```python
import numpy as np

No_of_antennas = [1, 2, 4, 8, 16, 32]

# The number of blocks for ergodic channel (realizations)
No_of_blocks_ergodic = 1000

# SNR values in dB
SNR_range = np.arange(0, 21)

# Convert SNR values from dB to linear
SNR = 10 ** (SNR_range / 10)

# Function to calculate capacity for a given SNR and number of antennas
def calculate_capacity(No_of_transmitter, snr):
    # Set number of receivers equal to number of transmitters
    No_of_reciever = No_of_transmitter

    # Generate channel matrix H
    H = (1 / np.sqrt(2)) * (np.random.randn(No_of_transmitter, No_of_reciever, No_of_blocks_ergodic) + 1j * np.random.randn(No_of_transmitter, No_of_reciever, No_of_blocks_ergodic))

    # Initialize array to store channel capacities for each block
    Channel_Capacity_Block = np.zeros(No_of_blocks_ergodic)

    # Loop over each block
    for k in range(No_of_blocks_ergodic):
        # Extract H for current block
        Hcurr = H[:, :, k]

        # Compute singular values of H
        singular_values = np.linalg.svd(Hcurr, compute_uv=False)

        # Calculate eigenvalues from squared singular values
        eigenvalues = singular_values ** 2

        # Calculate channel capacity for each block
        Channel_Capacity_Block[k] = np.sum(np.log2(1 + (snr * eigenvalues) / No_of_transmitter))

    # Calculate mean channel capacity
    return np.mean(Channel_Capacity_Block)

# Calculate Ergodic Rayleigh Fading channel capacity for each configuration
for No_of_transmitter in No_of_antennas:
    capacities = [calculate_capacity(No_of_transmitter, snr) for snr in SNR]
    print(f"Antennas={No_of_transmitter}: ", capacities)
```