Assignment_DSA_LAB_03

Kashan Shahid

53686

CS 3-1

Q1. 2D Array: Sum, Multiplication, and Average.

Code:

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

int main() {
    int rows, cols;
    cout << "Enter number of rows and columns: ";
    cin >> rows >> cols;

    int arr[rows][cols];
    int sum = 0, product = 1;
    double avg = 0;

    // Input elements of the array
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            cout << "Enter element [" << i << "][" << j << "]: ";
            cin >> arr[i][j];
```

```
            sum += arr[i][j];
            product *= arr[i][j];
        }
    }

    avg = static_cast<double>(sum) / (rows * cols);

    // Display results
    cout << "Sum: " << sum << endl;
    cout << "Product: " << product << endl;
    cout << "Average: " << avg << endl;

    return 0;
}
```

Output:

```
/tmp/h3ucdMk2w8.o
Enter number of rows and columns: 3
2
Enter element [0][0]: 1
Enter element [0][1]: 2
Enter element [1][0]: 3
Enter element [1][1]: 4
Enter element [2][0]: 5
Enter element [2][1]: 6
Sum: 21
Product: 720
Average: 3.5


=== Code Execution Successful ===
```

Q2. Swap Values Using Pointers

Code:

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

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

int main() {
    int x, y;
    cout << "Enter two numbers: ";
    cin >> x >> y;

    swap(&x, &y);

    cout << "After swapping: x = " << x << ", y = " << y << endl;

    return 0;
}
```

Output:

```
Enter two numbers: 12
15
After swapping: x = 15, y = 12


=== Code Execution Successful ===
```

Q3. Find Largest and Smallest in Array.

Code:

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

int main() {
    int arr[10], largest, smallest;

    // Input elements
    for (int i = 0; i < 10; i++) {
        cout << "Enter value " << i + 1 << ": ";
        cin >> arr[i];
    }

    largest = smallest = arr[0];

    // Finding largest and smallest
    for (int i = 1; i < 10; i++) {
```

```
        if (arr[i] > largest)
            largest = arr[i];
        if (arr[i] < smallest)
            smallest = arr[i];
    }

    cout << "Largest: " << largest << endl;
    cout << "Smallest: " << smallest << endl;

    return 0;
}
```

Output:

```
/tmp/TBJQ83oHCk.o
Enter value 1: 32
Enter value 2: 33
Enter value 3: 23
Enter value 4: 44
Enter value 5: 55
Enter value 6: 10
Enter value 7: 9
Enter value 8: 6
Enter value 9: 23
Enter value 10: 11
Largest: 55
Smallest: 6


=== Code Execution Successful ===
```

Q4. Rainfall Calculation.

Code:

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

int main() {
    double rainfall[12], total = 0, average;
    int highestMonth = 0, lowestMonth = 0;

    // Input rainfall data
    for (int i = 0; i < 12; i++) {
        cout << "Enter rainfall for month " << i + 1 << ": ";
        cin >> rainfall[i];
        total += rainfall[i];

        if (rainfall[i] > rainfall[highestMonth])
            highestMonth = i;
        if (rainfall[i] < rainfall[lowestMonth])
            lowestMonth = i;
    }

    average = total / 12;

    // Display results
    cout << "Total rainfall: " << total << endl;
```

```
    cout << "Average monthly rainfall: " << average << endl;
    cout << "Highest rainfall in month: " << highestMonth + 1 << endl;
    cout << "Lowest rainfall in month: " << lowestMonth + 1 << endl;

    return 0;
}
```

Output:

```
/tmp/2tDpFXEGeU.o
Enter rainfall for month 1: 5
Enter rainfall for month 2: 3
Enter rainfall for month 3: 1
Enter rainfall for month 4: 2
Enter rainfall for month 5: 4
Enter rainfall for month 6: 3
Enter rainfall for month 7: 1
Enter rainfall for month 8: 4
Enter rainfall for month 9: 2
Enter rainfall for month 10: 5
Enter rainfall for month 11: 4
Enter rainfall for month 12: 2
Total rainfall: 36
Average monthly rainfall: 3
Highest rainfall in month: 1
Lowest rainfall in month: 3


=== Code Execution Successful ===
```

Q5. 2D Array Operations.

Code:

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

```
int main() {
    int rows = 3, cols = 3, arr[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int total = 0, rowTotal, colTotal;

    // Calculate total and average
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            total += arr[i][j];
        }
    }
    double average = total / static_cast<double>(rows * cols);

    // Get row and column totals
    int rowIndex = 1, colIndex = 2;
    rowTotal = 0;
    colTotal = 0;
    for (int j = 0; j < cols; j++) {
        rowTotal += arr[rowIndex][j];
    }
    for (int i = 0; i < rows; i++) {
        colTotal += arr[i][colIndex];
    }

    // Find highest in row and column
```

```
    int highestInRow = arr[rowIndex][0];
    int highestInCol = arr[0][colIndex];
    for (int j = 1; j < cols; j++) {
        if (arr[rowIndex][j] > highestInRow)
            highestInRow = arr[rowIndex][j];
    }
    for (int i = 1; i < rows; i++) {
        if (arr[i][colIndex] > highestInCol)
            highestInCol = arr[i][colIndex];
    }

    cout << "Total: " << total << endl;
    cout << "Average: " << average << endl;
    cout << "Row total: " << rowTotal << endl;
    cout << "Column total: " << colTotal << endl;
    cout << "Highest in row: " << highestInRow << endl;
    cout << "Highest in column: " << highestInCol << endl;

    return 0;
}
```

Output:

```
/tmp/iJE65xxtko.o
Total: 45
Average: 5
Row total: 15
Column total: 18
Highest in row: 6
Highest in column: 9
```

Q6. Sum of Odd Integers Using Dynamic Array.

Code:

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

int main() {
    int n, sum = 0;
    cout << "Enter the size of the array: ";
    cin >> n;

    int *arr = new int[n];

    // Input values and calculate sum of odd numbers
    for (int i = 0; i < n; i++) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> arr[i];
        if (arr[i] % 2 != 0) {
            sum += arr[i];
        }
    }

    cout << "Sum of odd integers: " << sum << endl;

    delete[] arr;
```

```
    return 0;
}
```

Output:

```
/tmp/da1QRNs8H8.o
Enter the size of the array: 4
Enter element 1: 3
Enter element 2: 7
Enter element 3: 2
Enter element 4: 4
Sum of odd integers: 10


=== Code Execution Successful ===
```

Q7. Accessing Value Using Pointer.

Code:

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

int main() {
    int var = 10;
    int *ptr = &var;

    // Display value and address

    cout << "Value of var: " << var << endl;

    cout << "Value at pointer ptr: " << *ptr << endl
```

```
    return 0;
}
```

Output:

```
Value of var: 10
Value at pointer ptr: 10


=== Code Execution Successful ===
```

Q8. Assigning Values to Pointers.

Code:

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

int main() {
    int a, b;
    int *ptrA, *ptrB;

    cout << "Enter value for a: ";
    cin >> a;
    cout << "Enter value for b: ";
    cin >> b;

    ptrA = &a;
    ptrB = &b;
```

```
    cout << "Value pointed by ptrA: " << *ptrA << endl;
    cout << "Value pointed by ptrB: " << *ptrB << endl;

    return 0;
}
```

Output:

```
/tmp/kDM2Qw8h3w.o
Enter value for a: 12
Enter value for b: 45
Value pointed by ptrA: 12
Value pointed by ptrB: 45


=== Code Execution Successful ===
```

Q9. Calculator Using Functions.

Code:

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

void menu() {
    cout << "1. Addition\n2. Subtraction\n3. Multiplication\n4. Division\n5. Power\n";
}
```

```
int addition(int a, int b) {
    return a + b;
}

int subtraction(int a, int b) {
    return a - b;
}

int multiplication(int a, int b) {
    return a * b;
}

double division(int a, int b) {
    return static_cast<double>(a) / b;
}

int power(int number, int pow) {
    return pow == 0 ? 1 : number * power(number, pow - 1);
}

int main() {
    int choice, a, b;

    menu();
    cout << "Choose an option: ";
```

```
cin >> choice;
cout << "Enter two numbers: ";
cin >> a >> b;

switch (choice) {
    case 1:
        cout << "Result: " << addition(a, b) << endl;
        break;
    case 2:
        cout << "Result: " << subtraction(a, b) << endl;
        break;
    case 3:
        cout << "Result: " << multiplication(a, b) << endl;
        break;
    case 4:
        if (b != 0)
            cout << "Result: " << division(a, b) << endl;
        else
            cout << "Error: Division by zero." << endl;
        break;
    case 5:
        cout << "Result: " << power(a, b) << endl;
        break;
    default:
        cout << "Invalid option." << endl;
```

```
    }

    return 0;
}
```

Output:

```
/tmp/AVoM1XompF.o
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Power
Choose an option: 1
Enter two numbers: 2
2
Result: 4


=== Code Execution Successful ===
```