↳ Product of ele. except itself

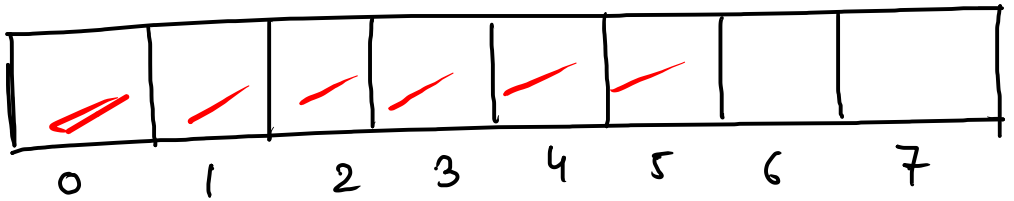
↳ Count smaller no.

↳ Array concept

↳ Number theory

↳ calculator 36

⇒ Arrays



⇒ Number Theory

$n = 12345$

$\rightarrow \begin{cases} n = n\%10; \\ n = n/10; \end{cases}$

$\begin{cases} 345 \\ 45 \end{cases}$

$\begin{cases} n = n\%1000; \\ n = n/100; \end{cases}$

Ques Palindrome (n = 12321)

dry run n = 12321 ans = 0

1) 12321 > 0 , rem = 12321 % 10 = 1 , ans = 0 * 10 + 1 = 1
   n = 12321 / 10 = 1232

2) 1232 > 0 , rem = 1232 % 10 = 2 , ans = 1 * 10 + 2 = 12
   1232 / 10 = 123

3) 123 > 0 , rem = 123 % 10 = 3 , ans = 12 * 10 + 3
   n = 123 / 10 = 12 = 123

4) 12 > 0 , rem = 12 % 10 = 2 , ans = 123 * 10 + 2
   n = 12 / 10 = 1 = 1232

5) 1 > 0 , rem = 1 % 10 = 1 , ans = 1232 * 10 + 1
   n = 1 / 10 = 0 = 12321

rem

rem

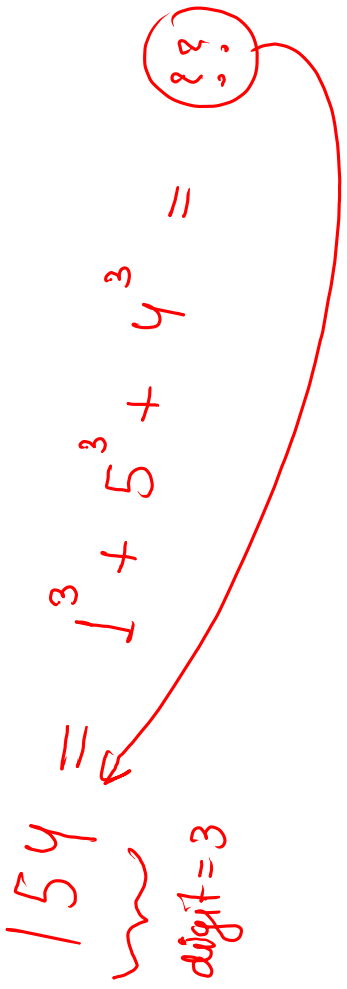
if(ans == num) P
else NP

n = 0

code

```
n = 12321;  temp = n;
int ans = 0
while( temp > 0 ){
    int rem = temp % 10;
    temp = temp/10;
    ans = ans * 10 + rem;
}
if( ans == n ){
    Syso( P );
} else{
    Syso( NP );
}
```

Ques Check for armstrong no. if it is of any size




```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int t = scn.nextInt();
    while (t-- > 0) {
        int n = scn.nextInt();
        System.out.println(armstrongNo(n));
    }
}

public static boolean armstrongNo(int num) {
    // int digit = 0;
    // int temp1 = num;
    // while (temp1 > 0) {
    //     temp1 = temp1 / 10;
    //     digit++;
    // }
    int digit = 3;
    int ans = 0;
    int temp = num;
    while (num > 0) {
        int rem = num % 10;
        num = num / 10;
        ans = ans + (int)Math.pow(rem, digit);
    }
    if (temp == ans) {
        return true;
    } else {
        return false;
    }
    // return (temp == ans) ? true : false;
}
```

HW_Counting Smaller Numbers in an Array

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    solve(arr, n);
}

public static void solve(int[] arr, int n) {
    for (int i = 0; i < n; i++) {
        int count = 0;
        for (int j = 0; j < n; j++) {
            if ( i != j && arr[i] > arr[j] ) {
                count++;
            }
        }
        System.out.print(count + " ");
    }
}
```

Permutation without Repetation

HW_calculator 36

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    int x = scn.nextInt();

    int[] ans = addOne(arr, n, x);
    for (int i = 0; i < ans.length; i++) {
        System.out.print(ans[i] + " ");
    }
}

// main logic
public static int[] addOne(int[] a, int n, int s) {
    for(int i=n-1;i>=0;i--){
        int ans=a[i]+s;
        a[i]=ans%10;
        s=ans/10;
    }
    if(s == 1){
        int[] ans = new int[n + 1];
        ans[0] = 1;
        for (int i = n - 1; i >= 0; i--) {
            ans[i + 1] = a[i];
        }
        return ans;
    }
    return a;
}
```