Basic Introduction to NumPy: Problems

This problem set is designed to test your understanding of fundamental NumPy concepts. The problems are based on a scenario involving daily operational data from a food service business. Your task is to use NumPy to solve these analytical challenges.

Problem 1: Daily Revenue Analysis

You have been given the daily revenue data for two weeks. The data is provided as a Python list. Your goal is to convert this list into a NumPy array and perform basic analysis.

- **Task 1.1:** Create a 1-D NumPy array named `daily_revenue_lakhs` from the provided list. The values are in Lakhs INR.
- **Task 1.2:** Calculate and print the total revenue for the two weeks.
- **Task 1.3:** Calculate and print the average daily revenue.

> ***Hint:*** *Use the `np.array()` function to convert the list. NumPy arrays have built-in methods like `.sum()` and `.mean()` for easy calculations.*

```
# Given data
daily_revenue_data = [
    5.5, 6.2, 5.8, 7.1, 7.5, 8.0, 9.2,
    6.0, 6.5, 7.0, 7.8, 8.5, 9.0, 9.5
]

# Write your code below this line
```

Problem 2: Product and Cost Analysis

The business tracks the quantity and cost of three main product types: 'Cups', 'Cones', and 'Specials'. The data is organized in a 2-D structure. You need to use a 2-D NumPy array to analyze this data.

- **Task 2.1:** Create a 2-D NumPy array named `product_sales` from the nested list. The first row represents units sold, and the second row represents cost per unit. The columns correspond to 'Cups', 'Cones', and 'Specials'.
- **Task 2.2:** Use array indexing to print the total number of 'Cones' sold.
- **Task 2.3:** Use array slicing to print the costs for 'Cones' and 'Specials'.

> ***Hint:*** *Remember that NumPy arrays are indexed starting from 0. For slicing, use the colon : notation.*

```
# Given data
product_sales_data = [
    [1500, 2200, 800],  # Units sold for Cups, Cones, Specials
    [30, 20, 150]       # Cost per unit (in INR) for Cups, Cones, Specials
]

# Write your code below this line
```

Problem 3: Cost and Profit Calculations with Vectorization

The business wants to automate cost and profit calculations. You have an array of unit costs and a fixed profit margin. You need to use NumPy's vectorized operations to perform calculations efficiently without using a `for` loop.

- **Task 3.1:** Create a 1-D NumPy array named `unit_costs` with the provided data.
- **Task 3.2:** Define a variable `profit_margin_percent` with a value of 25.
- **Task 3.3:** Calculate the selling price for each unit by adding the profit margin. The formula is: `selling_price = unit_cost + (unit_cost * profit_margin_percent / 100)`. Store the result in a new array named `selling_prices`.

> ***Hint:*** *NumPy's operations on arrays are element-wise. You can multiply an array by a number, and that operation will be applied to every element in the array.*

```
# Given data
unit_costs_data = [45, 60, 55, 70, 80] # In INR
```

```
# Write your code below this line
```

Problem 4: Inventory Management with Broadcasting

You need to update inventory levels for various ingredients. You have the starting inventory and the ingredients used per batch for three different flavors. All ingredient amounts are in kilograms (kg).

- **Task 4.1:** Create a 2-D NumPy array named `ingredients_used`. The rows represent three flavors and the columns represent three ingredients (e.g., 'Vanilla', 'Chocolate', 'Fruits').
- **Task 4.2:** Create a 1-D NumPy array named `batches_produced` representing the number of batches for each flavor.
- **Task 4.3:** Use NumPy broadcasting to calculate the total amount of each ingredient used across all batches. Print the result.

> ***Hint:*** *You might need to change the shape of one of the arrays to allow broadcasting. Use `.reshape()` or to make a 1-D array compatible with a 2-D array for element-wise multiplication.*

```
# Given data
# rows: Flavors (A, B, C)
# cols: Ingredients (Vanilla, Chocolate, Fruit)
ingredients_used_per_batch = [
    [0.5, 0.3, 0.1],
    [0.2, 0.6, 0.0],
    [0.4, 0.2, 0.3]
]

batches_produced_data = [100, 150, 200]

# Write your code below this line
```