Hw task 8-7-25  
  
1) **Time Complexity: O(n)**

* The for loop runs from i = 0 to i < arr.length, so it executes **n** times, where n is the length of the input array arr.
* Inside the loop, the operation total += arr[i] is a constant time operation: O(1).
* Therefore, total time complexity = O(n).

### **Space Complexity: O(1)**

* Only a fixed amount of extra space is used: int total and int i.
* The input array arr is passed as a parameter, and no additional space is used that grows with input size.
* Therefore, space complexity = O(1) (constant space).

2) **Time Complexity: O(n^2)**

* There are **two nested loops**, each running from 0 to arr.length, so for an array of size n, the total number of iterations is n × n = n².
* The println inside the inner loop is a constant time operation O(1).

#### **Space Complexity: O(1)**

* No additional space is used except for loop variables i and j.
* Output to the console doesn't count toward algorithmic space complexity.  
    
    
  3)**Time Complexity: O(n)**
* The function calls itself recursively n times until n == 0.
* Each call does constant work (a multiplication).

#### **Space Complexity: O(n)**

* Due to **recursive call stack**, each call is stored in memory until the base case is reached.
* So, n recursive calls = O(n) space.