Optimal Strategy For A Game

You are given an array **arr** of size **n**. The elements of the array represent **n coin** of **values v_1, v_2, ...v_n**.
You play against an opponent in an **alternating** way. In each **turn**, a player selects either the **first or last coin** from the **row**, removes it from the row permanently, and **receives the value** of the coin.
You need to determine the **maximum possible amount of money** you can win if you **go first**.
Note: Both the players are playing optimally.

Example 1:

Input:

n = 4

arr[] = {5, 3, 7, 10}

Output:
15

Explanation: The user collects maximum

value as 15(10 + 5). It is guarantee that we cannot get more than 15 by any possible moves.

Example 2:

Input:

n = 4

arr[] = {8, 15, 3, 7}

Output:
22

Explanation: The user collects maximum

value as 22(7 + 15). It is guarantee that we cannot get more than 22 by any possible moves.

Your Task:
Complete the function **maximumAmount()** which takes an array **arr[]** (represent values of **n** coins) and **n** as a number of coins as a parameter and returns the **maximum possible amount of money** you can win if you **go first**.

Expected Time Complexity : O(n*n)
Expected Auxiliary Space: O(n*n)

Constraints:
$2 <= n <= 10^3$
$1 <= arr[i] <= 10^6$