Introduction to Stack

A List with the restriction that insertion and deletion can be performed only from one end called top of stack

ex. Stack of books LIFO

Stack of plates ⟹

operations

(1) push(x)

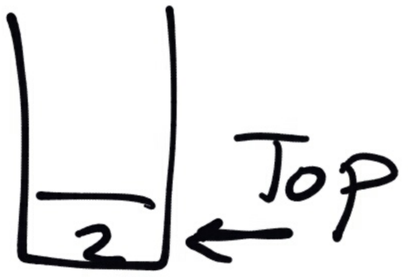
(2) pop()

(3) Top()

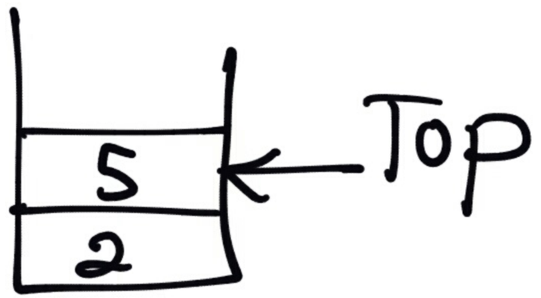
(4) IsEmpty()

(5) IsFull()

push(2)



push(5)



Top() → 5

IsEmpty → False

Pop() → 5

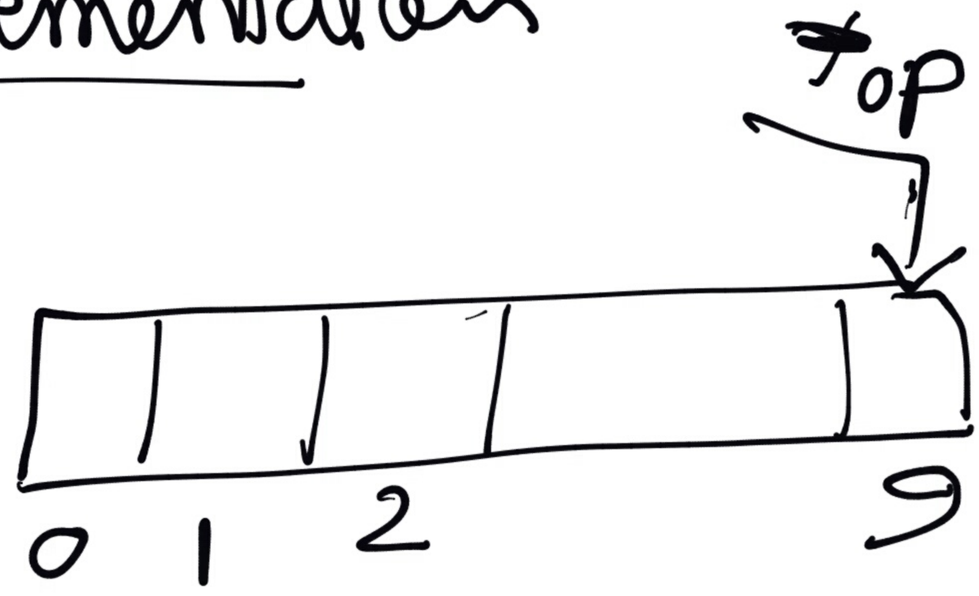
applications

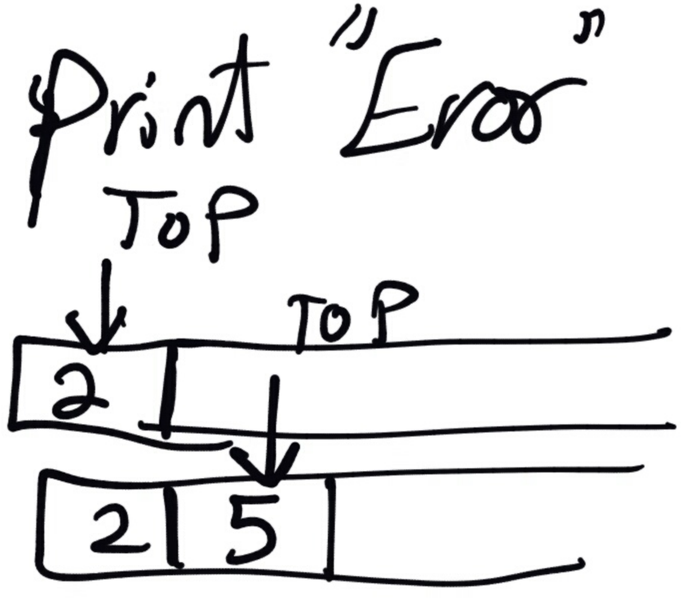
1. Function Calls
2. Recursion e.g. $n! = n(n-1)!$
3. Implementing undo operation
4. expression Parsing
   balanced Parentheses [( )]
5. Traverse any list at reverse order

array_based Implementation

```
int Max_size=10
    stack [Max_size]
    TOP = -1

push(X)
  { if (top = MaX_Size -1) Print "Eroo"
      else { Top ⇐ Top +1
             stack[TOP] ← X
  }          }
```



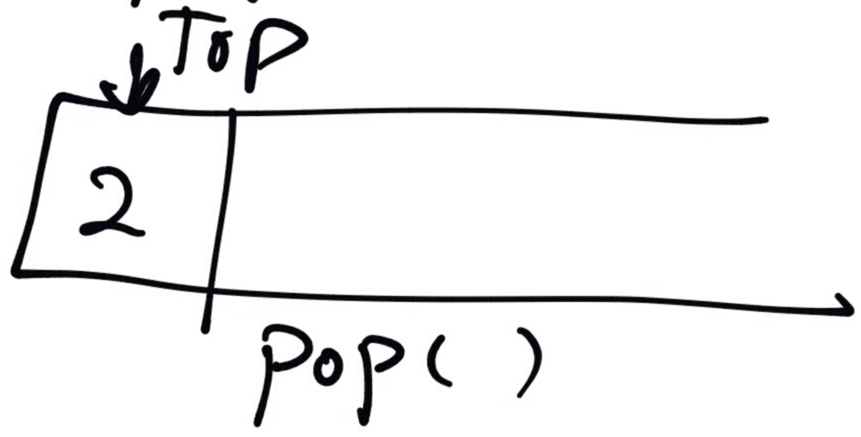


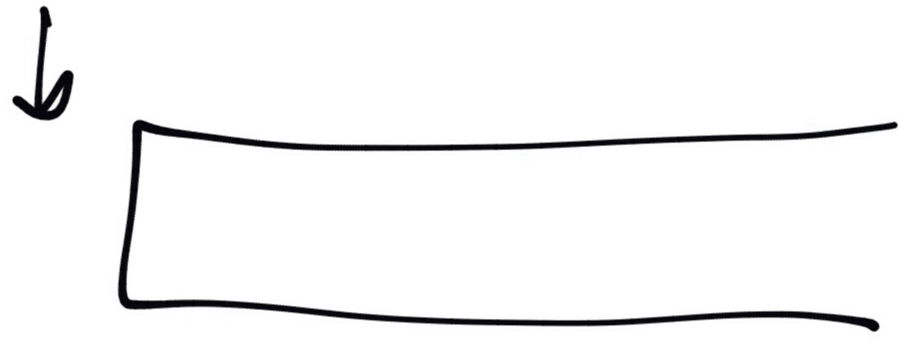
Pop ( )

{ If (tp = -1) print "Error"
else top ← top-1
}

push, pop O(1)

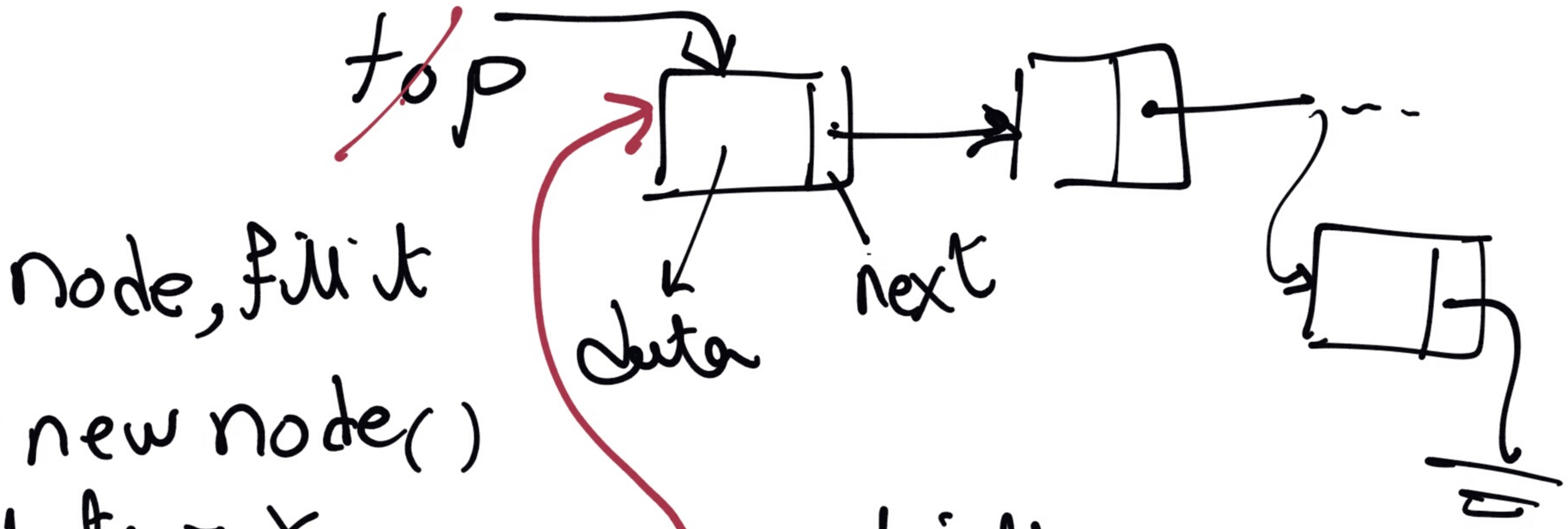
pop( )





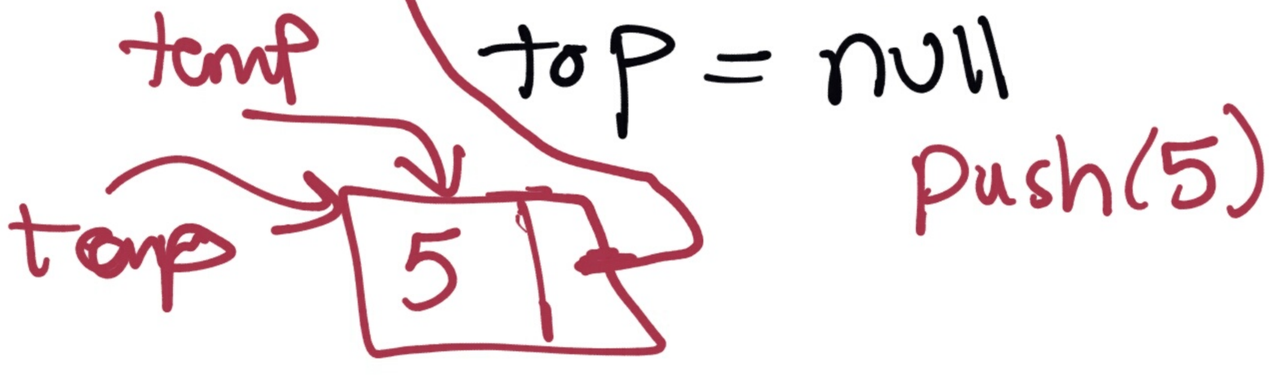
Linked list implementation of stack

```
push (x)
{// Create node, fill it
  temp = new node()
   temp.data = x
  temp.next = top
   top = temp
}
```



initially

top = null



push(5)

```
pop( )
  { if (top = null) ⇒ Error
else
  { temp = top
    top = top.next
  } return (temp.data)
  }
```