Arrays and Vectors

1 Arrays

Data is arranged in contiguous memory locations.

2 Data Structures

Data structures define how data is stored in memory and typically support the following operations:

1. Read
2. Insert
3. Delete
4. Update

3 Create

- Integer Array (fixed size): `int A[100];`
- Integer Vector (dynamic size): `std::vector<int> vec;`

4 Read

Read the i-th element:

- Array: `A[i]`
- Vector: `vec.at(i)`

5 Insert

Insert element at position i:

Vector:

```
vec.insert(vec.begin() + i, element)
```

Array: To insert an element at the i-th location:

- Shift all elements from index $i + 1$ onward one position back.
- Insert the new element at index i.
- Requires pre-allocated space.

Vector (internal mechanism):

- If no space, a new dynamic array (typically twice the size) is allocated on the heap.
- Existing elements are copied over.
- The old array is deallocated.

6 Delete

Delete element at position i:

Vector:

```
vec.erase(vec.begin() + i)
```

Array:

- Shift all elements from index $i + 1$ onward one position forward.
- This leaves wasted space in the array, since the size doesn't shrink.

Vector (internal mechanism):

- If the number of elements falls below half the capacity, a new dynamic array (half the size) is allocated.
- Elements are copied over, and the old one is deallocated.

7 Update

Update element at position i:

- Array: `A[i] = new_val;`
- Vector: `vec[i] = new_val;`

8 Time Complexity (Big-O)

Operation	Time Complexity
Read	$O(1)$
Insert	$O(n)$
Delete	$O(n)$
Update	$O(1)$

- Read and Update are constant-time operations: $O(1)$
- Insert and Delete are linear-time in the worst case: $O(n)$

9 Practice Problem

Leetcode Problem 27: *Remove Element*
`https://leetcode.com/problems/remove-element/description/`