
→ 3^3 + armstrongSum(15, 3)
→ 27 + 5^3 + armstrongSum(1, 3)
→ 27 + 125 + 1^3 + armstrongSum(0, 3)

→ 27 + 125 + 1 + 0
→ 153

6)Drasium Number NUMBER AND USING RECURSION METHOD

$89 \rightarrow 8^1 + 9^2 = 8 + 81 = 89$

$135 \rightarrow 1^1 + 3^2 + 5^3 = 1 + 9 + 125 = 135$

```
import java.util.Scanner;
class Main {
   public static void main(String[] args) {
      Scanner scn=new Scanner(System.in);
      System.out.println("Enter a Number");
      int n=scn.nextInt();
      int temp=n;
      int l=0;
      int sum=0;
      while(n>0){
         l++;
         n/=10;
      }
      n=temp;
      while(n>0){
         int p=1;
          int rem=n%10;
         for(int i=1;i<=l;i++){
            p*=rem;

         }
         sum+=p;
```

ANIL

```
            n/=10;
            l--;

        }
        n=temp;
        if(temp==sum){
             System.out.println(n + " is a Disarium Number");
        } else {
            System.out.println(n + " is NOT a Disarium Number");
        }



    }
}
```

RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static int countdigit(int n){
        if(n==0)return 0;
        return 1+countdigit(n/10);
    }
    public static int Drasium(int n,int pow){
        if(n==0)return 0;
        int d=n%10
;        return (int)Math.pow(d,pow)+Drasium(n/10,pow-1);
    }
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter a Number");
        int n=scn.nextInt();
        int temp=n;
```

ANIL

```
        int pow=countdigit(n);
        int sum=Drasium(n,pow);

        if(sum==temp){
            System.out.println(n + " is a Disarium Number");
        } else {
            System.out.println(n + " is NOT a Disarium Number");
        }
    }
}
```

7)Neon Number NUMBER AND USING RECURSION METHOD

Take the number **9**:

- Square of 9 = 81
- Sum of digits = 8 + 1 = **9** (which is equal to the original number)

So, **9 is a Neon Number**.

```
import java.util.Scanner;

class Main {

    public static void main(String[] args) {

        Scanner scn=new Scanner(System.in);

        System.out.println("Enter a Number");
```

ANIL

```
        int n=scn.nextInt();
        int temp=n;
        int square=n*n;
        int sum=0;
        while(square>0){
            int rem=square%10;
            sum+=rem;
            square/=10;
        }
        if(sum==temp){
            System.out.println(n + " is  a Neon Number.");
        }
        else{
            System.out.println(n + " is NOT a Neon Number.");
        }
    }
}
```

ANIL

RECURSION METHOD

```
import java.util.Scanner;
class Main {
   public static int Neon(int s){
      if(s==0)return 0;
      int d=s%10;
      return d+Neon(s/10);
   }
   public static void main(String[] args) {
      Scanner scn=new Scanner(System.in);
      System.out.println("Enter a Number");
      int n=scn.nextInt();
      int square=n*n;
      int sum=Neon(square);
      if(sum==n){
         System.out.println("its a Neon Number");
      }else{
         System.out.println("its not  Neon Number");

      }
}}
```

ANIL

8) fibonacci NUMBER AND USING RECURSION METHOD

0, 1, 1, 2, 3, 5, 8, 13, …

```
import java.util.Scanner;

class Main {

   public static void main(String[] args) {
```

```
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter a Number");
        int n=scn.nextInt();
        int a=0;
        int b=1;
        int c;
        for(int i=1;i<=n;i++){
            System.out.println(a);
            c=a+b;
            a=b;
            b=c;

        }

    }
}
```

ANIL

RECURSION METHOD

```
import java.util.Scanner;
class Main {
   public static int fib(int n){
      if(n==0)return 0;
      else if(n==1)return 1;
      else return fib(n-1)+fib(n-2);
   }
   public static void main(String[] args) {
      Scanner scn=new Scanner(System.in);
      System.out.println("enter a number");
      int n=scn.nextInt();

      for(int i=0;i<=n;i++){
         System.out.println(fib(i));
      }


   }
}
```

ANIL

9) **Binary number to Decimal number**. AND USING RECURSION METHOD

Binary: 1010 → Decimal: 10

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter a number");
        int n=scn.nextInt();
        int bin=n;
        int dec=0;
```

```
        int p=1;
       while(bin>0){

           int bit=bin%10;
           dec=dec+(p*bit);
             bin/=10;
           p*=2;

       }

       System.out.println(dec);
    }
}
```

RECURSION METHOD

```
import java.util.Scanner;
class Main {
   public static int binary(int n){
      if(n==0)return 0;
      int d=n%10;
      return d+2*binary(n/10);
   }
   public static void main(String[] args) {
      Scanner scn=new Scanner(System.in);
      System.out.println("Enter a binary Number");
      int n=scn.nextInt();
      int decimal=binary(n);
      System.out.println(decimal);
```

ANIL

```
    }
}
```

10) **Decimal number to Binary number**. AND USING RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("enter a number");
        int dec=scn.nextInt();
        int p=1;
        int bin=0;
        while(dec>0){
            int d=dec%2;
            bin=(d*p)+bin;
            dec=dec/2;
            p=p*10;

        }

        System.out.println(bin);
    }
}
```

ANIL

RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static int dectobin(int n,int sum,int i){
        if(n==0)return sum;
        sum+=(n%2)*i;
        return dectobin(n/2,sum,i*10);
```

```
    }
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter a number");
        int n=scn.nextInt();
        int res=dectobin(n,0,1);
        System.out.println("Binary of " + n + " is: " + res);


    }
}
```

10) **LCM**. AND USING RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter a num1");
            int n1=scn.nextInt();
        System.out.println("Enter a num2");

        int n2=scn.nextInt();
        int gcd=1;
        for(int i=1;i<=n1 &&i<=n2;i++)
            if(n1%i==0 && n2%i==0)gcd=i;
            System.out.println("gcd is "+gcd);
            int lcm=(n1*n2)/gcd;
            System.out.println(lcm);
```

ANIL

```
    }
}
```

RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static int gcd(int n1,int n2){
        if(n2==0)return n1;
        return gcd(n2,n1%n2);
    }
    public static int lcm(int n1,int n2){
        return(n1*n2)/gcd(n1,n2);

    }
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter a number n1");
        int n1=scn.nextInt();
        System.out.println("Enter a number n2");
        int n2=scn.nextInt();
        int res=lcm(n1,n2);
        System.out.println("gcd is="+gcd(n1,n2));
        System.out.println("Lcm is="+res);

    }
}
```

ANIL

11)Sum of Digits AND USING RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
```

```
        System.out.println("enter a number");
        int n=scn.nextInt();
        int sum=0;
        while(n>0){
            int d=n%10;
            sum+=d;
            n/=10;
            if(n==0 && sum>9){
                System.out.println(sum);
                 n=sum;
                 sum=0;

            }
           }
        System.out.println(sum);


    }
}
```

ANIL

RECURSION METHOD

```
import java.util.Scanner;
class Main {
    public static int sumofDigits(int n){
        if(n==0)return 0;
        return (n%10) + sumofDigits(n/10);
    }
    public static int digitRoot(int n){
        if(n<10) return n;
        return digitRoot(sumofDigits(n));
    }
    public static void main(String[] args) {
```

```
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter a number");
        int n=scn.nextInt();
        int result=digitRoot(n);

        System.out.println(sumofDigits(n));
        System.out.println(result);
    }
}
```

Patterns

```
  *
 ***
*****
*******
*****
 ***
  *
```

ANIL

```
class Main {
    public static void main(String[] args) {
        int n = 7;
        int star = 1;
        int spaces = n / 2;

        for (int i = 1; i <= n; i++) {
            // Print spaces
            for (int j = 1; j <= spaces; j++) {
                System.out.print(" ");
            }
            // Print stars
```

```
            for (int k = 1; k <= star; k++) {
                System.out.print("*");
            }

            // Move to next line
            System.out.println();

            // Update spaces and stars for next row
            if (i <= n / 2) {
                spaces--;
                star += 2;
            } else {
                spaces++;
                star -= 2;
            }
        }
    }
}
```



```
  *
 * *
*   *
*    *
*   *
 * *
  *
public class HollowDiamond {
    public static void main(String[] args) {
        int n = 7; // should be odd
        int spaces = n / 2;
        int stars = 1;
```

```
        for (int i = 1; i <= n; i++) {
            // Print leading spaces
            for (int j = 1; j <= spaces; j++) {
                System.out.print(" ");
            }

            // Print stars with space in between for hollow effect
            for (int k = 1; k <= stars; k++) {
                if (k == 1 || k == stars)
                    System.out.print("*");
                else
                    System.out.print(" ");
            }

            // Move to next line
            System.out.println();

            // Adjust spaces and stars
            if (i < n / 2 + 1) {
                spaces--;
                stars += 2;
            } else {
                spaces++;
                stars -= 2;
            }
        }
    }
}
```

ANIL

```
 *
***
```

```
  *****
 *******
*********

class Main {
    public static void main(String[] args) {
        int n=5;int space=n-1;
        int star=1;
        for(int i=1;i<=n;i++){
            for(int j=1;j<=space;j++){
                System.out.print(" ");
            }
            for(int k=1;k<=star;k++){
               System.out.print("*");

            }
            System.out.println();
            space--;
            star+=2;

        }

    }
}
    *
   * *
  *   *
 *     *
*********


class Main {
```

ANIL

```
    public static void main(String[] args) {
        int n=5;int space=n-1;
        int star=1;
        for(int i=1;i<=n;i++){
            for(int j=1;j<=space;j++){
                System.out.print(" ");
            }
            for(int k=1;k<=star;k++){
                if(k==1||k==star||i==n){
                    System.out.print("*");
                }else{
                    System.out.print(" ");
                }

            }
            System.out.println();
            space--;
            star+=2;

        }

    }
}
*************
 ***********
  *********
   *******
    *****
     ***
      *
public class Main {
    public static void main(String[] args) {
```

ANIL

```
    int n = 7; // number of rows

    for (int i = 0; i < n; i++) {
        // Print leading spaces
        for (int j = 0; j < i; j++) {
            System.out.print(" ");
        }

        // Print stars
        for (int k = 0; k < 2 * (n - i) - 1; k++) {
            System.out.print("*");
        }

        // Move to next line
        System.out.println();
    }


  }
}
```

ANIL

```
*********
 *    *
  *   *
   * *
    *
```

3)PRIME NUMBER given range print alternate numbers

```
import java.util.Scanner;
class Main {
   public static void main(String[] args) {
      Scanner scn=new Scanner(System.in);
      System.out.println("Enter start number");
      int start=scn.nextInt();
      System.out.println("Enter end number");
      int end=scn.nextInt();

      int count=0;
      for(int i=start;i<=end;i++){
         if(i<=1)continue;
           boolean flag=true;
         for(int j=2;j<=i/2;j++){
            if(i%j==0){
               flag=false;
               break;

            }
         }
         if(flag){


            count++;
            if(count%2==1){
                 System.out.println(i+" ");


            }
         }
```

ANIL

```
        }

    }
}
```

3)PRIME NUMBER given range

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        System.out.println("enter a start number");
        int start=scn.nextInt();
        System.out.println("enter a end number");
        int end=scn.nextInt();

        for(int i=start;i<=end;i++){
            boolean flag=true;
            if(i<=2)continue;
            for(int j=2;j<=i/2;j++){
                if(i%j==0){
                    flag=false;

                }
            }
            if(flag){
                System.out.println(i);
            }
        }

    }
}
```

ANIL

Any Doubt Dm this mail :anils.pvg8095@gmail.com