Parvatibai Chowgule College of Arts and Science
Autonomous
Accredited by NAAC with Grade 'A' (CGPA Score 3.41 on a 4 Point Scale in 3rd cycle)
Best affiliated College-Goa University Silver Jubilee Year Award

Department of Computer Science

2020 - 2021

M.Sc-IT (Part 1)

Course: `Data Structures and Algorithms`

Course Code: `MIT11`

Name: `Alexander Roque Rodrigues`
Roll Number: `202202`

Index

List of Source Codes

List of Algorithms

List of Figures

1 Minimum Element from Stack.

Sr.No: `1`

Date: `21-09-2020`

1.1 Problem Statement

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Conditions Include:

$$0 \leq x \leq 2 \times 10^{20}$$

1.2 Algorithmic Approach

Algorithm 1: Stack Algorithm

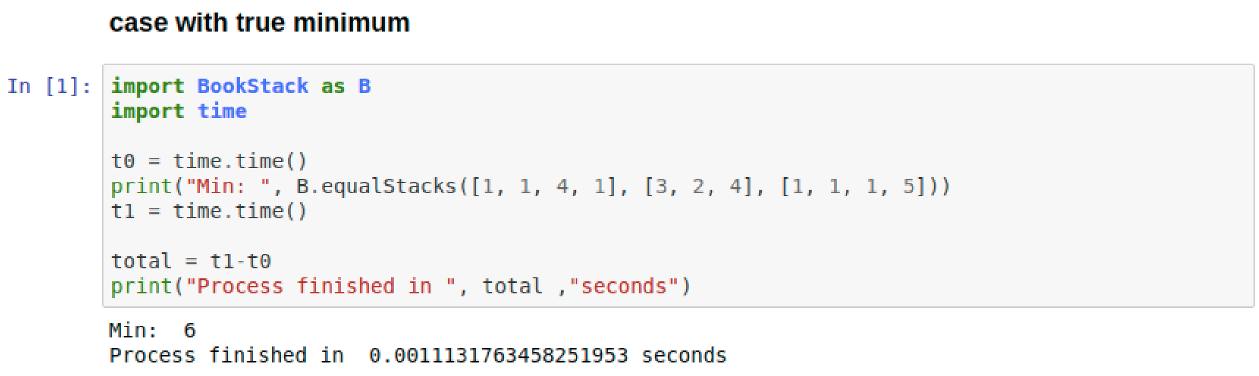
```
Result: Operations are carried out on the stack and the minimum value at any given
        time kept track of.
initialization;
while While condition do
    instructions;
    if condition then
        instructions1;
        instructions2;
    else
        instructions3;
    end
end
```

1.3 Source Code

```python
#!/usr/bin/env python3
from typing import Any
from collections import deque

class MinStack:

  # constructor function
  def __init__(self, factor: int):
    self.factor = factor
    self.s = deque()
    self.currentMinimum = None
    print("Factor: " ,self.factor)

  # push to stack
  def push(self, element: int) -> None:
    # if new stack with no elements
    if not self.s:
      self.s.append(element)
      self.currentMinimum = element

    # if greater than min, directly append to the top of the stack
    elif element > self.currentMinimum:
      self.s.append(element)

    # else is new minimum, append after calculations
    else:
      self.s.append(self.factor * element - self.currentMinimum)
      self.currentMinimum = element

  def pop(self) -> None:
    # no element on stack
    if not self.s:
      print("Under Flow Occured.")

    # top is the smallest
    if self.s[-1] < self.currentMinimum:
      # update with new minimum from the stack
      self.currentMinimum = self.factor * self.currentMinimum - self.s[-1]

    # invoke pop method after checking for new minimum value
    self.s.pop()

  def minimum(self):
    # return minimum
    return self.currentMinimum


'''
class stackInterfacer(MinStack):

    def __init__(self):
      # initialise parent class
        MinStack.__init__(self)

    # driver function
    def run(self):
        while(1):
            txt = input("> ")
            txt = txt.split()

            # push method
            if(txt[0]=="push"):
                s.push(int(txt[1]))

            # pop method
            elif(txt[0]=="pop"):
                s.pop()

            # peek method
            elif(txt[0]=="peek"):
                print(s.minimum())

            # print object
            elif(txt[0]=="print"):
                print(s.s)

            # size of dequeue object
            elif(txt[0]=="size"):
                print(len(s.s))

            # exit method
            elif(txt[0]=="exit"):
                print("bye")
                exit(0)

if __name__ == '__main__':

  # create interfacer class object and call driver function
  s = stackInterfacer()
  s.run()

'''
```

Listing 1: Python example

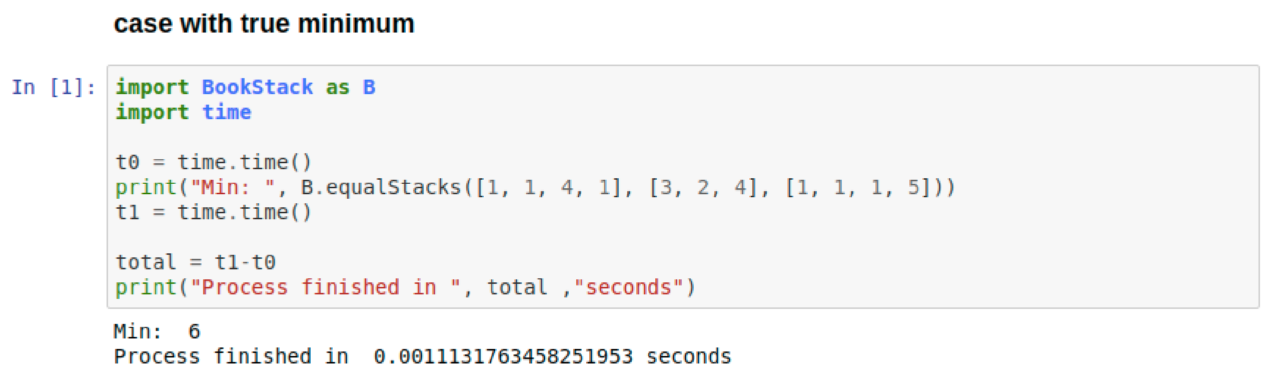
1.4 Output



case with true minimum

```
In [1]: import BookStack as B
        import time

        t0 = time.time()
        print("Min: ", B.equalStacks([1, 1, 4, 1], [3, 2, 4], [1, 1, 1, 5]))
        t1 = time.time()

        total = t1-t0
        print("Process finished in ", total ,"seconds")

Min:  6
Process finished in  0.0011131763458251953 seconds
```

Figure 1: Simple Test Case



case with true minimum

```
In [1]: import BookStack as B
        import time

        t0 = time.time()
        print("Min: ", B.equalStacks([1, 1, 4, 1], [3, 2, 4], [1, 1, 1, 5]))
        t1 = time.time()

        total = t1-t0
        print("Process finished in ", total ,"seconds")

Min:  6
Process finished in  0.0011131763458251953 seconds
```

Figure 2: Simple Test Case Duplicate

1.5 References

Ladd, S., Xin, Y., Yang, J., Liu, P., & Wu, L. (1998). Java suan fa = JAVA ALGORITHMS. Beijing: Dian Zi Gong ye Chu Ban She.