Abstract Data Types

- It is blueprint which give minimal requirement with some operation.

ADT ⟸ Modal to make DSA

ADT → Minimal req. , Operation

DSA → Actual implementation

MRF :- Minimal Requirement functionality.

Arrays as ADT

MRF
- get (i)
- set (i, num)
:

Methods / Operations
- Insert
- delete
- Add
- Resize.
:

- Max()
- Min()
- Search (num)

Abstraction :- Hiding details { Aam khao gunthiya mat gino }

↓

No implementation details // We don't know how the operation is implimented

Usage

- Array is collection of elements accessible by an index.

- Size of array can't be resized / changed.

- Resizing can be done by copying like.

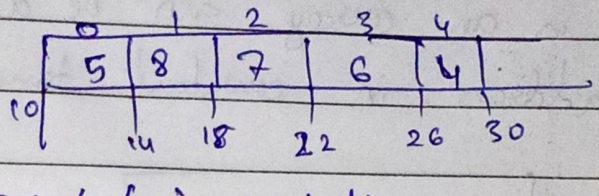
```
int * a = (int*) malloc ( 10 * sizeof (int));
int *b = (int*) malloc ( 20* size of (int));
   b = a ;
```

Advantage

Q. Why Array ??

⇒ Faster access of elements in array O(1).

Ex.



} constant time

10 + 4 (i) ⇒ Address and we can access element with constant time.

[Realloc don't change size, it point a new array from prev. array size]

⇒ Values can't be changed fastly.

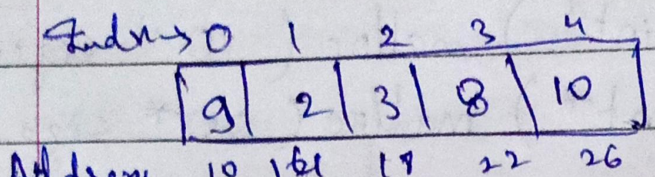
Disadvantage

Why not array ?

- Insertion and deletion is little tough as it require resize of arrays.

Static Arrays :- Size cannot be changed
Dynamic Array :- Size can be changed. // Copying array.

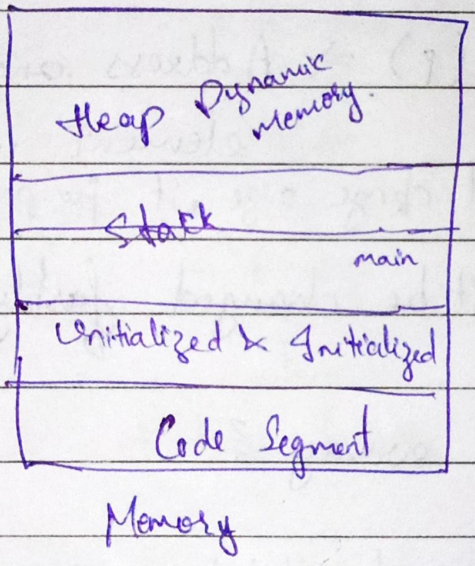
Memory representation of Arrays.



⇒ Array of size 5

- Elements in an array are stored in contiguous memory locations.

- Elements in an array can be accessed using the base address in constant time → O(1)



- Structure in Array are used to make custom datatype in C.