UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 8

Stacks

Submitted by:
Talagtag Mark Angel T.

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:

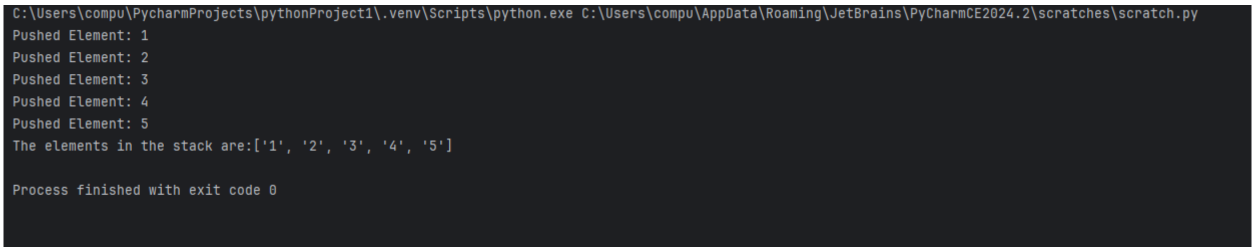
1 Upon typing the codes, what is the name of the abstract data type? How is it implemented?
2 What is the output of the codes?
3 If you want to type additional codes, what will be the statement to pop 3 elements from the top of the stack?
4 If you will revise the codes, what will be the statement to 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 (ADT) in the given code is **Stack**

A stack is implemented using a **list** in Python. In this implementation, we have the following functions:

- create_stack(): Initializes an empty stack (using a list).
- is_empty(stack): Checks if the stack is empty.
- push(stack, item): Adds an item to the top of the stack.
- pop(stack): Removes the item from the top of the stack (if the stack is not empty).



```
C:\Users\compu\PycharmProjects\pythonProject1\.venv\Scripts\python.exe C:\Users\compu\AppData\Roaming\JetBrains\PyCharmCE2024.2\scratches\scratch.py
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']

Process finished with exit code 0
```

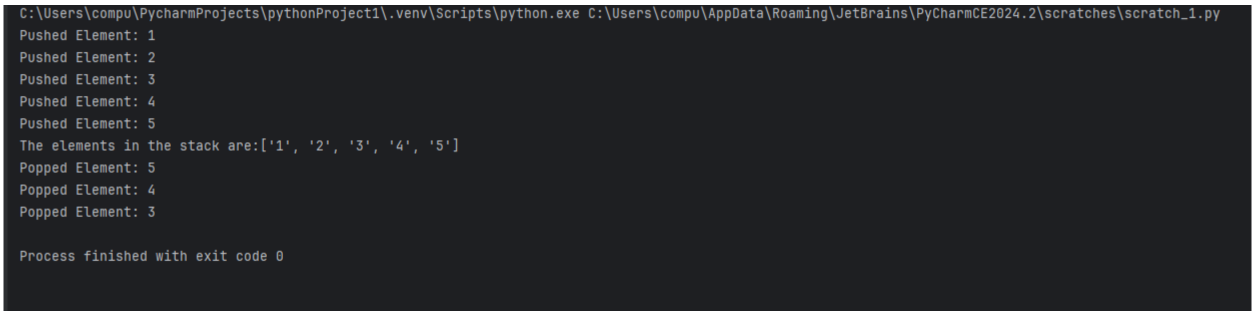
2. The code pushes five elements onto the stack and then prints the stack. Let's go through the output:

- 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.

Using a loop:

```
for _ in range(3):
    print("Popped Element: " + pop(stack))
```

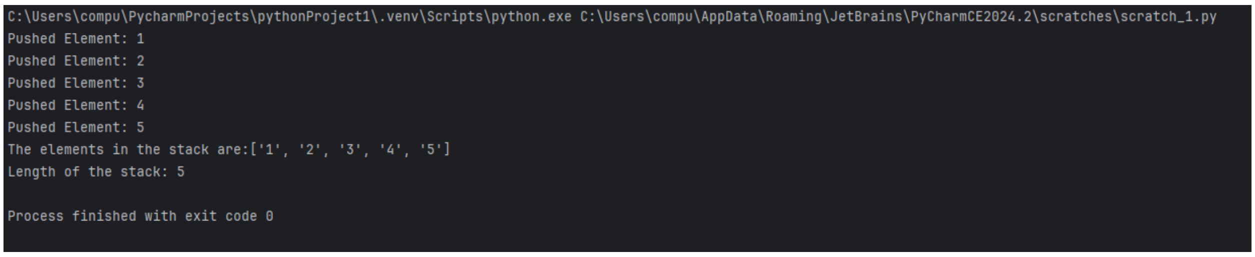


```
C:\Users\compu\PycharmProjects\pythonProject1\.venv\Scripts\python.exe C:\Users\compu\AppData\Roaming\JetBrains\PyCharmCE2024.2\scratches\scratch_1.py
Pushed Element: 1
Pushed Element: 2
Pushed Element: 3
Pushed Element: 4
Pushed Element: 5
The elements in the stack are:['1', '2', '3', '4', '5']
Popped Element: 5
Popped Element: 4
Popped Element: 3

Process finished with exit code 0
```

4.

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



```
C:\Users\compu\PycharmProjects\pythonProject1\.venv\Scripts\python.exe C:\Users\compu\AppData\Roaming\JetBrains\PyCharmCE2024.2\scratches\scratch_1.py
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']
Length of the stack: 5

Process finished with exit code 0
```

Figure 1 Screenshot of program

If an image is taken from another literature or intellectual property, please cite them accordingly in the caption. Always keep in mind the Honor Code [1] of our course to prevent failure due to academic dishonesty.

IV. Conclusion

In conclusion I gained a clear understanding of how to work with the **Stack** abstract data type using Python's built-in **list** data structure. Here's a summary of the key concepts learned

References

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