COMPUTER SCIENCE AND DA

Data Structures through Python

Stack

Lecture 03

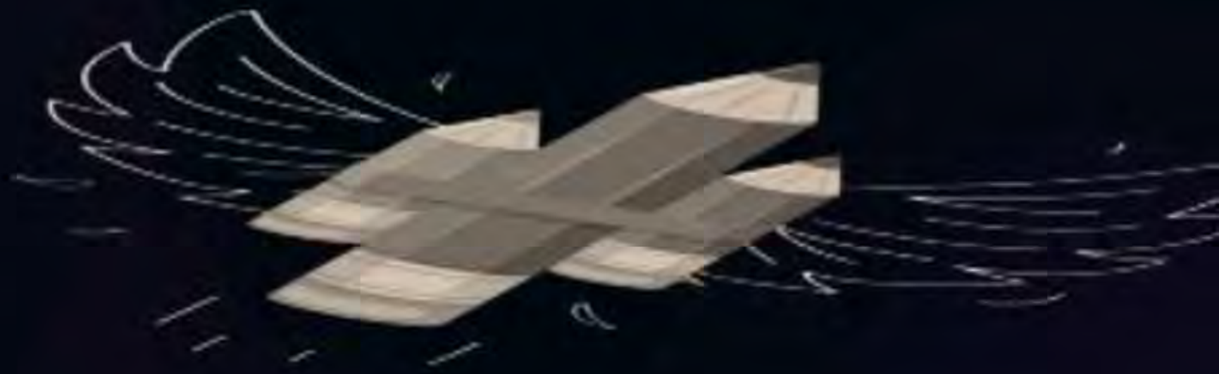
By- Kashif Sir

Topics to be Covered

Stack

STACK IMPLEMENTATION

- → Brute force approach
- → Using List
- → Using Deque
- → Using LifoQueue



Implementing stacks without inbuilt methods

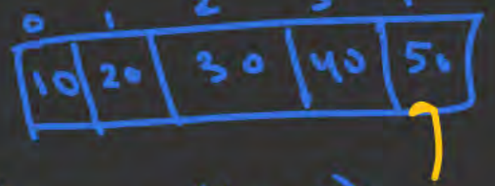
We have a stack S, using array/list with maxsize = 5.

maxsize = 5

Top = None

S = [ ]

Top = ~~None~~ 0




```
def push(item, S):
    if Top == maxsize-1:
        print("Stack overflow")
        return
    if Top == None:
        Top = 0
    else:
        Top = Top + 1
    S[Top] = item
```

push(10,S) ⇒ TOP = 0

S[0] = 10

push(20,S) ⇒ TOP = 1

S[1] = 20

push(30,S) ⇒ TOP = 2

S[2] = 30

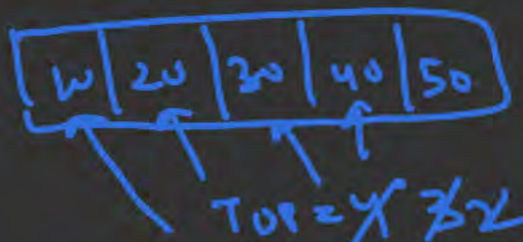
push(40,S) ⇒ TOP = 3

S[3] = 40

push(50,S)

⇒ TOP = 4

S[4] = 50




```
def pop(S):
    if TOP == None:
        print("Stack Underflow")
        return
    value = S[TOP]
    if TOP == 0:
        TOP = None
    else
        TOP = TOP - 1
    return value
```

pop(S) ⇒ value = 50

for a = pop(S) ⇒ value = 40

print(pop(S)) ⇒ 30

print(pop(S)) ⇒ 20

print(pop(S)) ⇒ 10

pop(S) Stack Underflow

```
def peek(s):
    if TOP == None:
        print("Stack Underflow")
        return
    return S[TOP]
```

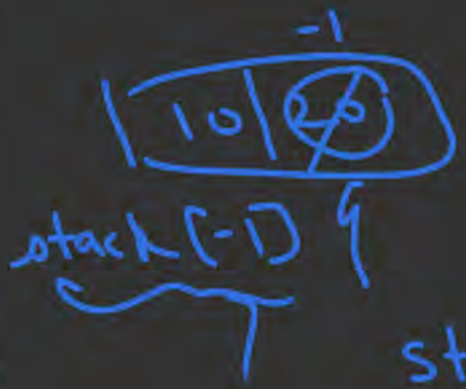
Stack Implementation using List

```
def create_stack( ):
    stack = []
    return stack
```

```
def push( item, S):
    S.append(item)
```

```
def empty_stack(S)
    return len(S) == 0
```

```
def pop(S):
    if empty_stack(S):
        print("Stack Underflow")
        return
```



val = S.pop( )

```
stack = create_stack( )
push(10, stack)
push(20, stack)
pop(S)
```

THANK - YOU