SVKM's NMIMS
Mukesh Patel School of Technology Management & Engineering
Computer Engineering Department
Program: B.Tech Integrated Sem V

Course: Basic Data Structures
2024-2025
Experiment No.04

A.1 Aim:

To study and implement concept of Stack data structure.

A.2 Prerequisite:

1. Knowledge of different operations performed on Stack data structure
2. Fundamental concepts of C\C++.

A.3 Outcome:

After successful completion of this experiment students will be able to

1. Identify the need of appropriate selection of data structure
2. Identify the steps of stack data structure selection.
3. Implement stack data structure to solve the given problem
4. Enlist the applications of stack data structure.

A.4 Theory:

A.4.1. Introduction of Stack

Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO (Last in first out) or FILO (First in last out).

The functions performed on stack are:

1. **Push:** Adds an item in the stack. If the stack is full, then it is said to be an overflow condition.
2. **Pop:** Removes an item from the stack. The items are popped in the reversed order in which they are pushed. If the stack is empty, then it is said to be an underflow condition.
3. **Peek:** Returns the top most element from the stack.

Applications of Stack:

1. Balancing of symbols

2. Infix to postfix\prefix conversion
3. Redo-undo features at many places like editors, Photoshop
4. Forward and backward feature in web browsers

A.5 Procedure/Algorithm:

A.5.1:

TASK 1:

Write a program to implement ADT of Stack.

TASK 2:

The organization structure of "ABC Pvt. LTD" is given as per joining order. The company is in crisis and wants to start lay off. The employee who joined last is fired first. Implement the suitable data structure and display the order in which the company will fire employees.

Employee id	Order of joining	
101	1	(joined first)
102	2	
107	3	
108	4	
111	5	
112	6	(joined last)

TASK 3: Checking for Balanced Braces in a String

Here is an example of balance braces: {()[]{}}

A useful application of stacks is exactly that: to check for balanced braces in an input string.

The idea is:

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.

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)

Roll No.	Name:
Class :	Batch :
Date of Experiment:	Date of Submission
Grade :	Time of Submission:
Date of Grading:	

B.1 Software Code written by student: (Task 1)

Task2:

B.2 Input and Output: (Task 1)

(Paste your program input and output in following format, If there is error then paste the specific error in the output part. In case of error with due permission of the faculty extension can be given to submit the error free code with output in due course of time. Students will be graded accordingly.)

Task1:

Task2:

B.3 Observations and learning [w.r.t. all tasks]:

(Students are expected to comment on the output obtained with clear observations and learning for each task/ sub part assigned)

B.4 Conclusion:

(Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.3)

B.5 Question of Curiosity

(To be answered by student based on the practical performed and learning/observations)

Checking for palindrome string

Suppose characters are arriving on a stream reader. Suggest an algorithm to see if the string forms a palindrome. Capitalization, spacing, and punctuation are ignored.

****