



UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT



Data Structure and Algorithm

Laboratory Activity No. 8

Stacks

Submitted by:
Nerio, Hannah Grace A.

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))
```

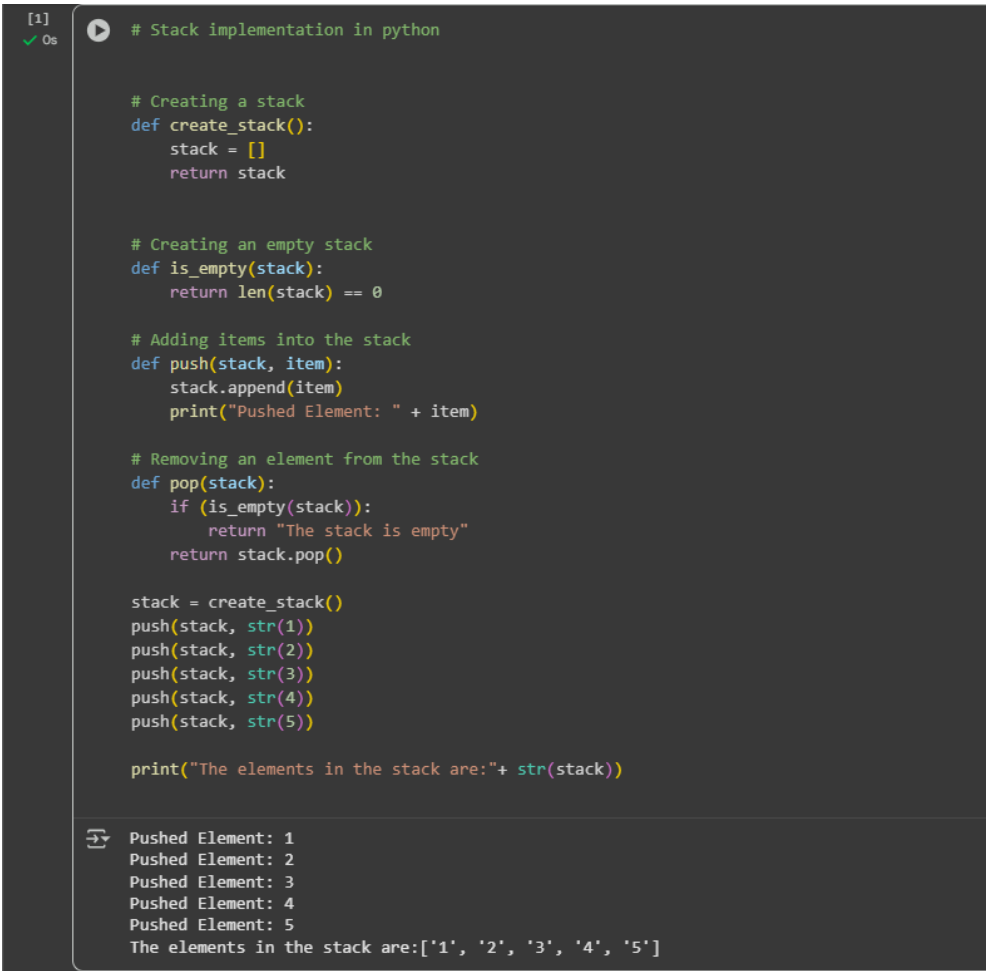
```
push(stack, str(5))
```

```
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] ✓ 0s # 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))
```

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']

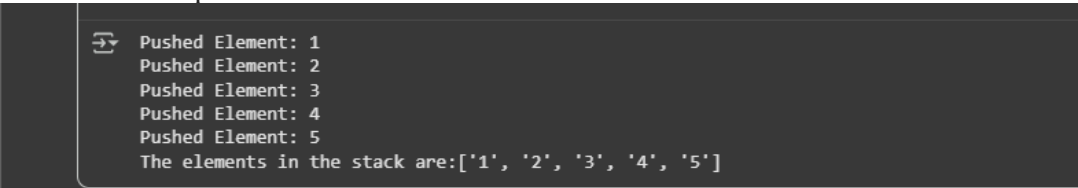
Figure 1 Screenshot of program

Answer the following questions:

- 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 is implemented using a python list, where `append ()` is used to push elements and `pop ()` is used to remove elements (LIFO: Last In First Out principle)

- 2 What is the output of the codes?



```
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?

```
print('pop ' + pop(stack))
print('pop ' + pop(stack))
print('pop ' + pop(stack))

print("Stack after elements are popped:")
print(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)

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

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

IV. Conclusion

In this laboratory activity, the task is to implement a stack using Python lists to better understand how the stack abstract data type works. This activity showed how the push operation adds elements on top of the stack and how the pop operation removes the most recent elements following the Last-In, First-Out (LIFO) principle. After pushing five elements and then popping three from the top, only the first two elements remained in the stack, which confirms that the operations worked as expected. This activity helped reinforce the concept of stacks and demonstrated how Python can be used to simulate abstract data structures effectively.

References

- [1] Co Arthur O.. "University of Caloocan City Computer Engineering Department Honor Code," UCC-CpE Departmental Policies, 2020.
- [2] Google, "Google Colaboratory," *Google Colab*. [Online]. Available: <https://colab.research.google.com>. [Accessed: 4-Oct-2025].