Note:-

if a function is of return type, then it must return something all the time

Maximum of Array

$n = 6$

arr =

-2	3	1	4	2	1
0	1	2	3	4	5

ans :- 4

Note :-

Integer.MIN_VALUE;

$-\infty$

Integer.MAX_VALUE;

$+\infty$

psudo code

```
1) create a variable max = $-\infty$
2) traverse in array from start to end
    2.1) check if current value > max
        2.1.1) max = current value
3) return max
```

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

```java
public static int maxOfArray(int[] arr) {
    int max = Integer.MIN_VALUE;
    int n = arr.length;
    for (int i = 0; i < n; i++) {
        max = Math.max( max, arr[i] );
    }
    return max;
}
```

$n = 6$

arr =

arr =	-2	3	1	4	2	1
	0	1	2	3	4	5

max = ~~$-\infty$~~ ~~-2~~ ~~3~~ 4

$n = 6$

$i=0, (-2 > -\infty)$ true

$i=1, (3 > -2)$ true

$i=2, (1 > 3)$ false

$i=3, (4 > 3)$ true

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

$i=5, (1 > 4)$ false

$i=6$

return 4

Product of Elements Except Itself (M. Imp)

$n = 4$

arr =

3	1	2	4
0	1	2	3

(product of all except itself)

$1 * 2 * 4$ $\quad$ $3 * 2 * 4$ $\quad$ $3 * 1 * 4$ $\quad$ $3 * 1 * 2$

Brute force

ans =

8	24	12	6
0	1	2	3

approach 1

arr =

3	~~1~~ 0	2	4
0	1	2	3

(discarded)

$$prod = 3 * 1 * 2 * 4$$

$i = 0$, $ans = \frac{prod}{arr[0]} = \frac{3*1*2*4}{3}$

$i = 1$, $ans = \frac{prod}{arr[1]} = \frac{3*0*2*4}{0}$?? not possible

$i = 2$, $ans = \frac{prod}{arr[2]} = \frac{3*1*2*4}{2}$

$i = 3$, $ans = \frac{prod}{arr[3]} = \frac{3*1*2*4}{4}$

approch 2

) find prod. of all elements when indexes are not same

$n = 4$

arr =

0	1	2	3
3	0	2	4

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

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

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

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

Brute force

$n = 4$

$$arr = \begin{array}{cccc} 0 & 1 & 2 & 3 \\ \hline \boxed{3} & \boxed{0} & \boxed{2} & \boxed{4} \end{array} \longrightarrow$$

$\boxed{\underline{\text{TLE}}}$

ans = 1

i = 0, j = 0,

j = 1, ans = ans * 0

j = 2, ans = ans * 0 * 2

j = 3, ans = ans * 0 * 2 * 4

print ans = 0

ans = 1

i = 1, j = 0 ans = ans * 3

j = 1,

j = 2, ans = ans * 3 * 2

j = 3, ans = ans * 3 * 2 * 4

print ans = 24

ans = 1

i = 2, j = 0, ans = ans * 3

j = 1, ans = ans * 3 * 0

j = 2,

j = 3, ans = ans * 3 * 0 * 4

print ans = 0

ans = 1

i = 3, j = 0, ans = ans * 3

j = 1, ans = ans * 3 * 0

j = 2, ans = ans * 3 * 0 * 2

j = 3,

print ans = 0

psudo code

```
1) input array
2) traverse from start to end with i index
    2.1) declare ans = 1
    2.2) traverse from start to end with j index
        2.2.1) check if (i != j)
               then ans = ans * arr[j]
    2.3) print 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();
    }
    productExceptItself(arr, n);
}
public static void productExceptItself(int[] arr, int n) {
    for (int i = 0; i < n; i++) {
        int ans = 1;
        for (int j = 0; j < n; j++) {
            if ( i != j ) { // except itself
                ans = ans * arr[j];
            }
        }
        System.out.println(ans);
    }
}
```