**Lab 05 Data Structures Lab**

**Date: 26 Aug 2024**

1. Implementation of Stack using arrays   
2. To check if the given parenthesized expression has properly matching open   
 and closing parenthesis  
3. To check a given string is palindrome or not using stack  
**CODE:**

#include <iostream>

#include <string.h>

using namespace std;

class Stack {

int top;

int max\_size;

char\* arr;

public:

Stack(int size) {

max\_size = size;

arr = new char[max\_size];

top = -1;

}

void push(char x) {

if (top >= max\_size - 1) {

cout << "Stack Overflow\n";

} else {

arr[++top] = x;

}

}

char pop() {

if (top < 0) {

cout << "Stack Underflow\n";

return '\0';

} else {

char x = arr[top--];

return x;

}

}

bool isEmpty() {

return (top < 0);

}

~Stack() {

delete[] arr;

}

};

bool isParenthesisMatching(const string& expr) {

Stack stack(expr.length());

for (char ch : expr) {

if (ch == '(' || ch == '{' || ch == '[') {

stack.push(ch);

} else if (ch == ')' || ch == '}' || ch == ']') {

// Pop matching opening bracket from stack

if (stack.isEmpty()) return false;

char top = stack.pop();

if ((ch == ')' && top != '(') ||

(ch == '}' && top != '{') ||

(ch == ']' && top != '[')) {

return false;

}

}

}

return stack.isEmpty();

};

bool isPalindrome(const char\* str) {

int length = strlen(str);

Stack stack(length / 2);

for (int i = 0; i < length / 2; i++) {

stack.push(str[i]);

}

int startIndex = (length % 2 == 0) ? length / 2 : (length / 2) + 1;

for (int i = startIndex; i < length; i++) {

if (str[i] != stack.pop()) {

return false;

}

}

return true;

}

int main() {

string expr;

cout << "Enter an expression: ";

cin >> expr;

if (isParenthesisMatching(expr)) {

cout << "Parentheses are balanced.\n";

} else {

cout << "Parentheses are not balanced.\n";

}

const char\* str = "madam";

if (isPalindrome(str)) {

cout << str << " is a palindrome\n";

} else {

cout << str << " is not a palindrome\n";

}

return 0;

}

**OUTPUT:**

Enter an expression: [({){}}[]]

Parentheses are not balanced.

madam is a palindrome