1 | Experiential Learning Component

1.1 | UNIT I : Experiential Learning - 1

[Level-1: 5Q, Level-2: 1Q]

1. **LEVEL - 1 : Size of Data Types**
   - Objective: Write a program to determine and print the sizes of different data types in C.
   - Instructions:
     - Use the `sizeof` operator to find the sizes of the following data types: `int`, `float`, `double`, `char`, and `long`.
     - Print the sizes in bytes.
   - Sample Output:

```
Size of int: 4 bytes
Size of float: 4 bytes
Size of double: 8 bytes
Size of char: 1 byte
Size of long: 8 bytes
```

2. **LEVEL - 1 : Pointers and Arrays**
   - Objective: Write a program to demonstrate pointer arithmetic with a 1-dimensional array.
   - Instructions:
     - Declare a 1-dimensional array of integers and initialize it with 5 values.
     - Use a pointer to traverse the array and print each element using pointer arithmetic.
   - Sample Output:

```
Array elements: 10 20 30 40 50
```

3. **LEVEL - 1 : 2-Dimensional Arrays**
   - Objective: Create a program that manipulates a 2-dimensional array.
   - Instructions:
     - Declare and initialize a 2-dimensional array (e.g., a 3x3 matrix).
     - Write a function to calculate the sum of the elements of the matrix and return the result.
     - Print the sum.
   - Sample Output:

```
Sum of the matrix elements: 45
```

4. **LEVEL - 1 : Calculating Address in Arrays**
   - Objective: Create a program that calculates the address of a specific element in both 1-dimensional and 2-dimensional arrays using their respective formulas.

- Instructions:
  - **1-Dimensional Array:**
    - Declare and initialize a 1-dimensional array with 5 elements.
    - Write a function to calculate the address of an element using the formula:

$$\text{Address} = \text{Base Address} + (i \times \text{Size of element})$$

      where i is the index of the element.
    - Print the address of a specified element.
  - **2-Dimensional Array:**
    - Declare and initialize a 2-dimensional array with 3 rows and 3 columns.
    - Write a function to calculate the address of an element using the formula for row-major order:

$$\text{Address} = \text{Base Address} + [(i \times \text{Total Columns}) + j] \times \text{Size of element}$$

      where i and j represent the row and column indices of the element.
    - Print the address of a specified element.
- Sample Output:

```
Address of element at index 2 in 1D Array: 2012
Address of element at (1, 2) in 2D Array: 3056
```

5. **LEVEL - 1 : Arrays as Parameters**
   - Objective: Write a program that uses arrays as function parameters.
   - Instructions:
     - Create a function that takes an array of integers and its size as parameters.
     - The function should calculate and return the average of the elements in the array.
     - In the `main` function, declare an array, initialize it, and call the average function.
   - Sample Output:

```
Average of the array: 25.0
```

6. **LEVEL - 2 : Structures and Arrays of Structures**
   - Objective: Create a program that demonstrates the use of structures and passing them as parameters.
   - Instructions:
     - Define a structure called `Student` with fields for name, age, and marks.
     - Create an array of `Student` structures and initialize it with sample data.
     - Write a function that takes the array of `Student` structures and the size of the array as parameters to calculate the average marks.
     - Print the average marks.
   - Sample Output:

```
Average marks of students: 78.5
```