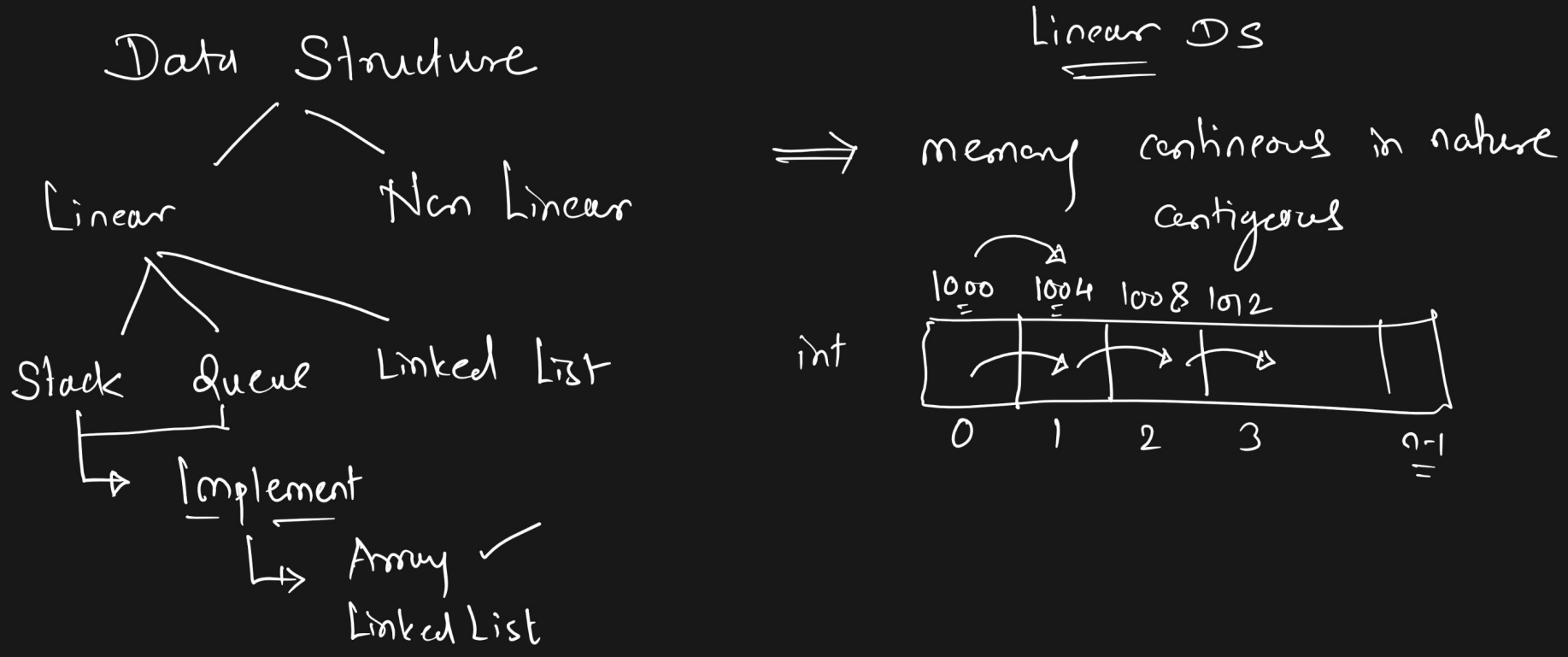
Day - 12 Stack & Queue



Stack

i) what is the stack ? ⇒ Linear DS ⇒ Stored the element in the principle of LIFO

⇒ we can implement the stack by using < Array / Linked List.

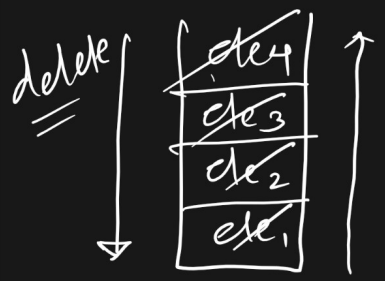
Technical Example / Appln :-

1) Message notification => Latest → first
   First → Last

2) Email => LIFO

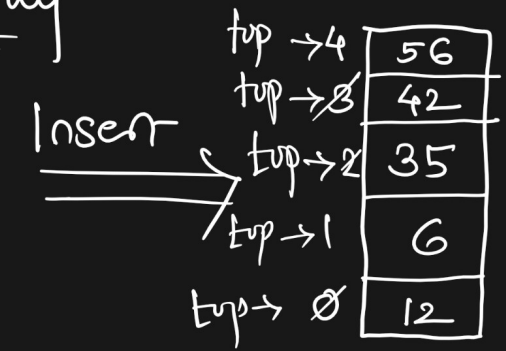
3) Browser History => whichever we search at last that url or website we can see first.

4) Download History 5) Undo operation - Text Editor



2) Implementation :- By using Array ⟹ Insert & Delete both the operation performed by one end ⇒ Top

1, 6, 35, 42, 56



top = -1

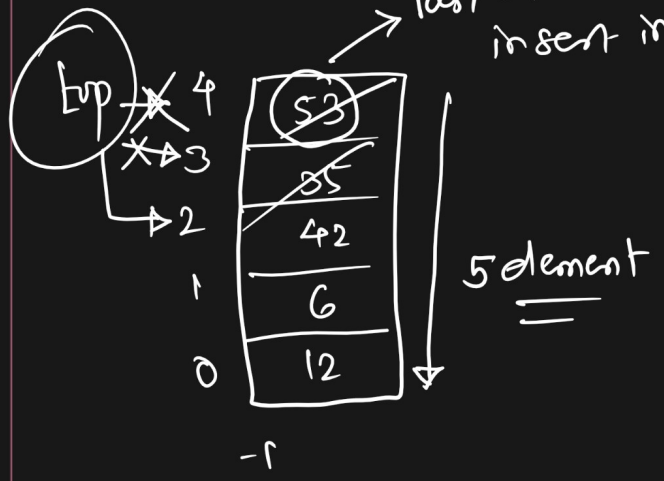
A) When stuck is empty then top at -1 location

B) 12 is coming to insert ⇒ top++ ⇒ 0th location

then we can insert 12 at arr[top] = 12

C) 6 ⇒ top++ ⇒ 1st index

6 at place arr[1] = arr[top] = 6



insert operation ⇒ top++ ⇒ arr[top] = new element

delete operation ⇒ LIFO ⇒ arr[top] ⇒ arr[4]

53

top--

O/p ⇒ 12, 6, 42, 35

delete op ⇒ LIFO ⇒ arr[top] ⇒ arr[3] ⇒ 35

top--

e.g.

~~Insert~~ ⇒ 4

~~Insert~~ ⇒ 10

~~Delete~~ = ?

~~Insert~~ = 51

~~Insert~~ = 37

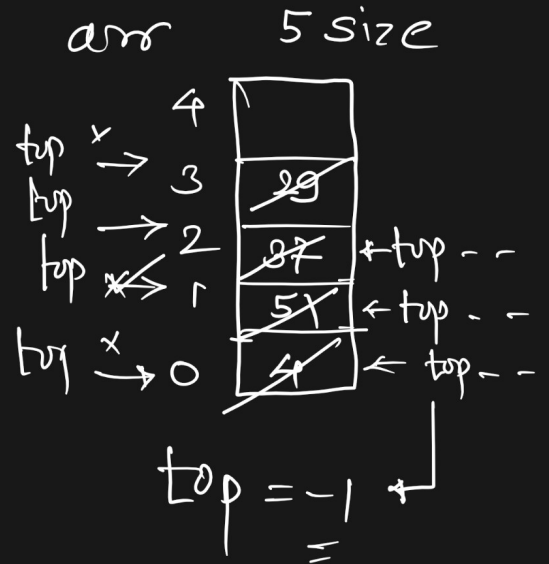
~~Insert~~ = 29

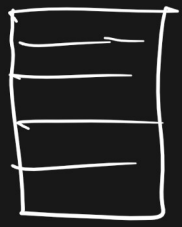
~~Delete~~ ⇒ ?

~~Delete~~ ⇒ ?

~~Delete~~ ⇒ ?

~~Delete~~ = ?





1) top++ ⇒ -1 to 0
arr[top] = 4

2) top++ ⇒ 0 to 1
arr[top] = 10

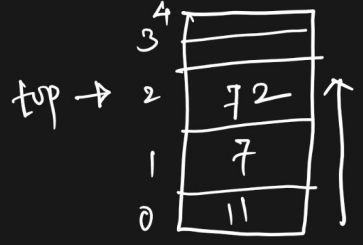
3) delete = arr[top]
top--;

4) top++ ⇒ 0 to 1
arr[top] = 51

push ⇒ Insert ⟶ top++ ⟶ arr[top] = ele

pop ⇒ Delete ⟶ result = arr[top] ⟶ top--

arr[5]

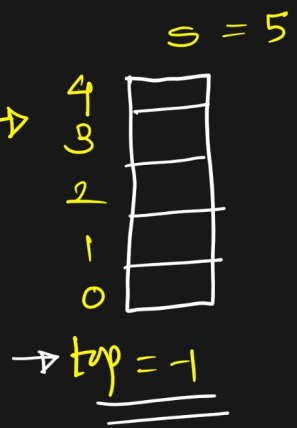


1) what value at arr[top]
peek ⇒ ? ⇒ 72

2) what is the size of stack ⇒ 3 ⇒ top+1

Programming (JAVA)

```
class Stack
{   int size;
    int arr[];
    int top;
    Stack(int s)
    {   size = s;
        arr = new int[size];
        top = -1;
    }
```




```
    boolean isFull()
    {   if(top == size-1)
            return true;    // stack full
        else
            return false;   // stack is not full
    }
    boolean isEmpty()
    {   if(top == -1)
            return true;    // stack is empty
```

1) Insert ⇒ push

Cond^ ⇒ ? ×

cond^ → my stack is full or not

How = ?

if(top == size-1) push x

2) Delete ⇒ pop

what cond^ we need to check

Cond^ ⇒ stack is empty or not

if(top == -1) ⇒ stack empty

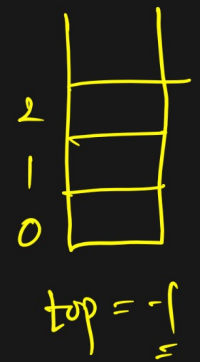
× → stack is not empty

```
    else
        return false;
}

Void push (int num)
{
    if ( isFull() )
          T/F
        S.o.p. ("Stack is full");

isFull()
↳ false → else
        {
            top ++;
            arr [top] = num;
        }
}

Void pop()
{
    if ( isEmpty() )
            T
        S.o.p. ("stack is empty");
    else
        {
```



```
            S.o.p("popElement = " + arr[top]);
        } top--;
}
```

```
void display()
{
    for(int i = top; i>=0 ; i--)
    {
        S.o.p(arr[i]);
    }
}
```

```
Void size()
{
    S.o.p(top+1);
}
```

```
void peek/top()
{
    S.o.p(arr[top]);
}
```