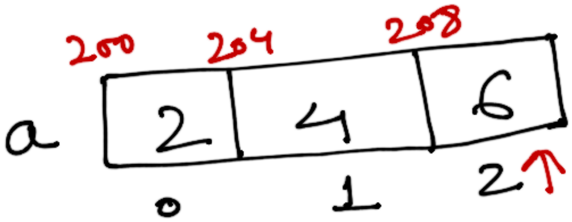
Arrays vs Linked List

1. Cost of accessing an element



Eg: $i = 2$

Address of $a[i]$

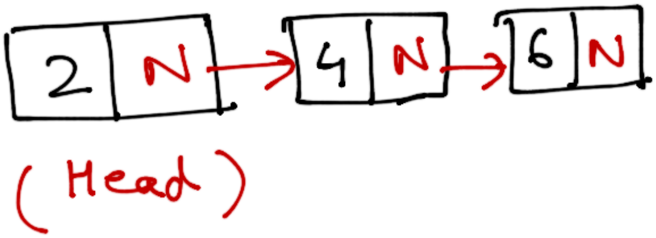
$= \text{Base Add.} + i \times \text{Size of Datatype (in bytes)}$

$= 200 + 2 \times 4$

$= \underline{208}$

$O(1)$

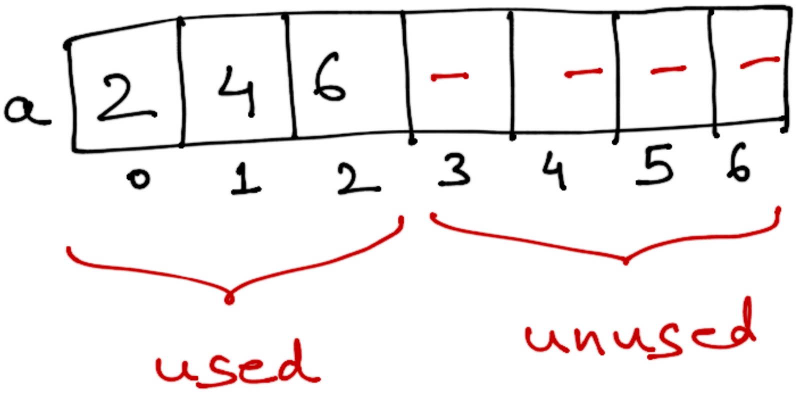
Random Access



→ Traverse the list → $O(n)$

Sequential Access

2. Memory Usage

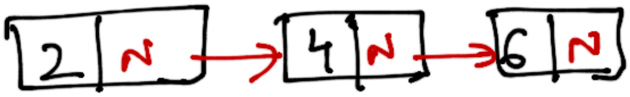


Eg!
7 X 4 = 28 bytes

Eg: 7 X 16 = 112 bytes

Used: ?

Unused: ?

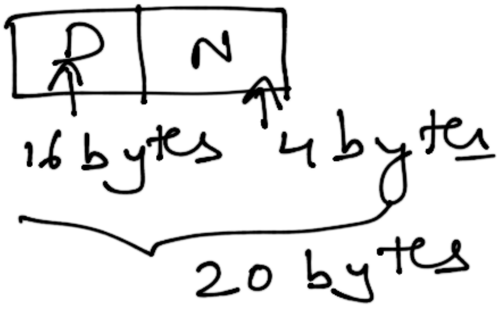


→ No unused memory

→ Extra memor
(
for pointers

Eg: for above,
8 X 3 = 24 byt

→ Another datatype of 16 bytes



Eg: 20 X 3 = 60 byte

*Memory Requirement & Utilization

→ One big chunk

→ multiple Small block

→ when array becomes full, we need to copy the whole data to a new array.

3) Cost of inserting an element

a) → At Beginning

shifting O(n)

b) → At End → Not full O(1)

→ List Full O(n)

c) → At ith position – O(n)

for Linked List

a) O(1)

b) O(n)

c) O(n)

4. Deleting an element	Same as insertio
* Linear / Binary search both possible	* Binary Search NOT possible
5. Ease of Use	Errors: → Segmentation fault • Trying to access memory that does NOT belong to you Eg: Write violatio Accessing address that is freed
*Cache Friendly Arrays are contiguous so locality of reference	→ Memory Leaks Memory NOT needed any longer is not released