Day 3

Friday, 10 March 2023 11:31 PM

Main Topics → Introduction to data Structures & Algorithms
→ Arrays, Strings and Matrix

Side Topics → Structs
→ Lambda

• Introduction to DSA

Data? → qty, Char, Symbol on which operations are performed by computer. eg: $c = a + b$.

→ Information?

Meaningful / processed data is called information.

Data Structures? → To provide a meaningful way to structure our data, we need to know about Data Structures.

↳ Organizing data so that it can be used efficiently.

Real World Ex,
↓ Social Media ↳ Graph
↳ Stack used in Redo / Undo feature.
↳ Undo / Redo Stack

• Arrays

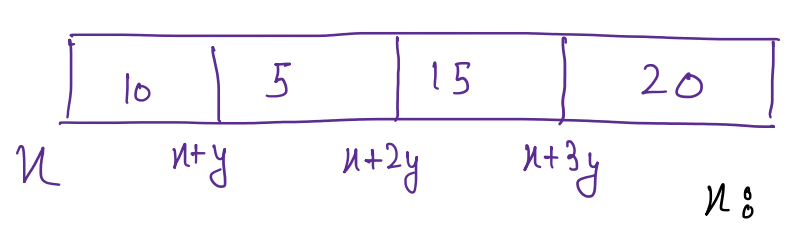
↳ Store multiple item together at one place, usually of same data type.

- Why do we need array? → Store 10 variables, etc.

✯ Which Data Structure is used to Store BitMap Images?
↳ BitMap image → 2D Array

↳ Other DS that can do same job ex Linked List, Hash, Self Balancing BST etc.

- Elements are stored at continous Location.
- Concrete implementation of ADT's. → ex List



x: Address where array is stored
y: Size of array element.

https://stackoverflow.com/questions/1239938/accessing-an-array-out-of-bounds-gives-no-error-why

Good thread to check why are we able to access array out of bounds.

→ Defining array

- `int A[5];` → Stack (fixed Size Array)
- `int *A = new int[5];` → heap (Dynamic Size Array)
  ↳ Double the Size if Size of insertion increase.
- `array<int, 5> A;`
- `int A[] = {10, 5, 15, 20}`

* Java allocate all arrays in heap.

→ Space / Time Complexity

Access : O(1) Remove : O(1)
Insert : O(n), O(1) ← at End Search : O(1)
↳ If array is full / Insert at beginning

✓ Random Access
✓ Cache Friendliness

• Strings

```
string s = "nit warangal";

→ s.find("nit")      // 0 index
→ s.substr(1, 2)     // it
```

Begin Index, length

→ Problem: Check if two strings are anagram of each other or not.

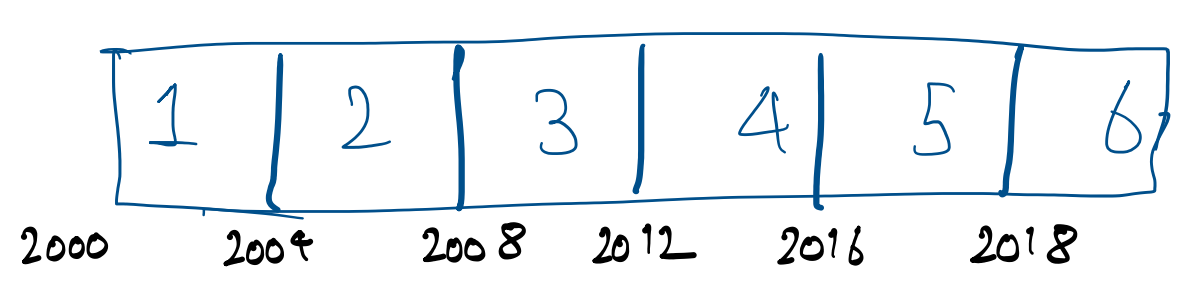
• Matrix

Multi Dimensional Arrays in C++.

A[3][2] = [[1, 2]
[3, 4]
[5, 6]]

2D Array

- C++ follows row major order to store matrix



- Skip internal Curly brackets ✗ Not recommended.
- While passing multi dimensional array to functions
  ↳ only first dimension is allowed to be ommitted.

```
int A[][2] = {{1, 2}, {3, 4}}
int A[][2][2] = {{{1, 2}, {3, 4}},
                 {{5, 6}, {7, 8}}}
```

- C++ 14 allows,

```
int m = 3, n = 2;
int A[m][n];
```

← Variable Sized array Allowed

- Dynamic Allocation

```
int m = 3, n = 2;
int **arr;
arr = new int*[m];

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

→ Double pointer (language Specific for C++)

Adv, → Dynamic Size Array
→ Pass to func. easily

D. Adv, → Not cache friendly

→ Problem : Print Snake pattern for 2D Matrix.

→ Assignment : Pattern related questions as assignment.