UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 8

# Stacks

*Submitted by:*
Uy, Junichiro H.

*Instructor:*
Engr. Maria Rizette H. Sayo

October 04, 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 used is STACK. It was implemented using the push and pop functions, and most importantly, a createStack() function:

Create the STACK:

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

Create the PUSH function:

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

Create the POP function:

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

Then we INITIALIZE the STACK variable:

```
stack = create_stack()
```

2. The output of the code provided above 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 on top of the stack list, I used a for loop in range of 3:

```
def pop(stack):
  if(isEmpty(stack)):
    return"Stack is Empty."

  for x in range(3):
    stack.pop()
```

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

4. To determine the length of the stack, I made a function using len():

```
def checkLen(stack):
  print(f"The stack is {len(stack)} elements long.")
```

Output:

```
The elements in the stack are: ['1', '2', '3', '4', '5']
The stack is 5 elements long. # This is the output
The elements in the stack are: ['1', '2']
The stack is 2 elements long. # This is the output
```

Lastly, I also made a function that prints the stack as str:

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

Link to the Google Colab: https://colab.research.google.com/drive/1OeZxejCoLbYvdSrp9ztyGZWiZd-lWLUl#scrollTo=c-6WKYdv847t&line=23&uniqifier=1

# IV. Conclusion

This lab activity provided me with knowledge on how to make STACK. I can use this if I want to make applications needing this in the future, or if I want to make an undo and redo button. Also, this activity made me realize that I can use a different function inside a function, this opens a lot of opportunity to explore python.

# References

[1] Co Arthur O.. “University of Caloocan City Computer Engineering Department Honor Code,” UCC-CpE Departmental Policies, 2020.