University Of Caloocan City
Computer Engineering Department

Data Structure and Algorithm

Laboratory Activity No. 8

Stacks

Submitted by:
Bron, Jhustine A.

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

Answers:

1. The abstrack data type that is used in this code is **Stack**. It is implemented using append, pop, and push.

2. The output of the code is;

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. To pop 3 elements from the top of the stack:

```
print("\nPopping elements\n")
print("Popped Element: ", pop(stack))
print("Popped Element: ", pop(stack))
print("Popped Element: ", pop(stack))
print("The elements in the stack are:"+ str(stack))
```

Output:

```
Popping elements

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

4. Statement tor code to determine the length of the stack:

```
print("Length of the stack: ", len(stack))
```

Output:

```
Length of the stack:  5
```

Final Code With Revision:

```
def create_stack():
    stack = []
    return stack

def is_empty(stack):
    return len(stack) == 0

def push(stack, item):
```

```
    stack.append(item)
    print("Pushed Element: " + item)

def pop(stack):
    if (is_empty(stack)):
        return "The stack is empty"
    return stack.pop()

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

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

print("Length of the stack: ", len(stack))
print("The elements in the stack are:"+ str(stack))

print("\nPopping elements\n")
print("Popped Element: ", pop(stack))
print("Popped Element: ", pop(stack))
print("Popped Element: ", pop(stack))
print("The elements in the stack are:"+ str(stack))
```

VI. Conclusion

In conclusion, this activity helped us understand how a stack works as an abstract data type and how it can be implemented using Python lists. By using push, pop, and checking the size of the stack, we saw how elements are managed in a Last-In, First-Out (LIFO) manner. The outputs confirmed that the stack correctly adds and removes items, and by popping multiple elements, we can see how the stack shrinks. Overall, this exercise gave us a clearer idea of stack operations and how they can be applied in solving programming problems.