



## Data Structure and Algorithm

### Laboratory Activity No. 8

---

# Stacks

---

*Submitted by:*  
Ruperto, April Anne 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 name of the abstract data type is called stacks. It is a collection of elements that follows the Last-In-Fist-Out (LIFO) principle. The elements can only be inserted and removed at the top of the stack using push(element) and pop().

2. The output of the code is:

Pushed Element: 1

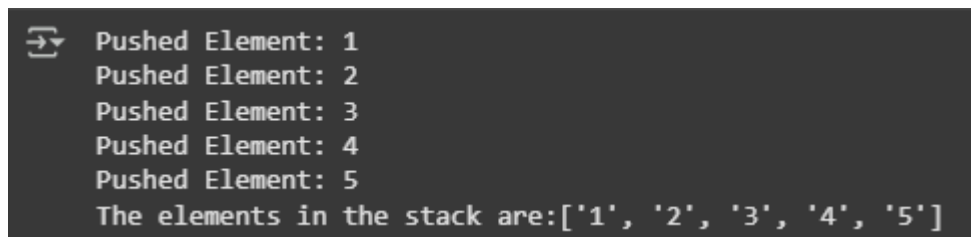
Pushed Element: 2

Pushed Element: 3

Pushed Element: 4

Pushed Element: 5

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



```
➦ 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']
```

Figure 1: Output of the Program

3. Additional code:

# Popping 3 elements from the stack

pop(stack)

pop(stack)

pop(stack)

print("The elements in the stack after popping 3 elements: " + str(stack))

```
# 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))

# Popping 3 elements from the stack
pop(stack)
pop(stack)
pop(stack)

print("The elements in the stack after popping 3 elements: " + str(stack))
```

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']  
The elements in the stack after popping 3 elements: ['1', '2']

Figure 2: Screenshot of Program

4. Additional code:

```
# Method to get the size of the stack
def size(stack):
    return len(stack)

print("The size of the stack is: " + str(size(stack)))
```

```
# 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()

# Method to get the size of the stack
def size(stack):
    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("The elements in the stack are:" + str(stack))
print("The size of the stack is: " + str(size(stack)))
```

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']  
The size of the stack is: 5

## IV. Conclusion

Stacks are fundamental data structure that operates on a Last-In-Fist-Out (LIFO) principle. In python, stacks can be efficiently implemented using lists, where elements can be added and removed at the end. Stacks are essential part of computer programming. They're simple yet powerful for many tasks involving orderly data management.

## References

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