LAB: 2

Subject: Data Structures

Topic: Introduction to Arrays and its implementation

What is Array?

An array is a data structure that stores a fixed-size sequential collection of elements of the same type. In other words, it’s a collection of variables (called elements), all of the same type, stored under a single variable name.

Key Characteristics of an Array:

1. Fixed Size: Once you declare an array with a certain size, it cannot change.

2. Same Type: All elements in the array are of the same data type (e.g., integers, characters, floats).

3. Indexing: Elements are stored in contiguous memory locations, and each element is accessed using its index (starting from 0 in most programming languages).

Why Arrays Are Used:

1. Efficient Access: Arrays allow you to store multiple elements in a single variable, which is easier to manage than multiple individual variables.

2. Random Access: You can directly access any element in the array using its index in constant time.

3. Organized Data Storage: Arrays are used to store data that can be processed together, like a list of numbers, characters, or objects.

4. Memory Efficiency: Since all elements are of the same type and stored in contiguous memory locations, arrays are efficient in terms of memory usage.

Example: 1

```
#include <iostream>
#include <string>
using namespace std;

int main() {
  string cars[4] = {"Volvo", "BMW", "Ford", "Mazda"};
  cout << cars[0];
  return 0;
}
```

Explained:

```
#include <iostream> // Includes the input/output stream library for using cout
#include <string>// Includes the string library for using the string type
using namespace std; // Allows us to avoid using the std:: prefix before standard library objects like cout

int main() {
    // Declare and initialize an array of strings named 'cars' with 4 elements
    string cars[4] = {"Suzuki", "Corolla", "Ford", "Mazda"};

    // Output the first element of the 'cars' array (which is "Volvo") to the console
    cout << cars[0];

    return 0;  // Return 0 indicates that the program ended successfully
}
```

Example: 2

```
#include <iostream>
#include <string>
using namespace std;

int main() {
  string cars[5] = {"Volvo", "BMW", "Ford", "Mazda", "Tesla"};
  for (int i = 0; i < 5; i++) {
    cout << cars[i] << "\n";
  }
  return 0;
}
```

Explained:

```
#include <iostream>  // Includes the input/output stream library for using cout
#include <string>    // Includes the string library for using the string type
using namespace std; // Allows us to avoid using the std:: prefix before
standard library objects like cout

int main() {
    // Declare and initialize an array of strings named 'cars' with 5 elements
    string cars[5] = {"Volvo", "BMW", "Ford", "Mazda", "Tesla"};

    // Loop through each element in the array
    // The loop starts at index 0 and runs until it reaches index 4 (less than 5)
    for (int i = 0; i < 5; i++) {
        // Output each element in the 'cars' array followed by a newline character
        cout << cars[i] << "\n";
    }
    return 0;  // Return 0 indicates that the program ended successfully
}
```

Example: 3

```
#include <iostream>
 #include <string>
 using namespace std;
 int main() {
 string cars[5] = {"Volvo", "BMW", "Ford", "Mazda", "Tesla"};
 for (int i = 0; i < 5; i++)
{ cout << i << " = " << cars[i] << "\n";
}
return 0;
 }
```

Explained:

```
#include <iostream>  // Includes the input/output stream library for using cout
#include <string>    // Includes the string library for using the string type
using namespace std; // Allows us to avoid using the std:: prefix before standard library objects like cout

int main() {
    // Declare and initialize an array of strings named 'cars' with 5 elements
    string cars[5] = {"Volvo", "BMW", "Ford", "Mazda", "Tesla"};
    // Loop through each element in the array
    // The loop runs from index 0 to 4 (less than 5)
    for (int i = 0; i < 5; i++) {
        // Output the index (i) and corresponding element in the 'cars' array
        // Each line will print in the format: index = car_name
        cout << i << " = " << cars[i] << "\n";
    }

    return 0;  // Return 0 indicates that the program ended successfully
}
```

Example: 4

Explained:

```
#include <iostream>  // Includes the input/output stream library for using cout
using namespace std; // Allows us to avoid using the std:: prefix before standard library objects like cout

int main() {
    // Declare and initialize an array of integers named 'myNumbers' with 5 elements
    int myNumbers[5] = {10, 20, 30, 40, 50};

    // Loop through each element in the 'myNumbers' array
    // The loop runs from index 0 to 4 (less than 5)
    for (int i = 0; i < 5; i++) {
```

```
    // Output each element in the 'myNumbers' array followed by a newline character
    cout << myNumbers[i] << "\n";
  }

  return 0;  // Return 0 indicates that the program ended successfully
}
```

Example: 5

```
#include <iostream>
#include <string>
using namespace std;

int main() {
 string cars[5];
 cars[0] = "Volvo";
 cars[1] = "BMW";
 cars[2] = "Ford";
 cars[3] = "Mazda";
 cars[4] = "Tesla";
 for(int i = 0; i < 5; i++) {
  cout << cars[i] << "\n";
 }
 return 0;
}
```

Explained:

```
#include <iostream>  // Includes the input/output stream library for using cout
#include <string>    // Includes the string library for using the string type
using namespace std; // Allows us to avoid using the std:: prefix before standard library objects like cout


int main() {
    // Declare an array of strings named 'cars' with 5 elements
    string cars[5];


    // Assign values to each element of the 'cars' array
    cars[0] = "Volvo";
    cars[1] = "BMW";
    cars[2] = "Ford";
    cars[3] = "Mazda";
    cars[4] = "Tesla";


    // Loop through each element in the 'cars' array
    // The loop runs from index 0 to 4 (less than 5)
    for(int i = 0; i < 5; i++) {
        // Output each element in the 'cars' array followed by a newline character
        cout << cars[i] << "\n";
    }


    return 0;  // Return 0 indicates that the program ended successfully
}
```

Example: 6

```
#include <iostream>
using namespace std;

int main() {
  int myNumbers[5] = {10, 20, 30, 40, 50};
  for (int i = 0; i < 5; i++) {
    cout << myNumbers[i] << "\n";
  }
  return 0;
}
```

Explained:

```
#include <iostream>  // Includes the input/output stream library for using cout
using namespace std; // Allows us to avoid using the std:: prefix before standard library objects like cout

int main() {
    // Declare and initialize an array of integers named 'myNumbers' with 5 elements
    int myNumbers[5] = {10, 20, 30, 40, 50};

    // Loop through each element in the 'myNumbers' array
    // The loop runs from index 0 to 4 (less than 5)
    for (int i = 0; i < 5; i++) {
        // Output each element in the 'myNumbers' array followed by a newline character
        cout << myNumbers[i] << "\n";
    }
    return 0;  // Return 0 indicates that the program ended successfully
}
```

Example: 7

```
#include <iostream>
using namespace std;

int main() {
  int myNumbers[5] = {10, 20, 30, 40, 50};
  cout << sizeof(myNumbers);
  return 0;
}
```

Explained:

```
#include <iostream>  // Includes the input/output stream library for using cout
using namespace std; // Allows us to avoid using the std:: prefix before standard library objects like cout

int main() {
    // Declare and initialize an array of integers named 'myNumbers' with 5 elements
    int myNumbers[5] = {10, 20, 30, 40, 50};

    // Output the size (in bytes) of the entire 'myNumbers' array
    // The sizeof operator returns the total number of bytes occupied by the array
    cout << sizeof(myNumbers);
    return 0;  // Return 0 indicates that the program ended successfully
}
```

The sizeof operator returns the size (in bytes) of the array myNumbers.
Since each int typically takes up **4 bytes** and there are 5 integers in the array, the output would generally be **20 bytes** (5 elements * 4 bytes per element).

Example: 8

```
#include <iostream>
using namespace std;
int main() {
    int numbers[5] = {10, 20, 30, 40, 50}; // Initializing an array of size 5
    for (int i = 0; i < 5; i++) {
        cout << "Element at index " << i << " is: " << numbers[i] << endl;
    }
    return 0;
}
```

Explained:

```
#include <iostream>  // Include the input-output stream library for standard I/O
using namespace std; // Allows the use of standard names without the 'std::' prefix
int main() {
    // Initializing an array of 5 integers with predefined values
    int numbers[5] = {10, 20, 30, 40, 50};
    // Loop through the array elements
    for (int i = 0; i < 5; i++) {
        // Output the current element with its index
        cout << "Element at index " << i << " is: " << numbers[i] << endl;
    }
    return 0; // Return 0 to indicate successful program execution
}
```

Example: 9

```
#include <iostream>
using namespace std;
int main() {
    int arr[5], largest;
    cout << "Enter 5 numbers:" << endl;
    for (int i = 0; i < 5; i++) {
        cin >> arr[i];
    }
    largest = arr[0]; // Assume first element is the largest
    for (int i = 1; i < 5; i++) {
        if (arr[i] > largest) {
            largest = arr[i];
        }
    }
    cout << "The largest number is: " << largest << endl;
    return 0;
}
```

Explained:

```
#include <iostream>  // Include the input-output stream library
using namespace std; // Allow usage of standard names without the 'std::' prefix
int main() {
    int arr[5], largest;  // Declare an array to store 5 integers and a variable to hold the largest number

    // Prompt the user to enter 5 numbers

    cout << "Enter 5 numbers:" << endl;

    // Loop to input 5 numbers from the user and store them in the array

    for (int i = 0; i < 5; i++) {

        cin >> arr[i]; // Store each input in the corresponding index of the array

    }

    largest = arr[0]; // Assume the first element is the largest initially

    // Loop through the remaining elements of the array to find the largest

    for (int i = 1; i < 5; i++) {

        if (arr[i] > largest) { // If the current element is larger than 'largest'

            largest = arr[i];   // Update 'largest' with the current element

        }

    }

    // Output the largest number found

    cout << "The largest number is: " << largest << endl;

    return 0; // Return 0 to indicate successful program execution

}
```