Name	Roll No	LAB No .	Date
AROOKHA SALEEM	5121323001	06	22/12/2024

Submitted to

Sir Hurraira

Subject

DSA(LAB)

Department

BS SOFTWARE ENGINEERING (3^{RD} SEMESTER)

LAB TASK

1.Two stacks of the same type are the same if they have the same number of elements and their elements at the corresponding positions are the same. Write a template based function isStacksSame, which takes two stacks as input, compares them and return, true if both stack are equal, false otherwise.

2. Write a template-based function, reverseStack, which takes a stack object as input parameter and uses a queue object to reverse the elements of the stack.

3. Write a C++ function to reverse a string using a stack. Your task is to implement the function with the following prototype: std::string reverse String (const std::string& input); The function should take a string as input and return a new string that is the reverse of the input string using a stack. For example, if the input string is "Hello", the function should return "olleH".

4. Write a function evaluate Postfix that evaluates a given postfix expression. The function should take a string representing the postfix expression as input and return the result of the expression. You should use a stack to implement the algorithm.

5. A palindrome is a word, phrase, number, or other sequence of symbols or elements, whose meaning may be interpreted the same way in either forward or reverse direction. Famous examples include “mom”, “dad”, “123454321” “Amore, Roma”, “A man, a plan, a canal: Panama” and “No ‘x’ in ‘Nixon’ ”, etc. Let us try to use stack to evaluate whether a given string is palindrome or not. Write a function isPalindrome, which takes a string as input and returns a Boolean value describing that given string was palindrome or not. (Note: Stack must be used in this algorithm.)

TASK 1:-

```
#include <stack>
#include <iostream>

using namespace std;

template<typename T>
bool isStacksSame(stack<T> stack1, stack<T> stack2) {
   if (stack1.size() != stack2.size()) {
      return false;
   }

   while (!stack1.empty()) {
      if (stack1.top() != stack2.top()) {
         return false;
      }
      stack1.pop();
      stack2.pop();
   }

   return true;
}

int main() {
   stack<int> stack1, stack2;
   stack1.push(1); stack1.push(2); stack1.push(3);
```

```
    stack2.push(1); stack2.push(2); stack2.push(3);

    if (isStacksSame(stack1, stack2)) {
        cout << "Stacks are same" << endl;
    } else {
        cout << "Stacks are not same" << endl;
    }

    return 0;
}
```

OUTPUT

```
Stacks are same
```

TASK 2:-

```
#include <stack>
#include <queue>
#include <iostream>

using namespace std;

template<typename T>
void reverseStack(stack<T>& stack)
{
    queue<T> queue;

    while (!stack.empty())
    {
        queue.push(stack.top());
        stack.pop();
    }

    while (!queue.empty())
    {
        stack.push(queue.front());
        queue.pop();
    }
}

int main()
 {
    stack<int> myStack;
    myStack.push(1); myStack.push(2); myStack.push(3);

    reverseStack(myStack);

    while (!myStack.empty()) {
        cout << myStack.top() << " ";
```

```
        myStack.pop();
    }

    return 0;
}
```

OUTPUT

TASK 3:-

```
#include <stack>

#include <string>

#include <iostream>

using namespace std;

string reverseString(const string& input)
{
    stack<char> stack;

    for (char ch : input)
    {
        stack.push(ch);
    }

    string reversed;
    while (!stack.empty())
    {
        reversed += stack.top();
        stack.pop();
    }

    return reversed;
}

int main()
{
    string input = "Hello";
    string reversed = reverseString(input);

    cout << "Original: " << input << endl;
    cout << "Reversed: " << reversed << endl;
```

```
    return 0;
}
```

OUTPUT

```
Original: Hello
Reversed: olleH
```

TASK 4:-

```
#include <stack>
#include <string>
#include <sstream>
#include <iostream>

using namespace std;

int evaluatePostfix(const string& expression) {
    stack<int> stack;
    istringstream tokens(expression);
    string token;
    while (tokens >> token)
    {
        if (isdigit(token[0]) || (token.size() > 1 && isdigit(token[1])))
{
            stack.push(stoi(token));
        } else {
            int val2 = stack.top(); stack.pop();
            int val1 = stack.top(); stack.pop();
            switch (token[0]) {
                case '+': stack.push(val1 + val2); break;
                case '-': stack.push(val1 - val2); break;
                case '*': stack.push(val1 * val2); break;
                case '/': stack.push(val1 / val2); break;
            }
        }
    }
    return stack.top();
}

int main() {
    string expression = "3 4 + 2 * 7 /";
    int result = evaluatePostfix(expression);

    cout << "Result of postfix evaluation: " << result << endl;
```

```
    return 0;
}
```

OUTPUT

```
Result of postfix evaluation: 2
```

TASK 5:-

```
#include <stack>
#include <string>
#include <iostream>
#include <cctype>

using namespace std;

bool isPalindrome(const string& input) {
    stack<char> stack;
    string cleanInput;

    for (char ch : input) {
        if (isalnum(ch)) {
            cleanInput += tolower(ch);
        }
    }

    for (char ch : cleanInput) {
        stack.push(ch);
    }

    for (char ch : cleanInput) {
        if (ch != stack.top()) {
            return false;
        }
        stack.pop();
    }

    return true;
}

int main() {
    string input = "A man, a plan, a canal: Panama";
    bool result = isPalindrome(input);
    cout << "Is the string a palindrome? " << (result ? "Yes" : "No") << endl;
```

```
    return 0;
}
```

OUTPUT