# Department of Computing

# School of Electrical Engineering and Computer Science

**CS-250: Data Structure and Algorithms**

**Class: BESE 13AB**

# 

# Lab 4: Implmentation of Stacks in different problems

**Date: 13th October, 2023**

**Time: 10 am - 1 pm & 2:30 pm – 5pm**

**Name:** Aimen Munawar

**Class:** BESE-13-A

**CMS ID:** 415867

# Lab Instructor: Anum Asif

# 

# Lab 4: Implmentation of Stacks in different problems

**Introduction**

This lab consists of stacks implementation and some of its applications.

**Objectives**

Objective of this lab is to enable students to build stack ADT using linked list and arrays, perform the following tasks on it and analyze the performance of each implementation.

**Tools/Software Requirement**

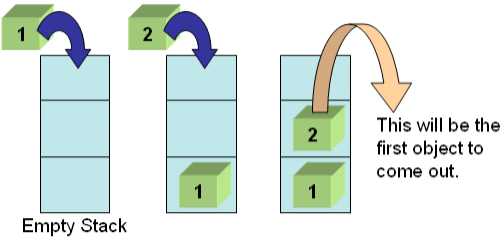
Visual Studio c++

**Helping Material**

Lecture slides, text book

**Description**

A stack meant to mimic the information storage and retrieval in LIFO (Last In First Out) order.



**Stack Operations**

1. void Push(element) – pushes an element on the top of stack
2. element Pop() – removes and display the element on the top of stack
3. boolisEmpty() – checks if the stack is empty or not
4. boolisFull() – checks if the stack is full or not
5. void Clear() – release the memory allocated by stack
6. void Peak() – display the contents of the top element of stack

**Lab Tasks**

**Task 1:**

The idea is rather simple: You keep a Stack of braces, and every time you encounter an open brace, you push it into your stack. Every time you encounter a close brace, you pop the top element from your stack. At the end, you check your stack for being empty. If so, indeed your input string contained balanced braces. Otherwise, it didn't.

**Expected Input**

1. 1 + 2 \* (3 / 4)
2. 1 + 2 \* [3 \* 3 + {4 – 5 (6 (7/8/9) + 10) – 11 + (12\*8)] + 14
3. 1 + 2 \* [3 \* 3 + {4 – 5 (6 (7/8/9) + 10)} – 11 + (12\*8) / {13 +13}] + 14

Your program will determine whether the open brackets (the square brackets, curly braces and the parentheses) are closed in the correct order.

**Expected Output**

1. This expression is correct.
2. This expression is NOT correct e.g. error at character # 10. ‘{‘- not closed.
3. This expression is correct.

Your program should be able to take generic input expression from user

Solve the above problem using an **array based stack**

**Important Note:** Practice your knowledge of OOP with C++ when creating a solution.

**Solution:**

|  |
| --- |
| Solution |
| Task 1 Code:  #include <iostream>  #include <stack>  #include <string>  using namespace std;  bool valid\_paren(string input)  {  stack <char> s;  for (char c : input) {  if (c == '(' || c == '{' || c == '['){  s.push(c);  }  else if ((c == ')') ||  (c == '}' ) ||  (c == ']' )){  if (s.empty() )  {  return false;  }  char top = s.top();  s.pop();  if ((c == ')' && top != '(') || (c == '}' && top != '{') || (c == ']' && top != '[')){  return false;  }  }  }  return s.empty();  }  int main() {  string input;  cout << "Enter a string with parentheses: ";  getline(cin, input);  if (valid\_paren(input)) {  cout << "Valid parentheses." << endl;  }  else {  cout << "Invalid parentheses." << endl;  }  return 0;  }  Task 1 Output Screenshot: |
|  |

### Deliverables

Compile a single word document by filling in the solution part and submit this Word file on LMS. You must show the implementation of the tasks in the designing tool, along with your complete Word document to get your work graded. You must also submit this Word document on the LMS. In case of any problems with submissions on LMS, submit your Lab assignments by emailing it to [anum.asif@seecs.edu.pk](mailto:anum.asif@seecs.edu.pk).