UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 8

Stacks

Submitted by:
Directo, Hannah Thea B.

Instructor:
Engr. Maria Rizette H. Sayo

October 4, 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)

III. Results

1. The abstract data type is called Stacks. It is implemented in Python using list, where elements are being added(pushed) and removed (popped) following the Last-In-First-Out (LIFO) principle of stacks.

2. The output shows:

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. def pop(stack):

```
If (is_empty(stack)):
        return "The stack is empty"
return stack.pop()
print("Popped Element: " + item)
```

To pop 3 elements from the top of the stack, first, I add this statement “print("Popped Element: " + item)” to the pop method to use it as simple like this.

```
#remove item
pop(stack)
pop(stack)
pop(stack)
```

So, the output after using the pop method show:

Pushed Element: 1

Pushed Element: 2

Pushed Element: 3

Pushed Element: 4

Pushed Element: 5

The elements in the stack are: ['1', '2']

4. If I revise the code to determine the length of the stack, this statement will be added to the code:

```
def length(stack):
    if (is_empty(stack)):
        return "The stack is empty"
    return len(stack)
```

IV. Conclusion

In this laboratory activity, I learned how a stack works by following the Last-In-First-Out (LIFO) rule. I was able to perform basic operations such as pushing elements into the stack and popping them out. I also revised the code to count the number of elements, which made it easier to check the size of the stack. Through this, I understood how stacks can organize and manage data in a simple way.