AIO25 - M1W3

GRID137

June 16, 2025

Contents

List of Figures

List of Tables

Chapter 1

Classes and Objects

Chapter 2

OOP with Python (Custom Pytorch Class)

Chapter 3

Database - SQL(3)

Chapter 4

OOP + Data structure (Graph and Tree)

4.1 Stack and Queue

Stack and Queue definition

Stack and Queue are special uses of List in Python, with specific constraints:

- Stack: only add/remove element from one end (LIFO).
- Queue: Add at one end, remove from the other one (FIFO).

4.1.1 Stack

Stack Visualization

Stacks return elements in the reverse order in which they are stored; that is, the most recent element to be added is returned. We call this kind of data structure last-in-first-out (LIFO).

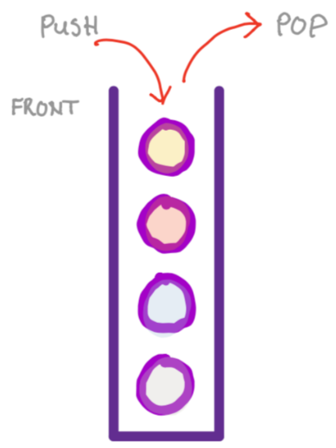

Figure 4.1: Stack visualized

Push/Pop Visualization

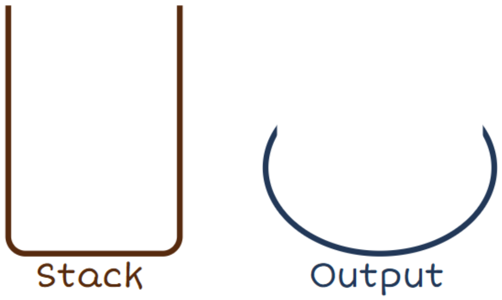

Figure 4.2: Create an Empty stack

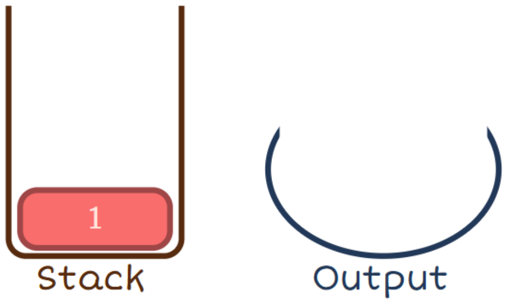

Figure 4.3: Push in the first element

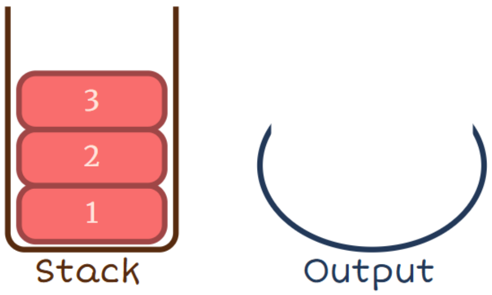

Figure 4.4: Push in the second and third element

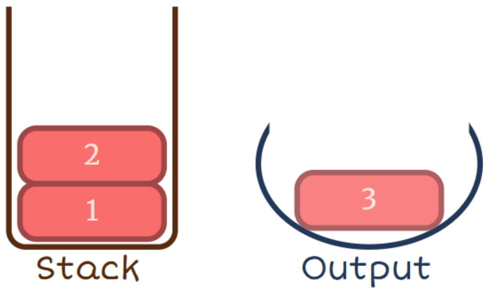

Figure 4.5: Pop out the element at the top of the stack

4.1.2 Coding

```
class MyStack:
  def _init_(self, capacity):
    self._capacity = capacity
    self._stack = []

  def push(self, value):
    if self.is_full():
      print('Do nothing!')
    else:
      self._stack.append(value)

  def pop(self):
    if self.is_empty():
      print('Do nothing')
      return None
    else:
      return self._stack.pop()

  def print(self):
    print(self._stack)

  def is_full(self):
    return len(self._stack) == self._capacity
```

Listing 4.1: Define Stack data structure class

```
stack1 = MyStack(5)
stack1.push(12)
stack1.push(8)
stack1.push(21)
stack1.push(33)
stack1.push(34)
stack1.push(35)
stack1.print()

//Output: Do nothing!
```

```
        [12, 8, 21, 33, 34]
```

Listing 4.2: Push and Pop example

Chapter 5

Unix and Docker