```
import numpy as np

def equal_frequency_binning(data, num_bins):
    sorted_data = sorted(data)
    bin_size = len(data) // num_bins
    bins = []
    for i in range(num_bins):
        start_index = i * bin_size
        end_index = start_index + bin_size
        bins.append(sorted_data[start_index:end_index])
    return bins



def mean_binning(data, num_bins):

    mean = int(round(np.mean(data)))
    sorted_data = sorted(data)
    bin_size = len(data) // num_bins

    bins = []
    for i in range(num_bins):
        start_index = i * bin_size
        end_index = start_index + bin_size
        bin_data = sorted_data[start_index:end_index]
        bin_mean = int(round(np.mean(bin_data)))
        bin_values = [bin_mean] * len(bin_data)
        bins.extend(bin_values)

    return bins

def boundary_binning(data, num_bins):
    sorted_data = sorted(data)
    bin_size = len(data) // num_bins

    bins = []
    for i in range(num_bins):
        start_index = i * bin_size
        end_index = start_index + bin_size
        bin_data = sorted_data[start_index:end_index]

        low, high = bin_data[0], bin_data[-1]
        binned_data = []
        for element in bin_data:
            if abs(element - low) < abs(element - high):
                binned_data.append(low)
            else:
                binned_data.append(high)

        bins.append(binned_data)
```

```
    return bins


def get_user_input():
    data_str = input("Enter the data (comma-separated): ")
    data = [int(x) for x in data_str.split(",")]
    num_bins = int(input("Enter the number of bins: "))
    return data, num_bins

data, num_bins = get_user_input()
data.sort()
print("Sorted Data: ", data)

equal_frequency_bins = equal_frequency_binning(data, num_bins)
print("* Partition into Equal-Frequency Bins:")
for i, bin in enumerate(equal_frequency_bins, start=1):
    print(f"Bin{i}: {bin}")

mean_bins = mean_binning(data, num_bins)
print("\n* Smoothing by bin Means:")
for i, bin_group in enumerate(zip(*[iter(mean_bins)] * 4), start=1):
    print(f"Bin{i}: {bin_group}")

boundary_bins = boundary_binning(data, num_bins)
print("\n* Smoothing by Bin Boundary:")
for i, bin in enumerate(boundary_bins, start=1):
    print(f"Bin{i}: {bin}")
```

Output:

```
Enter the data (comma-separated): 5,9,11,12,14,16,18,20,23,26,28,30
Enter the number of bins: 3
Sorted Data:  [5, 9, 11, 12, 14, 16, 18, 20, 23, 26, 28, 30]
* Partition into Equal-Frequency Bins:
Bin1: [5, 9, 11, 12]
Bin2: [14, 16, 18, 20]
Bin3: [23, 26, 28, 30]

* Smoothing by bin Means:
Bin1: (9, 9, 9, 9)
Bin2: (17, 17, 17, 17)
Bin3: (27, 27, 27, 27)

* Smoothing by Bin Boundary:
Bin1: [5, 12, 12, 12]
Bin2: [14, 14, 20, 20]
Bin3: [23, 23, 30, 30]
PS C:\Users\tejas\Desktop>
```