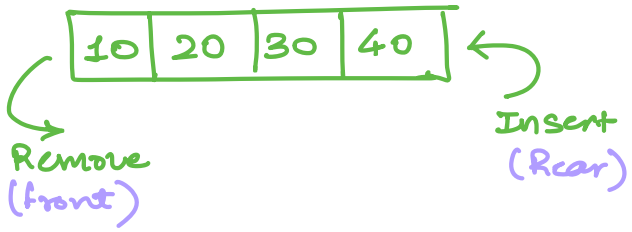
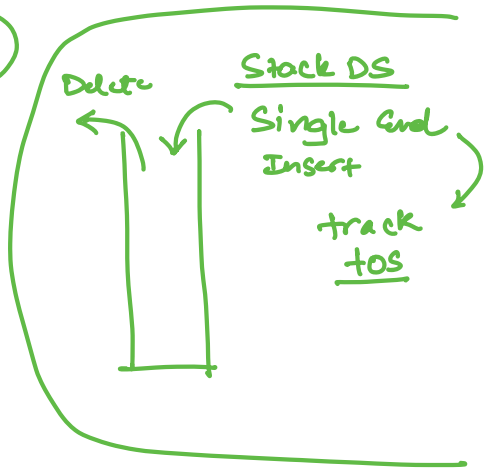
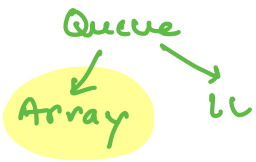
Queue DS : FIFO (First In First Out)



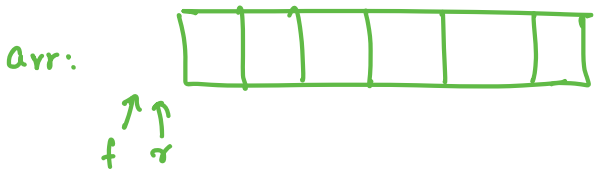
2 different ends





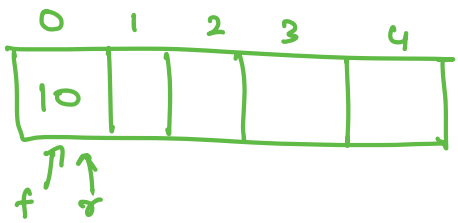
Queue using Array

cap ? — Default / User.



f = -1
r = -1
size = 0

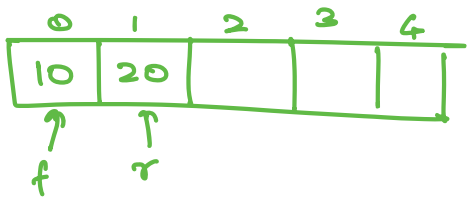
insert 10



f = 0
r = 0
size = 1

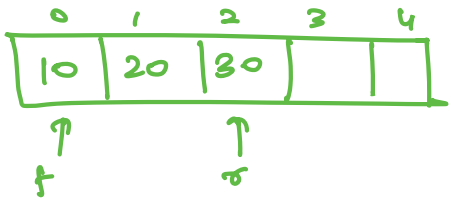
front : delete
rear : insert

insert 20



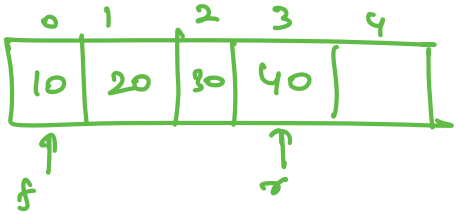
size = 2

insert 30



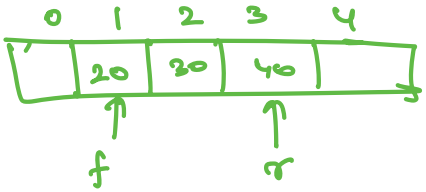
size = 3

insert 40



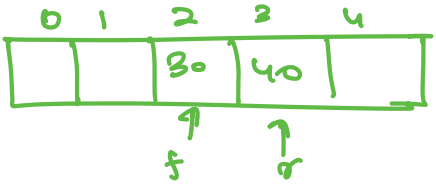
size=4

delete : remove the element which is the oldest



size=3

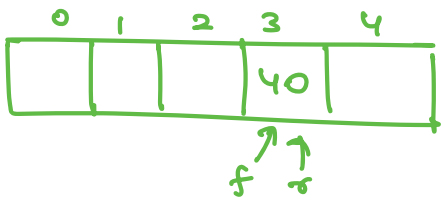
delete



size=2

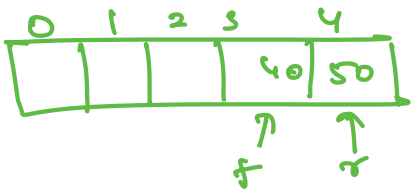
DRAWBACK

delete



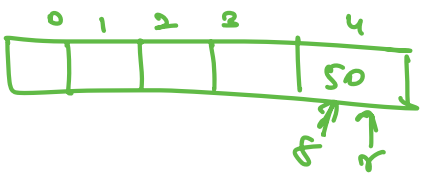
size=1

insert(50)



size=2

delete



size=1

insert

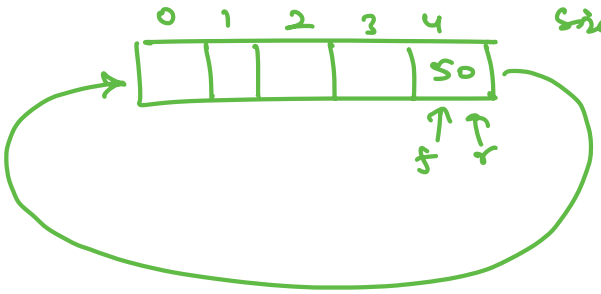
rear=5

arr(5)=60 ? 5th index array not present

Soln? Circular Queue

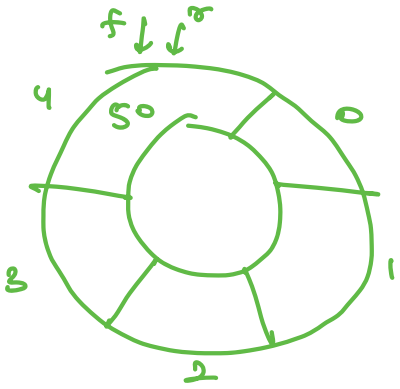
insert(60)

rear=rear+1
=4+1=5



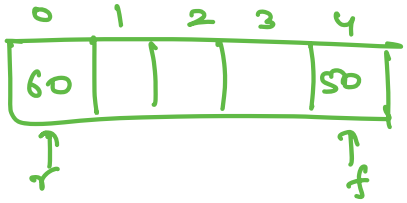
Size=1

50



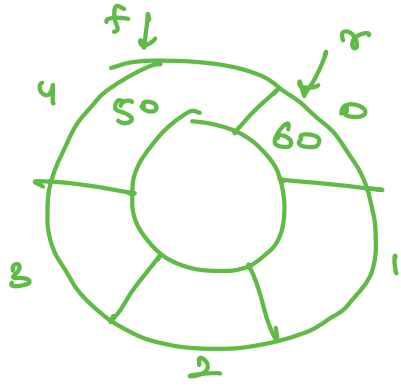
5 → 0

5%5=0

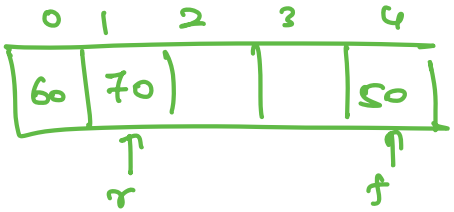


Size=2

5 60

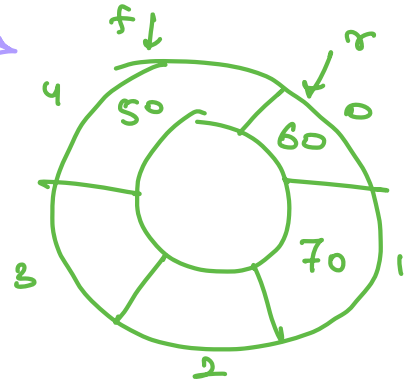


insert(70)

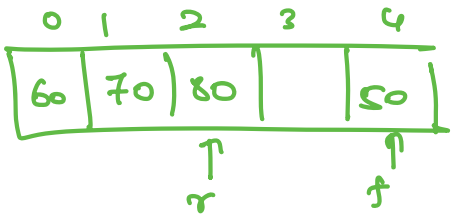


Size=3

50 60 70



insert(80)

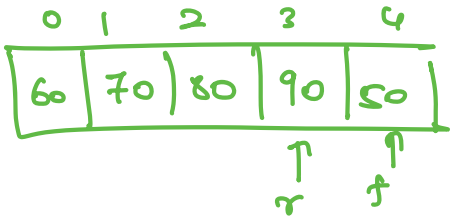


Size=4

50 60 70 80

User: Abstraction

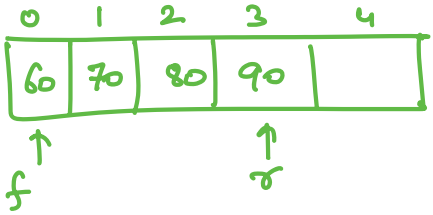
insert(90)



Size=5

50 60 70 80 90

delete

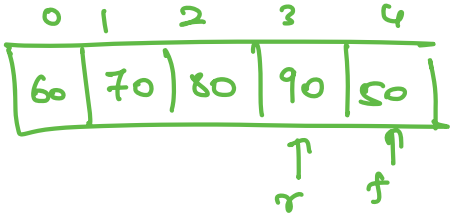


Size=4

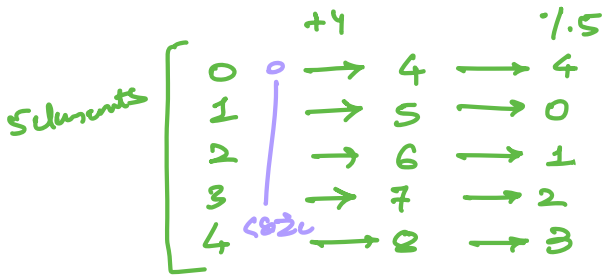
f=f+1
=4+1=5
5%5
0

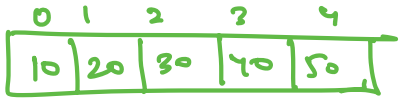
60 70 80 90

DISPLAY:



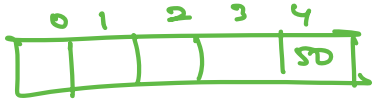
size = 5





10 20 30 40 50

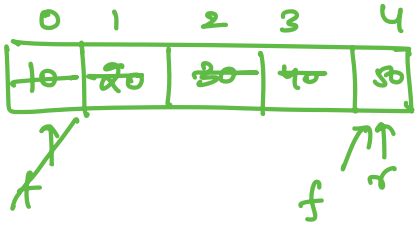
delete
delete
delete
delete



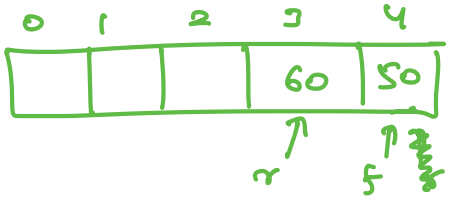
50

push(10)
(70)
(10)
(90)

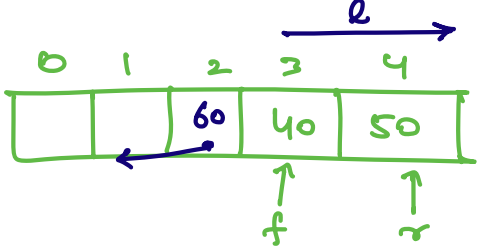
50 60 70 80 90



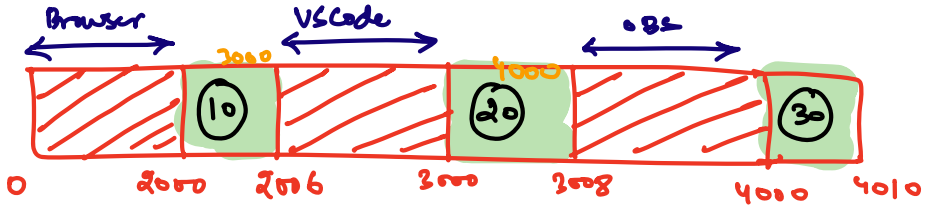
dqⁿ
0
1
rear++ ✓
rear-- ✓



rear--



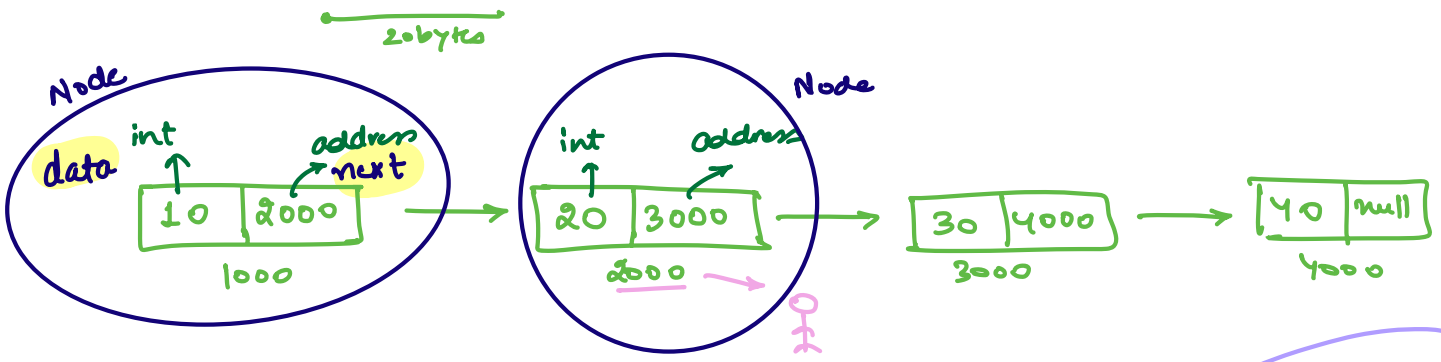
Linked list



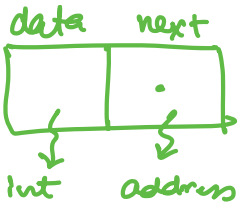
6 + 8 + 10 : 24 bytes

array 5 int → 20 bytes

→ contiguous space




```
Class Node
{
    int data;
    Node *next;
}
```



int a=10

a 10 (1000)

int *ptr = &a = 1000

Traverse

head : track of the first node : 1000

```
Node *head = 1000;
cout << head -> data;          // 10
cout << head -> next -> data;  // 20
          (2000)
```

cout<< head->next->next->data; //30

3000

cout << head->next->next->next->data; //40

4000

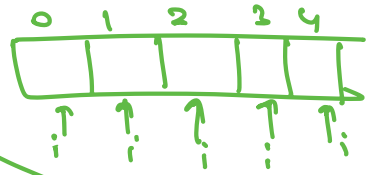
i → temp

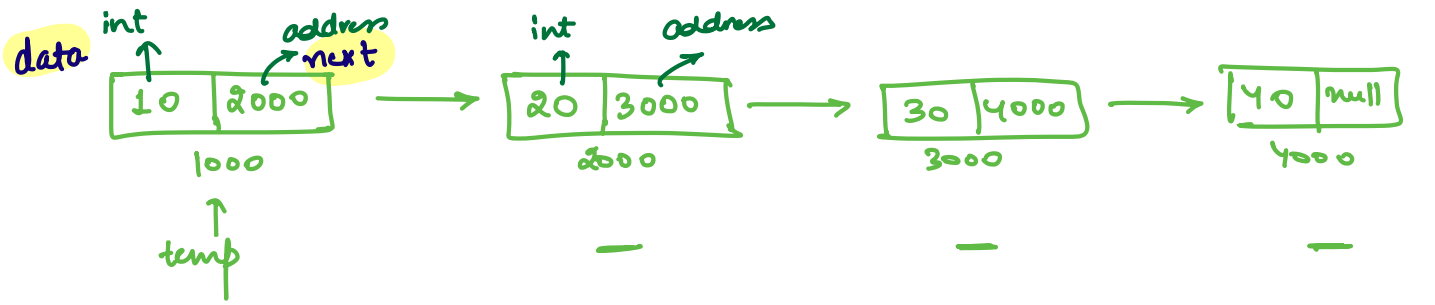
0 → head

int i=0 → Node *temp = head

i=i+1 → temp = temp->next

```
for(int i=0; i<n; i++)
{
cout<<arr[i];
}
```





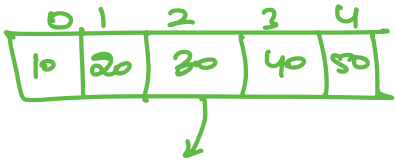
Node *temp = head; 1000

while(temp != NULL)

cout<<temp->data; 10 20 30 40

temp = temp->next temp=2000 3000 4000 NULL

Array: Random Access



= ba + (idx * dt size)

index get/set : O(1)

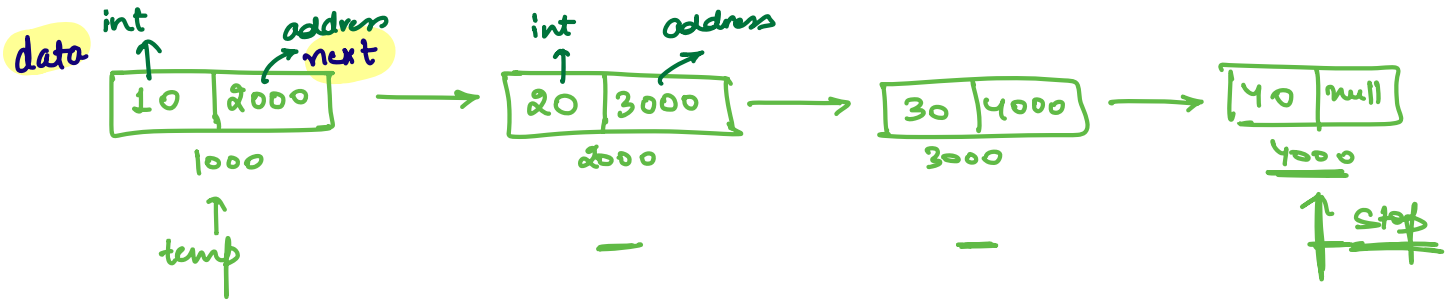
Reason: Contiguous

Linked List: Sequential access

0 1 2

O(n)

getLast




```
while (temp→next!=null)
```

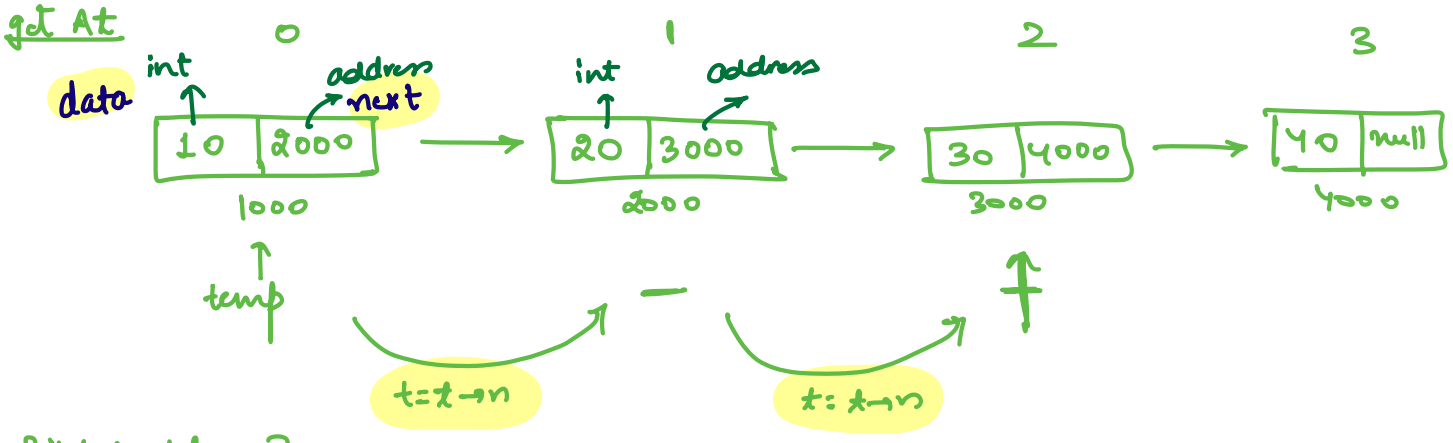
t=1000 — 1000→next = 2000 — !=null ✓

t=2000 — 2000→next = 3000 — !=null ✓

t=3000 — 3000→next = 4000 — !=null ✓

t=4000 — 4000→next = null — !=null ✗ stop

getAt



2 Index Value ?

AddLast

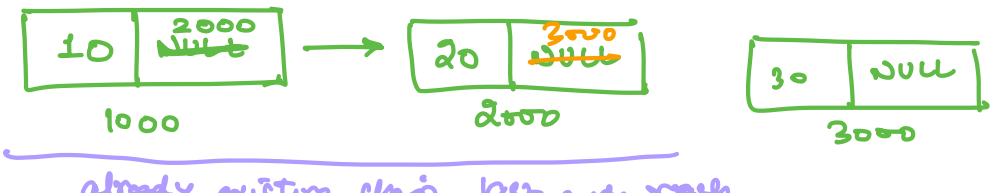
LL Empty, Adding the first Element.

add node 10

add node 20

add node 30

head = ~~null~~ 1000



Create a node Node *nn = new Node(10); 1000 2000 → next = 3000

Node *nn = new Node(20); 2000

Stack