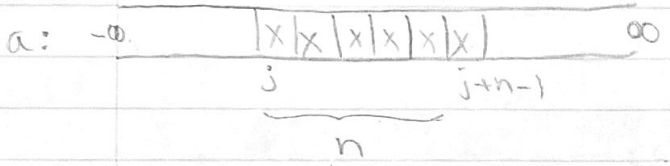
# ASSIGNMENT 4 COMP2402 MIDTERM STUDY

| | WASTED SPACE | get(i)/set(i,x) | add(i,x)/remove(i) |
|---|---|---|---|
| **ARRAYSTACK** : (ARRAYLIST IN JLF) | $O(n)$ | $O(1)$ | $O(1+n-i)$ |

- TOTAL # OF CALLS TO add(i,x) OR REMOVE(i) BEFORE RESIZE() IS CALLED IS : $n/2 - 1$

| | | | |
|---|---|---|---|
| **ARRAYQUEUE** : | $O(n)$ | | add(x)/remove() $O(1)$ |

- FIFO (NOT LIST INTERFACE)




- add(x)

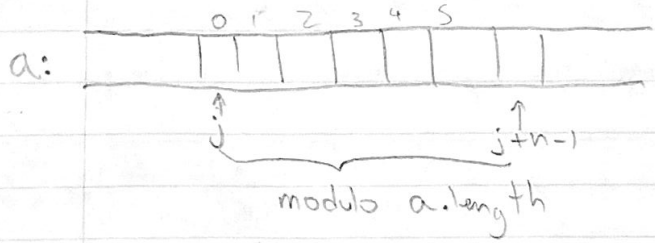
```
if(n+1 > a.length)
    resize();
a[(j+n)%a.length] = x;
n++;
```

- remove(i)

```
j = (j+1)%a.length;
n--;
```

| | | | |
|---|---|---|---|
| **ARRAYDEQUE** : | $O(n)$ | $O(1)$ | $O(1+\min\{i, n-i\})$ |

- IMPLEMENTS LIST INTERFACE




- get(i) → return a[(j+i)%a.length]
- set(i,x) → a[(j+i)%a.length] = x

- add(i,x) ← if(n+1 > a.length) resize()

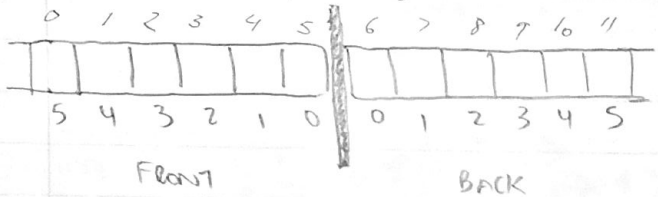
```
if(i < n/2) //THEN SHIFT ELEMENTS 0...i-1 TO THE LEFT
else        //SHIFT ELEMENTS i...n-1 TO THE RIGHT
    a[i] = x;
n++;
```

- remove(i)

```
if(i < n/2) //SHIFT 0...i-1 RIGHT, and j = (j+1)%a.length
else        //SHIFT i...n-1 LEFT
n--;
```

## DUALARRAYDEQUE :

- IMPLEMENTS LIST INTERFACE WITH 2 ARRAY STACKS




| WASTED SPACE | get(i)/set(i,x) | add(i,x)/remove(i) |
|---|---|---|
| O(n) | O(1) | O(1+min{i, n-i}) |

- get(i)

```
if (i < front.size())
  return front.get(front.size()-i-1);
else
  return back.get(i-front.size());
```

- add(i,x)

```
if (i < front.size())
  front.add(front.size()-i, x);
else
  back.add(i-front.size(), x);
balance();
```

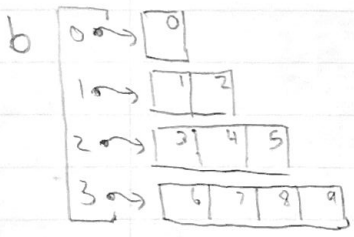
- AT LEAST $\frac{1}{2}n-1$ add/remove OPERATIONS BETWEEN SUBSEQUENT CALLS TO balance().

- POTENTIAL METHOD = $\Phi = |front.size() - back.size()|$

- remove(i)

```
if (i < front.size())
  return front.remove(front.size()-i-1);
else
  return back.remove(i-front.size());
balance();
```

## ROOTISHARRAYSTACK :




| | | |
|---|---|---|
| O(√n) | O(1) | O(1+n-i) |

- get(i)

$b = \lceil (-3+\sqrt{9+8i})/2 \rceil$ (ceil)

$j = i - \frac{b(b+1)}{2}$

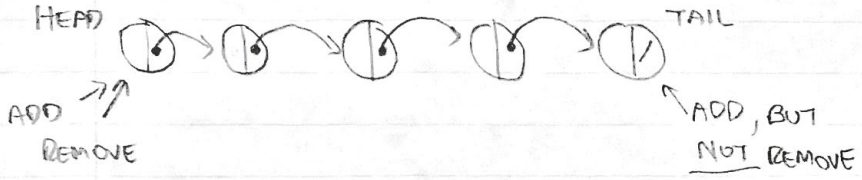
```
return blocks.get(b)[j];
```

- grow()/shrink() TAKE O(r) TIME

## SINGLYLINKEDLIST :

- IMPLEMENTS FIFO AND STACK

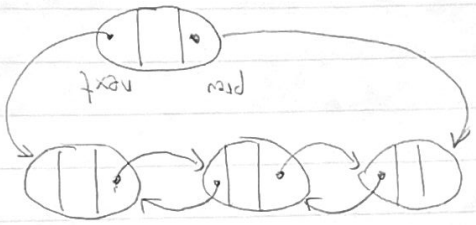
| push(x), pop() (AS A STACK) | add(x)/remove() (AS A FIFO QUEUE) |
|---|---|
| O(1) | O(1) |



## DOUBLY LINKED LIST:

- IMPLEMENTS LIST INTERFACE

| get(i)/set(i,x) | add(i,x)/remove(i) |
|---|---|
| $O(1+\min\{i, n-i\})$ | $O(1+\min\{i, n-i\})$ |



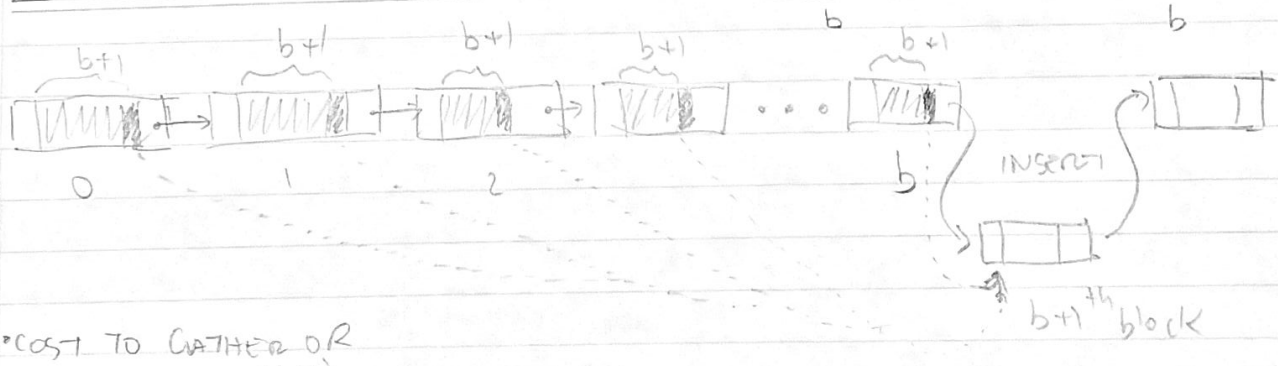

- ADDING NODE u BEFOR NODE p

```
u.next = p;
u.prev = p.prev;
u.next.prev = u;
u.prev.next = u;
```

---

## SPACE EFFICIENT LINKED LIST:

$O(1+\min\{i, n-i\})$ $O(b+\min\{i, n-i\})$




- COST TO GATHER OR SPREAD IS $O(b^2)$

---

## SKIPLISTS:

SORTED SET

↳ find(x), add(x), remove(x) } $O(\log n)$

RANDOM ACCESS LIST

get(i), set(i,x), add(i,x), remove(i) } $O(\log n)$

↳ SEARCH PATH FOR x IN $L_0$

- START AT TOP OF SENTINEL
- IF WE CAN GO RIGHT W/O REACHING OR OVERSHOOTING x
  - THEN GO RIGHT
  - ELSE GO DOWN

THEOREM: FOR ANY VALUE OF $x$, THE EXPECTED LENGTH OF THE SEARCH PATH FOR $x$ IS AT MOST

$$2 \cdot \log_2 n + \underbrace{O(1)}_{\leq 2}$$

IF $n = 1{,}000{,}000$, $\log_2 n \approx 20$, =-

$2 \cdot 20 + 2 = 42$ LENGTH OF SEARCH PATH, FOR $n$ OF $1{,}000{,}000$.