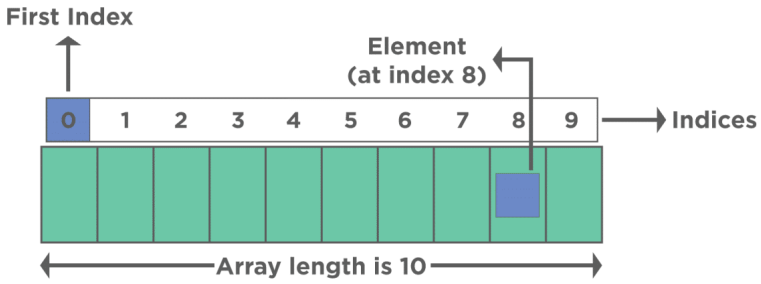


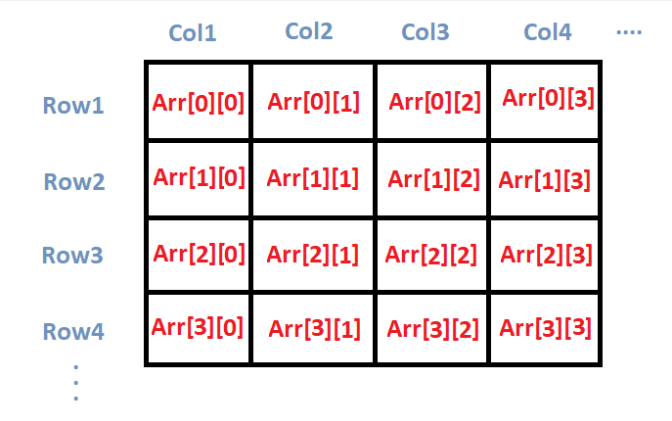
Arrays

Tags	Data-Structure
Companies	EVERY
Created time	@October 29, 2022 6:38 AM
Last edited time	@October 29, 2022 1:50 PM
URL	https://github.com/JiteshBhashwani/DSA/tree/master/Data-Structures/ArrayAdt

Definition - An Array is a linear data structure used to store similar datatypes in a contiguous memory locations. this makes it easy to calculate the position of any element from their indices and access them in O(1) time even in the worst case.

Points/Facts :-

1. Stores elements in contiguous memory blocks.
2. Each element has its index. Zero-based indexing is most often used in most popular languages like c,c++,java,python,…. etc..:The first element is in 0th index, the second in index 1 and so on. which can be called using the array name followed by square brackets with index inside. ex- arr[0] means we are accessing ‘0th’ element of an array ‘arr’.
3. it is fixed in size.means we have to declare how many elements we will be storing in that array at max.
4. since we can make an array of any data-type or data-structure.we can make an array of array where each element is first element an array. accessing them is a little different as each element is an array itself we can treat mat[0] as the name of the arrays, thinking like that it makes it easy to access any elements from array of an array. mat[0] is array name and each element can accessed as mat[0][0],mat[0][1]… we call it two dimensional array that can be visualised as a matrix. we can also make array of matrix(array of array) and matrix of matrix, possibility is endless but mostly useless, 1D 2D and 3D array are mostly used because we use programming to solve real world problems.



5. setting Array as parameters in any function definition will always be call by address of the argument from function call. not as copy.

Pros:

1. Elements can be accessed in O(1),
2. Constant time append(insertion of element at the end of the array).

Cons:

1. Fixed size. which can make it memory intensive.
2. searching, insertion and deletion takes linear time. because after deletion we cannot have an empty element we have to move every element on array accordingly.

⌛ **Time Complexity:**

Access O(1).

Search O(n) when unsorted O(log n) when array is sorted.

Insertion O(1) in unsorted(just append the element in back) and O(n) in sorted.

Deletion O(n).

Basic Questions On Array

Find min and max of array

```
int minE = arr[0];
int maxE = arr[0];
for(int i = 0;i < size;i++)
{
  minE = min(minE,arr[i]);
  maxE = max(maxE,arr[i]);
}
cout << minE << " " << maxE;
```

Find sum of all element of array

```
int sum = 0;
for(int i = 0;i < size;i++)
```

```
  sum+=arr[i];
cout<<sum<<endl;
```

Linear Search

```
int findIndexOf(int arr[],int size,int key)
  for(int i = 0;i < size;i++)
    if(arr[i] == key) return i;
  return -1;
}
```

Reverse Array

```
void rev(int arr[],int size)
{
  int i = 0;
  while(i<size/2)
    swap(arr[i++],arr[size-1-i]);
}
```

Swap Alternate

```
void swapAlt(int arr[],int size)
{
  int i = 0;
  while(i<size)
  {
    swap(arr[i],arr[i+1]);
    i+=2;
  }

}
```

More Questions solution 🔗

sheet leetcode