UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 8

Stacks

Submitted by:
Sumel, Hendrix Nathan L..

Instructor:
Engr. Maria Rizette H. Sayo

November 8, 2025

I. Objectives

Introduction

A stack is a collection of objects that are inserted and removed according to the last-in, first-out (LIFO) principle.

A user may insert objects into a stack at any time, but may only access or remove the most recently inserted object that remains (at the so-called “top” of the stack)

This laboratory activity aims to implement the principles and techniques in:

- Writing Python program using Stack
- Writing a Python program that will implement Stack operations

II. Methods

Instruction: Type the python codes below in your Colab. After running your codes, answer the questions below.

```
# Stack implementation in python


# Creating a stack
def create_stack():
    stack = []
    return stack


# Creating an empty stack
def is_empty(stack):
    return len(stack) == 0

# Adding items into the stack
def push(stack, item):
    stack.append(item)
    print("Pushed Element: " + item)

# Removing an element from the stack
def pop(stack):
    if (is_empty(stack)):
        return "The stack is empty"
    return stack.pop()

stack = create_stack()
push(stack, str(1))
push(stack, str(2))
push(stack, str(3))
push(stack, str(4))
push(stack, str(5))

print("The elements in the stack are:"+ str(stack))
```

Answer the following questions:

1 Upon typing the codes, what is the name of the abstract data type? How is it implemented?
2 What is the output of the codes?
3 If you want to type additional codes, what will be the statement to pop 3 elements from the top of the stack?
4 If you will revise the codes, what will be the statement to determine the length of the stack? (Note: You may add additional methods to count the no. of elements in the stack)

1. Upon typing the codes, what is the name of the abstract data type? How is it implemented?

- The abstract data type is a **Stack**. It's implemented using a Python list, where elements are added (pushed) and removed (popped) from the same end (LIFO - Last In, First Out principle).

2. What is the output of the codes?

- The output shows all the push operations and then displays the final stack contents:

```
Pushed Element: 1
Pushed Element: 2
Pushed Element: 3
Pushed Element: 4
Pushed Element: 5
The elements in the stack are:['1', '2', '3', '4', '5']
```

3. If you want to type additional codes, what will be the statement to pop 3 elements from the top of the stack?

```
# Pop 3 elements from the top of the stack
print("Popped Element:", pop(stack))
print("Popped Element:", pop(stack))
print("Popped Element:", pop(stack))
```

4. If you revise the codes, what will be the statement to determine the length of the stack?

- Method to get the length of the stack

```
def stack_length(stack):
    return len(stack)
```

- Or you can simply use:
  length = len(stack)

III. Results

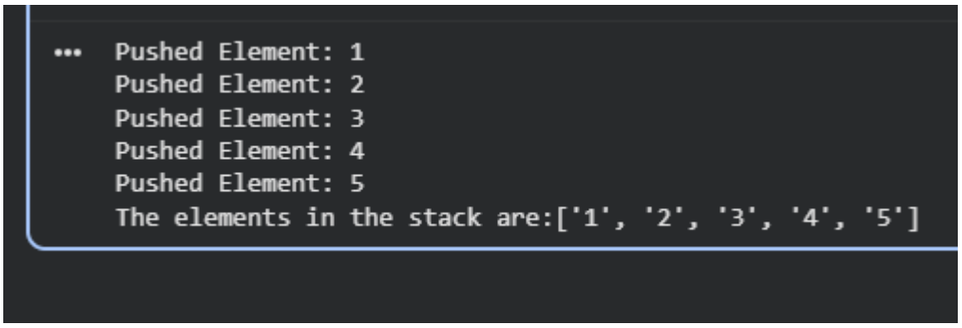

Figure 1 Screenshot of program of output. lab report 8 (SUMEL) - Colab

This output confirms that all five elements were successfully added to the stack in the correct order, with '5' being at the top and ready to be removed if a pop operation is performed.

IV. Conclusion

Based on the successful execution of the Python code, we conclude that the stack abstract data type was effectively implemented using a list structure, adhering to the LIFO (Last-In, First-Out) principle, as evidenced by the sequential output showing elements 1 through 5 being pushed and stored in the order ['1', '2', '3', '4', '5'] where '5' resides at the top, thereby demonstrating fundamental stack operations that can be extended with additional functionality like popping elements or determining stack length for broader applications in computer science.

References

[1] Python Software Foundation. "Data Structures," Python Official Documentation, 2023.

[2] GeeksforGeeks. "Stack Data Structure," GeeksforGeeks Algorithms Portal, 2023.

[3] Google Colab. "Python Execution Environment," Google Colab Documentation, 2023.

[4] Sedgewick, R., & Wayne, K. "Algorithms," Addison-Wesley Professional, 2011.