practice-2

November 25, 2024

1 Practice-2

[13]:
```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
using namespace std;
```

Find sum of all array elements using recursion.

[2]:
```cpp
int sumArray(int arr[], int n) {
    if (n <= 0)
        return 0;
    return arr[n - 1] + sumArray(arr, n - 1);
}
```

[3]:
```cpp
int arr[] = {1, 2, 3, 4, 5};
int n = sizeof(arr) / sizeof(arr[0]);
cout << "Sum of array elements: " << sumArray(arr, n) << endl;
return 0;
```

```
Sum of array elements: 15
```

Create an array ‘a1’ with ‘n’ elements. Insert an element in ith position of ‘a1’ and also delete an element from jth position of ‘a1’.

[4]:
```cpp
int n = 5;
vector<int> a1 = {1, 2, 3, 4, 5};
int i = 2;
int elementToInsert = 10;
a1.insert(a1.begin() + i, elementToInsert);
int j = 4;
a1.erase(a1.begin() + j);

cout << "Updated array: ";
for (int k = 0; k < a1.size(); k++) {
    cout << a1[k] << " ";
}
```

```
cout << endl;
```

```
Updated array: 1 2 10 3 5
```

Convert uppercase string to lowercase using for loop.

[5]:
```
string str = "HELLO WORLD";
for (int i = 0; i < str.length(); i++) {
    if (str[i] >= 'A' && str[i] <= 'Z') {
        str[i] = str[i] + 32;
    }
}
cout << "Lowercase string: " << str << endl;
```

```
Lowercase string: hello world
```

Find the sum of rows and columns of matrix of given order (row x column).

[7]:
```
int rows = 3, cols = 3;
vector<vector<int>> matrix = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};

for (int i = 0; i < rows; i++) {
    int rowSum = 0;
    for (int j = 0; j < cols; j++) {
        rowSum += matrix[i][j];
    }
    cout << "Sum of row " << i + 1 << ": " << rowSum << endl;
}

for (int j = 0; j < cols; j++) {
    int colSum = 0;
    for (int i = 0; i < rows; i++) {
        colSum += matrix[i][j];
    }
    cout << "Sum of column " << j + 1 << ": " << colSum << endl;
}
```

```
Sum of row 1: 6
Sum of row 2: 15
Sum of row 3: 24
Sum of column 1: 12
Sum of column 2: 15
Sum of column 3: 18
Sum of row 2: 15
Sum of row 3: 24
Sum of column 1: 12
Sum of column 2: 15
Sum of column 3: 18
```

Find the product of two matrices using pointers.

[8]:
```
void multiplyMatrices(int* mat1, int* mat2, int* result, int r1, int c1, int␣
↪c2) {
    for (int i = 0; i < r1; i++) {
        for (int j = 0; j < c2; j++) {
            *(result + i * c2 + j) = 0;
            for (int k = 0; k < c1; k++) {
                *(result + i * c2 + j) += *(mat1 + i * c1 + k) * *(mat2 + k *␣
↪c2 + j);
            }
        }
    }
}
```

[9]:
```
int r1 = 2, c1 = 3, r2 = 3, c2 = 2;
int mat1[2][3] = {{1, 2, 3}, {4, 5, 6}};
int mat2[3][2] = {{7, 8}, {9, 10}, {11, 12}};
int result[2][2];

multiplyMatrices((int*)mat1, (int*)mat2, (int*)result, r1, c1, c2);

cout << "Product of the matrices:" << endl;
for (int i = 0; i < r1; i++) {
    for (int j = 0; j < c2; j++) {
        cout << result[i][j] << " ";
    }
    cout << endl;
}
```

```
Product of the matrices:
58 64
58 64
139 154
```

Store ‘n’ numbers (integers or real) in an array. Conduct a linear search for a given number and report success or failure in the form of a suitable message.

[10]:
```
bool linearSearch(const vector<int>& arr, int target) {
    for (int i = 0; i < arr.size(); i++) {
        if (arr[i] == target) {
            return true;
        }
    }
    return false;
}
```

[11]:
```
int n = 5;
vector<int> arr = {10, 20, 30, 40, 50};
int target = 30;

if (linearSearch(arr, target)) {
    cout << "Element found in the array." << endl;
} else {
    cout << "Element not found in the array." << endl;
}
```

```
Element found in the array.
```

Write a program to reverse an array.

[12]:
```
void reverseArray(vector<int>& arr) {
    int start = 0;
    int end = arr.size() - 1;
    while (start < end) {
        swap(arr[start], arr[end]);
        start++;
        end--;
    }
}

vector<int> arr = {10, 20, 30, 40, 50};
reverseArray(arr);

cout << "Reversed array: ";
for (int i = 0; i < arr.size(); i++) {
    cout << arr[i] << " ";
}
cout << endl;
```

```
Reversed array: 50 40 30 20 10
```

Move all zeroes to end of array

[19]:
```
void moveZeroesToEnd(vector<int>& arr) {
    int nonZeroPos = 0;
    for (int i = 0; i < arr.size(); i++) {
        if (arr[i] != 0) {
            swap(arr[i], arr[nonZeroPos]);
            nonZeroPos++;
        }
    }
}
```

```cpp
vector<int> arr = {0, 1, 0, 3, 12};
moveZeroesToEnd(arr);

cout << "Array after moving zeroes to the end: ";
for (int i = 0; i < arr.size(); i++) {
    cout << arr[i] << " ";
}
cout << endl;
```

```
Array after moving zeroes to the end: 1 3 12 0 0
```