```
1 /** Returns the sum of the integers in given array. */
2 public static int example1() {
3 int n = arr.length;  2
4 int total = 0; 2
5 total = 100 ; 1
6 return total; 1
7 }
```

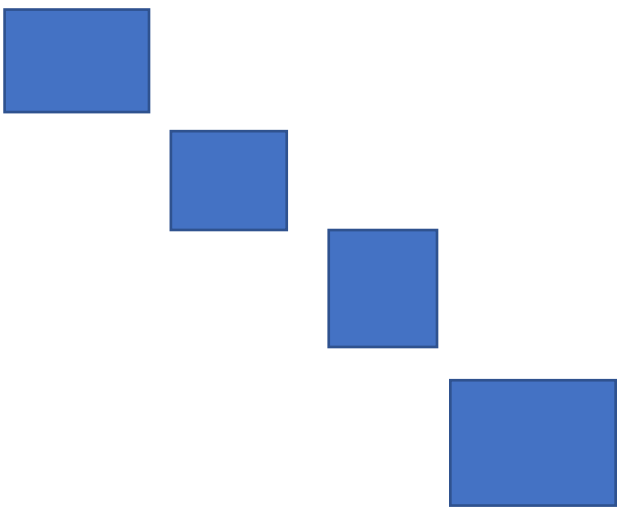
$$T(n) = 2 + 2 + 1 + 1 = 6,\ O(1)$$

```
/** Returns the sum of the integers in given array. */
2 public static int example1(int[ ] arr) {
3 int n = arr.length, total = 0;          1,4
4 for (int j=0; j < n; j++) // (j= 0 , 1 , 2 , …, n-1) + 1 = n +1
5 total += arr[j]; n
6 return total; 1
7 }
```

$$T(n) = 4 + n + 1 + n + 1 = 2n + 6\ ,\ O(n)$$

```
/** Returns the sum of the prefix sums of given array. */
18 public static int example3(int[ ] arr) {
19 int n = arr.length, total = 0; 4
20 for (int j=0; j < n; j++) ( j = 0 , 1 , 2 , .. , n -1) + 1 = n+1
21          for (int k=0; k <= n; k++)              n*[( k = 0 , 1 , 2 ,.. , n) + 1]= n^2 + 2n
22                    total += arr[j];                  n(n+1) = n^2 + n
23 return total; 1
24 }
```

$$T(n) = 4 + n + 1 + n^2 + 2n + n^2 + n + 1 = 2^* n^2 + 4n + 6,\ O(n^2)$$



```
/** Returns the sum of the prefix sums of given array. */
18 public static int example3(int[ ] arr) {
19 int n = arr.length, total = 0; 4
20 for (int j=0; j < n; j++) n+1
          arr[j] = 0;            n
21for (int k=0; k <= n; k++)  n+2
22                    total += arr[j];       n+1
23 return total; 1
24 }
```

$$4 + n + 1 + n + n + 2 + n + 1 + 1 = 4n + 9 \text{ , } O(n)$$

```
/** Returns the sum of the prefix sums of given array. */
18 public static int example3(int[ ] arr) {
19 int n = arr.length, total = 0; 4
20 for (int j=0; j < n; j++) n+1
21 for (int k=0; k <= j; k++) (k=0 , 0,1 , 0,1,2 , .. , 0,1,2,..,j) n(n+1)/2 + 1
22 total += arr[j]; n(n+1)/2
23 return total; 1
24 }
```

$O(n^2)$

nlog(n) n * n

1 + 2 + 3 + 4 + 5 + 6 = 7 + 7 + 7 = 21

[[1 + 2 + 3 + … + n-2 + n-1 + n] + [n + n-1 + n-2 + … + 3 + 2 + 1]]/2 =

[n+1 + n+1 + n+1 + … + n+1]/2 = n (n+1) / 2

	4n	2^log(n)
1	4	1
10	40	2
1000	4000	8
100000	400000	32

3n + 100 log(n)	4n = 3n + n
100 log(n)	n
Log(n)	n

1 3 2 6 3 12

1 -> 3 -> 2 -> 6 -> 3 -> 12

1326312

13 26 3

132 63

12 2 6 3