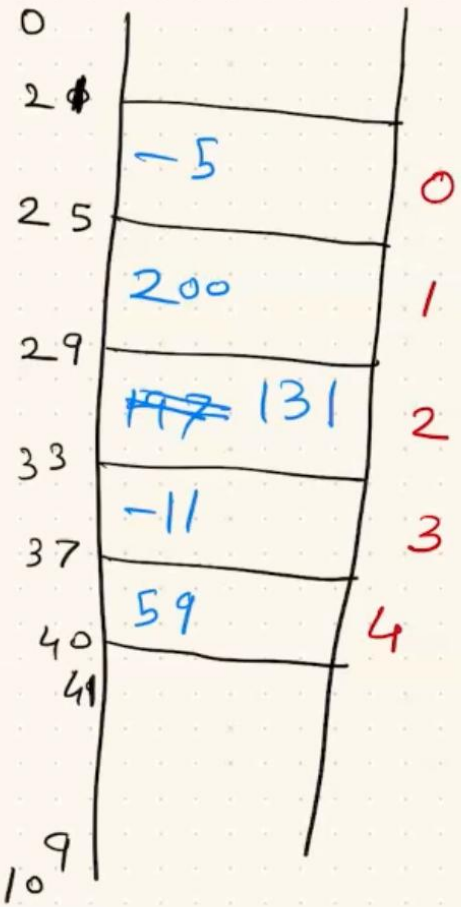
Array: Implementation with contiguous memory

createNew(5): ~~$\Theta(n)$~~ $\Theta(1)$



createNew(capacity, defaultVal)

createNew(5, 500)

$\Theta(1) + \Theta(n)$

$f(n) = 40 + 7n$

$f(n) \in \Theta(n)$?

$f(n) = 20 + 5n + 3n^2 + 0.5n^3$

$f(n) \in \Theta(n^3)$

Stack implementation using Array

Data: Array A, integer count

```
CreateNew (capacity, type){
    A.createNew(capacity, type)
    count=0
}
```

Θ(1)

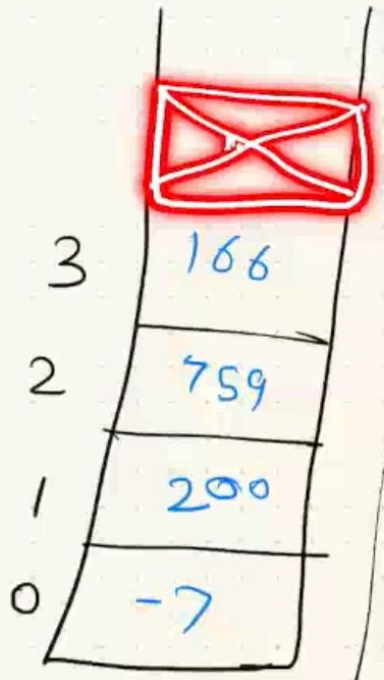
```
isEmpty () {
    if (count==0) {
        return TRUE
    }
    return FALSE
}
```

Θ(1)

```
isFull () {
    if(count=capacity) {
        return TRUE
    }
    return FALSE
}
```

Θ(1)

push (val) {



count = 0

Stack

A

Count

createNew

isEmpty()

count

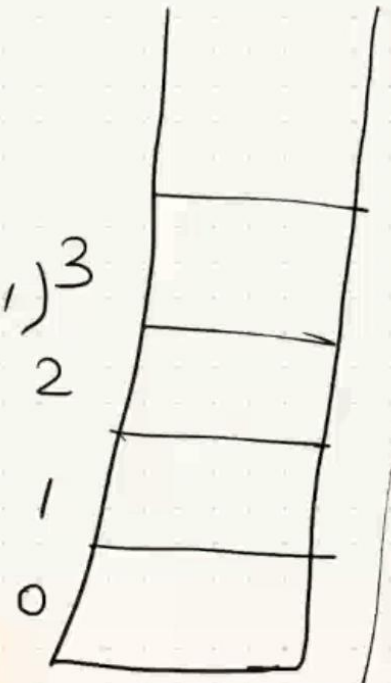
```
push ( val ) {
  if( isFull() ){
      return ERROR
  }

  A.StoreValue(count, val)
  count = count + 1
}
```

$\Theta(1)$

$a \cdot n + b + c$

$\Theta(n) + \Theta(1)$



count = 0

Stack

A

Count

createNew

isEmpty()

count

⟨value, arrival time⟩ → ⟨value, array index⟩

use count as an indicator of current time

Stack implementation using Array

Data: Array A, integer count, integer capacity

```
CreateNew (pCapacity, type){                    Θ(1)
    capacity = pCapacity
    A.createNew(capacity, type)
    count = 0
}

isEmpty() {                                     Θ(1)
    if (count == 0) {
        return TRUE
    }
    return FALSE
}

isFull() {                                      Θ(1)
    if(count = capacity) {
        return TRUE
    }
    return FALSE
}
```

```
pop ( ) {
    if ( isEmpty() ){
        return ERROR
    }

    count = count - 1

answer = A.accessVal( 0 )
    return answer
}
```

$\Theta(1)$

S. createNew(4, integers)

S. pop → error

S. push(14)

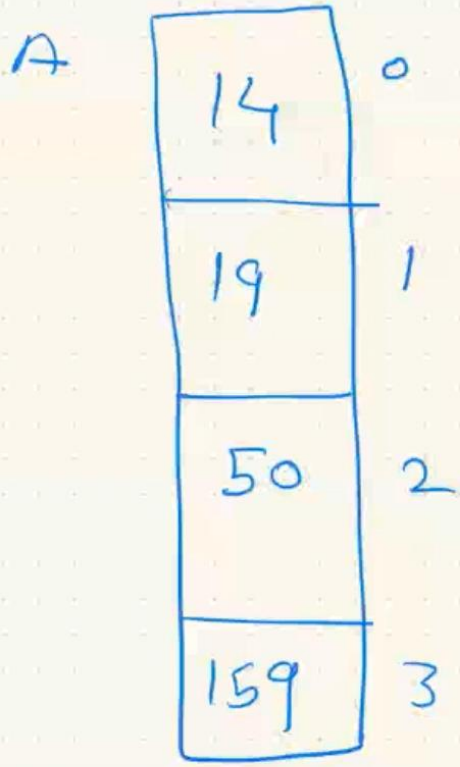
S. push(19)

S. push(50)

S. push(159)

S. push(99) → error

S. pop() → 159



Count = ~~0 1 2 3~~ 4