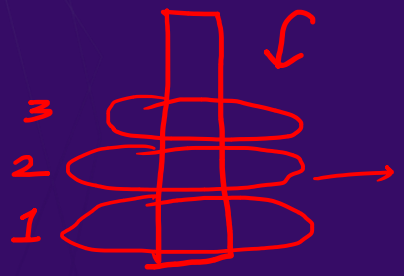
PW SKILLS

Stacks -1

Today's checklist

1. Introduction
2. Operations performed on stacks
3. Overflow
4. Underflow
5. Array implementation of a stack
6. Linked list implementation of a stack
7. Linked list vs Array implementation
8. STL for Stack

What is a Stack?



Can insert at only 1 position

Can access only topmost disk

To access any disk, I need to remove from top.

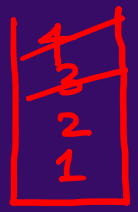
LIFO : Last-In-First-Out

Operations on Stack

1) Push : st.push(x) (To insert an element in stack)

2) Pop : st.pop() (To remove the topmost element).

3) Peek : st.peek()



st.push(1)
st.push(2)
st.push(3)
st.push(4)

st.pop() → 4

st.peek() → 3

Print : 1 2 3

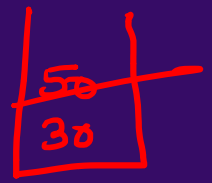
st.pop() → 3

Print : 1, 2

STL for Stack

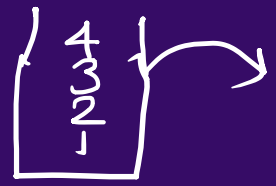
PW SKILLS

```
Stack <Data_Type> st = new Stack<>();

Stack <Integer> st = new Stack <>();

st.push (50);

st.push (30);
```




```
S.O.P (st.pop());   → 50
```

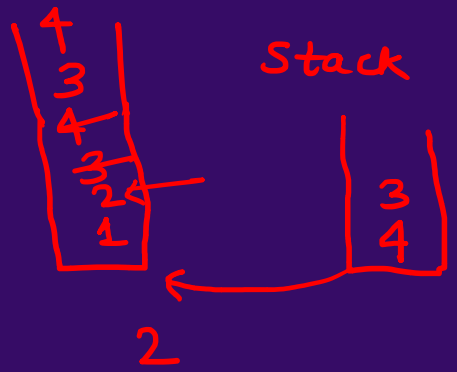
False ← st.isEmpty() → whether stack is empty / not



4, 3, 2

TC: O(N) → To access 1st element

Comparing Arrays, Linked Lists and Stacks



Get		T.C	S.C
Arrays	a[5]	O(1)	O(1)
Linked list		O(n)	O(1)
Stack		O(n)	O(n)

Ques:

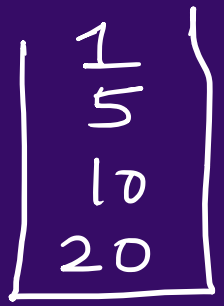
Q1 : Reverse a stack

Original

Reversed

n elements

Original



Reversed

TC: O(N)

∵ we pop elements from original stack & push in reverse stack

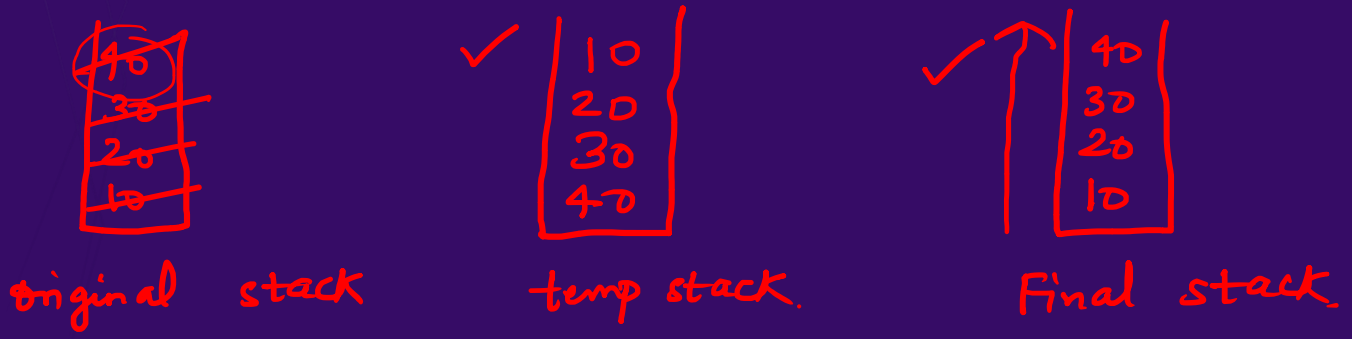
SC: O(N)

for auxillary stack space.

Ques:

PW SKILLS

Q2 : Copy stack into another stack in same order



TC: $O(N) + O(N) + O(N) = O(N)$

SC: $O(N) + O(N) = O(N)$

Ques:

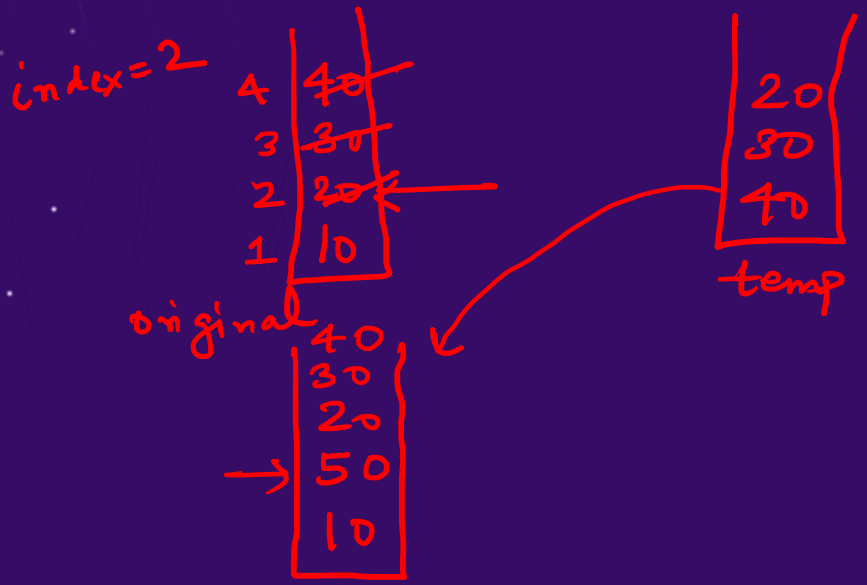
Q3 : Display stack

s.o.p (st)

TC: O(N)

Ques:

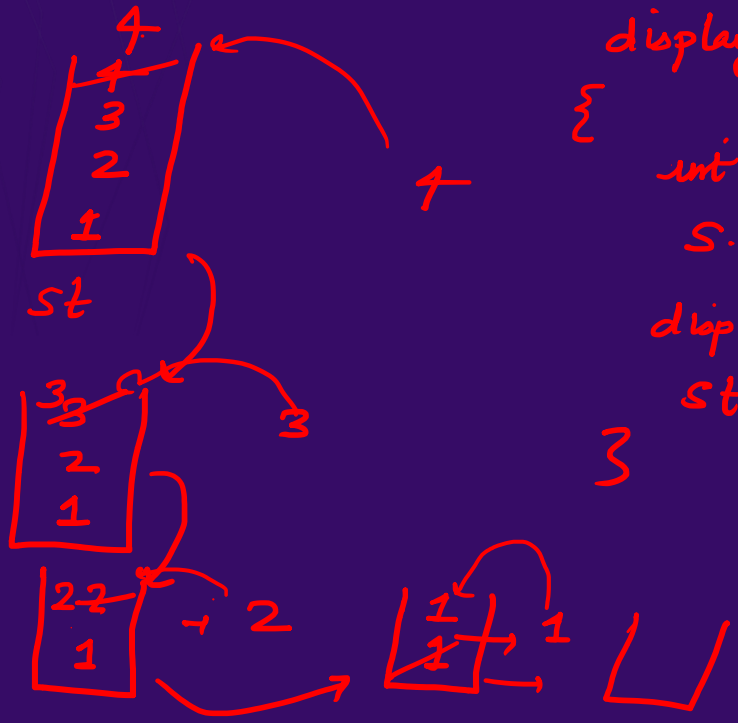
Q4 : Push element at bottom / any index




```
while(st.size() >=index)
{
  temp.push(st.pop());
  }
st.push(x);
while (temp.size >0)
{  st.push(temp.pop());
  }
```

Ques:

Q5 : Reverse stack recursively



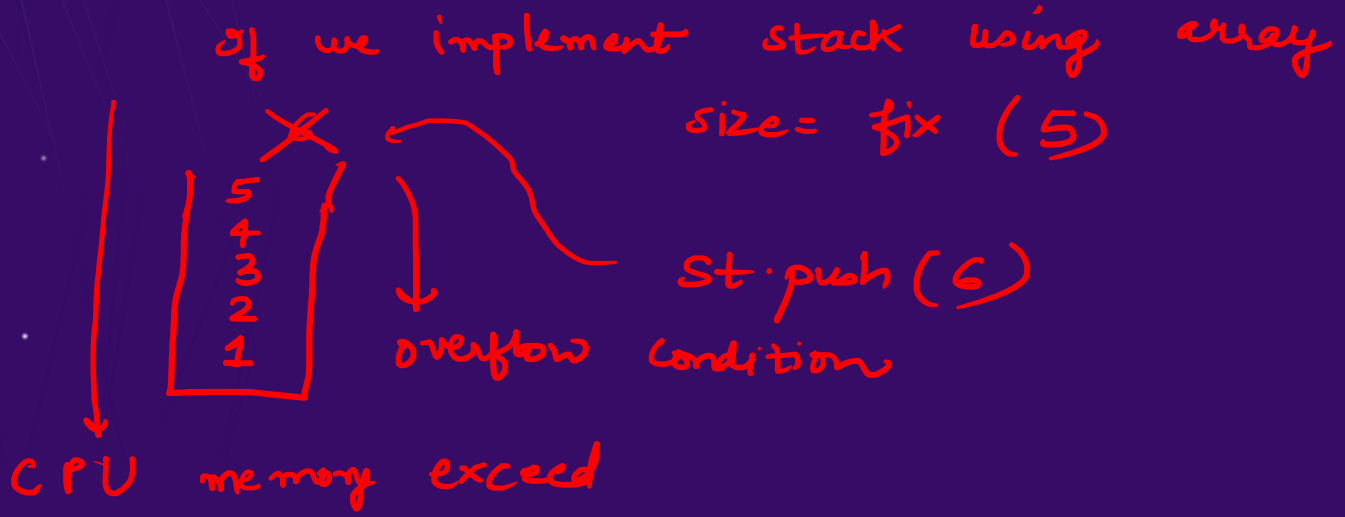

```
displayRev(st)
{
    if(st.isEmpty()) return;
    int top = st.pop();
    S.o.p(top);
    displayRev(st);
    st.push(top);
}
```

TC: O(N)

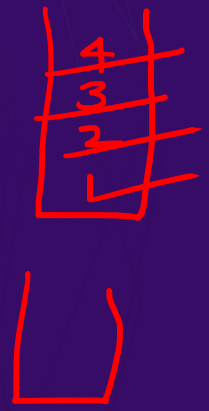
SC: O(N)

Recursive stack space

Overflow


If we implement stack using array
size = fix (5)
5
4
3
2
1
st.push(6)
overflow condition
CPU memory exceed

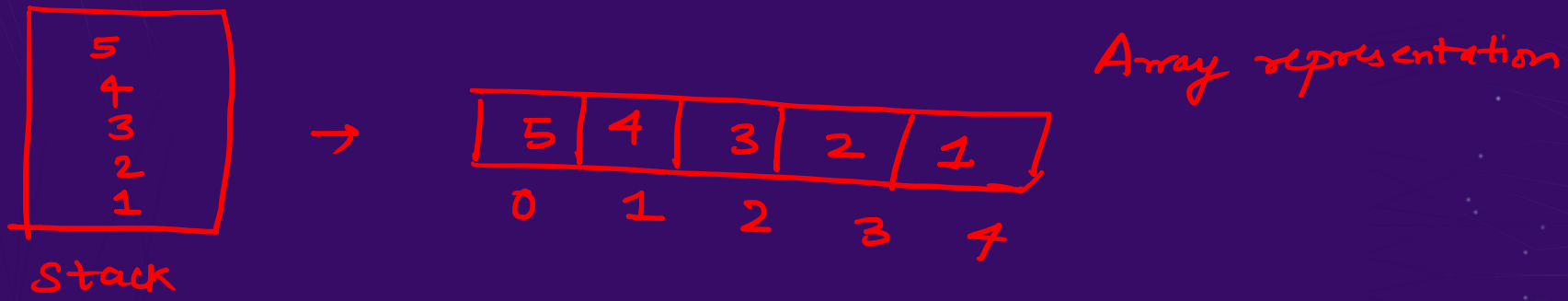
Underflow



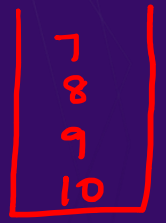
4 3 2 1

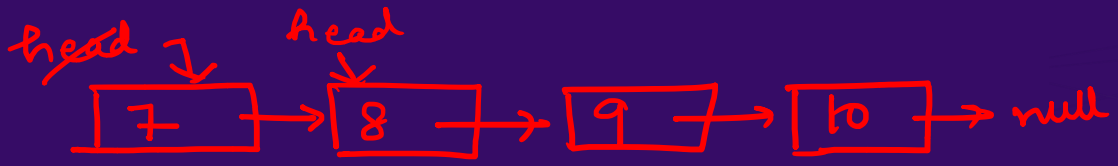
st.pop() / st.peek() ← if stack is empty, this will throw an exception

Array / ArrayList Implementation



Linked List Implementation






```
class Node
{
  int val;
  Node nxt;
}

class Stack {
   Node head = null;
   int size = 0;
```

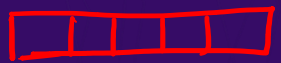
st.push(10);
st.push(9)

Push
1) create temp node
2) Point temp to head
3) Make temp as head
4) size++

Linked List VS ArrayList Implementation

PW SKILLS

Advantages of Array Repr



1) size: for every element space taken is 1 block. In LL, we have to store address as well which takes more space.

2) Display : O(1)

Disadv:

1) size is fixed
↓
overflow

Adv of LL Repr :— 1) size unlimited

Disadv of LL :- 1) size for storing data is more.

2) Display: space complexity O(N).

THANK YOU