**Find the Second Largest Element in an Array**

**Problem:** Write a function to find the second largest element in a given array.

cpp

#include <iostream>

#include <vector>

#include <limits.h>

using namespace std;

int findSecondLargest(const vector<int>& arr) {

int largest = INT\_MIN, secondLargest = INT\_MIN;

for (int num : arr) {

if (num > largest) {

secondLargest = largest;

largest = num;

} else if (num > secondLargest && num != largest) {

secondLargest = num;

}

}

return (secondLargest == INT\_MIN) ? -1 : secondLargest;

}

int main() {

vector<int> arr = {12, 35, 1, 10, 34, 1};

cout << "Second Largest: " << findSecondLargest(arr) << endl;

return 0;

}

**2. Reverse a String Using Recursion**

**Problem:** Reverse a string using a recursive function.

cpp

#include <iostream>

using namespace std;

void reverseString(string& str, int left, int right) {

if (left >= right) return;

swap(str[left], str[right]);

reverseString(str, left + 1, right - 1);

}

int main() {

string str = "Hello, World!";

reverseString(str, 0, str.size() - 1);

cout << "Reversed String: " << str << endl;

return 0;

}

**3. Implement a Stack Using a Linked List**

**Problem:** Create a stack implementation using a linked list.

cpp

#include <iostream>

using namespace std;

struct Node {

int data;

Node\* next;

Node(int val) : data(val), next(nullptr) {}

};

class Stack {

private:

Node\* top;

public:

Stack() : top(nullptr) {}

void push(int val) {

Node\* newNode = new Node(val);

newNode->next = top;

top = newNode;

}

void pop() {

if (!top) {

cout << "Stack Underflow" << endl;

return;

}

Node\* temp = top;

top = top->next;

delete temp;

}

int peek() {

if (!top) {

cout << "Stack is Empty" << endl;

return -1;

}

return top->data;

}

bool isEmpty() {

return top == nullptr;

}

};

int main() {

Stack stack;

stack.push(10);

stack.push(20);

stack.push(30);

cout << "Top element: " << stack.peek() << endl;

stack.pop();

cout << "Top element after pop: " << stack.peek() << endl;

return 0;

}

**4. Check if a Number is a Power of Two**

**Problem:** Determine if a given number is a power of two.

cpp

#include <iostream>

using namespace std;

bool isPowerOfTwo(int n) {

return (n > 0) && ((n & (n - 1)) == 0);

}

int main() {

int num = 16;

if (isPowerOfTwo(num)) {

cout << num << " is a power of two." << endl;

} else {

cout << num << " is not a power of two." << endl;

}

return 0;

}

**5. Rotate a Matrix by 90 Degrees**

**Problem:** Rotate an N×NN \times NN×N matrix 90 degrees clockwise.

cpp

#include <iostream>

#include <vector>

using namespace std;

void rotateMatrix(vector<vector<int>>& matrix) {

int n = matrix.size();

// Transpose the matrix

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

for (int j = i; j < n; ++j) {

swap(matrix[i][j], matrix[j][i]);

}

}

// Reverse each row

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

reverse(matrix[i].begin(), matrix[i].end());

}

}

void printMatrix(const vector<vector<int>>& matrix) {

for (const auto& row : matrix) {

for (int elem : row) {

cout << elem << " ";

}

cout << endl;

}

}

int main() {

vector<vector<int>> matrix = {

{1, 2, 3},

{4, 5, 6},

{7, 8, 9}

};

rotateMatrix(matrix);

cout << "Rotated Matrix:" << endl;

printMatrix(matrix);

return 0;

}