Note:- int x = 5, y = 6;

sumOf2Numbers(x, y); // calling

public static void sumOf2Numbers(int a, int b){

// statement

}

x == a
y == b

Note:- variables in f^n calling & f^n declaration are different with same value

Note:- only thing matters is f^n name & number of parameters should be same.

Sum of all Elements of Array

$n = 5$

arr =

7	-3	4	8	-2
0	1	2	3	4

ans = 14

ans = 0

psudo code

1) input array
2) declare ans as 0
3) traverse from 0 to (n-1) in array
   3.1) add each element in ans variable
4) return ans;

Code

```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 ans = sumOfArray(arr, n);
    System.out.println(ans);
}

public static int sumOfArray(int[] arr, int n) {
    int ans = 0;
    for (int i = 0; i < n; i++) {
        ans += arr[i];
    }
    return ans;
}
```

Maximum of Array

$n = 5$

arr =

7	-3	4	8	-2
0	1	2	3	4

ans = 8

ans = ~~$-\infty$~~ ~~7~~ 8

if (arr[i] > ans)

$i = 0$, $(7 > -\infty)$

$i = 1$, $(-3 > 7)$

$i = 2$, $(4 > 7)$

$i = 3$, $(8 > 7)$

$i = 4$, $(-2 > 8)$ false

psudo code

1) input array
2) declare ans as $-\infty$
3) traverse from 0 to (n-1) in array
   3.1) check (current ele. > ans)
      3.1.1) ans = current ele.
4) return ans;

Code

```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();
    }

    System.out.println(maximumOfArray(arr, n));
}
public static int maximumOfArray(int[] arr, int n) {
    int ans = Integer.MIN_VALUE;
    for (int i = 0; i < n; i++) {
        if ( arr[i] > ans ) {
            ans = arr[i];
        }
    }
    return ans;
}
```

GKSTR35 Count_Even

$n = 9$

arr =

5	2	3	4	7	8	10	9	1
0	1	2	3	4	5	6	7	8

count = 4

count = ~~0~~ ~~1~~ ~~2~~ ~~3~~ 4

pSudo code

1) input array
2) declare count = 0;
3) traverse from 0 to (n-1)
   3.1) check if current element is even
      3.1.1) count ++;
4) return count

Code

```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();
    }

    System.out.println(countEven(arr, n));
}
public static int countEven(int[] arr, int n) {
    int count = 0;
    for (int i = 0; i < n; i++) {
        if ( arr[i] % 2 == 0 ) {
            count++;
        }
    }
    return count;
}
```

Product of Elements Except Itself

$n = 4$

arr =

0	1	2	3
2	0	3	2

prod = $2 * 5 * 3 * 2$

ans =

0	1	2	3
30	12	20	30

$5 * 3 * 2$

$2 * 3 * 2$

$2 * 5 * 2$

$2 * 5 * 3$

(discarded)

$ans[0] = \frac{prod}{2}$

$ans[1] = \frac{prod}{0}$

$ans[2] = \frac{prod}{3}$

$ans[3] = \frac{prod}{2}$

Product of Elements Except Itself

(V. Imp)

$n = 4$

arr =

0	1	2	3
2	0	3	2

$i = 0$, $j = 0$ to $(n-1)$ — if(i != j) ans = ans * arr[i];

$i = 1$, $j = 0$ to $(n-1)$ — if(i != j) ans = ans * arr[i];

$i = 2$ $j = 0$ to $(n-1)$ — if(i != j) ans = ans * arr[i];

$i = 3$, $j = 0$ to $(n-1)$ — if(i != j) ans = ans * arr[i];

ans =

0	1	2	3
30	12	20	30

$5 * 3 * 2$

$2 * 3 * 2$

$2 * 5 * 2$

$2 * 5 * 3$