Page No.
Date

# $\boxed{\text{MAP} \rightarrow}$ are associative container / and it's non-sequential to, it store the element corresponding to the its key

A key value has it map value. and map value has to be unique according to keys

for example	key value		map vale
	[1]	=	"abc"
	[2]	=	"FGH"
	[1]	=	"abg"

key value has to be unique ~~[illegible]~~ we can't use same key value if we then it will replace the previous map value to new one

-> begin()
-> end()
-> size()
-> max_size()
-> empty()
-> insert (key value, map value)
-> erase (iterator position)
-> clear()

Time complexity of insertion -> O(log(N))
Find/traverse -> O(log(N))

```
            void learn_map()
        {
syntax ->   map<int, string> A;
            A[1] = {"ABC"};
            A[2] = {"DEF"};
            A[4] = {"GHI};
            A[3] = {"JKL"};

            for (auto it : A)
              {
                cout<<It.first<<" "<<it.second;
              }

       output -> A[1]  =  {ABC}
                 A[2]  =  {DEF}
                 A[3]  =  {JKL}
                 A[4]  =  {GHI}
```

Keys has to be always unique and it will be in sorted order after execution in map

Page No. ______
Date

Q1 Give N strings, Print unique string in lexiography order with their frequenly.

```
void solution ()
{
    Map <string, int> A;        <- int N;
                                   cin>>N;
    for (int=i; i<N; i++)
    {
        int S;
        cin>>S;
        A[S]++;
    }
    for (Auto it : A)
    {
    cout<< it->First << it ->second;
    }
}
```

Input
ABC
ABC
EDF
GHI
EDF

Output

ABC	2
EDF	2
GHI	1

we store that in map A[S]

$\boxed{\begin{array}{l} A[S] \rightarrow \text{Key} \\ ++ \rightarrow \text{value} \end{array}}$

Q Leetcode 169
Find majority of element in given arr
if element is appear more the
n/2 (length/2) then it is maj-
ority element

Input
↓
1
1
2
3
1
1

Output
↓
(1)

length = is 6 = 6/2 = 3
1 is appear more than
3 times

```
void solution ( )
{
    map <int, int> A;
    int N;
    cin >>N;
    for (int i=0; i<N; i++)
        {
            int x ;
            cin >>x;
            A[x] ;
            A[x] = A[x] +1;
            if (A[x] > N/2)
            {
                cout << x ;
            }
        }
}
```

Page No.
Date

☆ ☆ ☆ ☆☆

Key point's

① This ordered map has time coplexity of O (log (N)) of ~~every~~ insertion/access operation

If we deal with string so it's time coplexity will affect because strin g can be of any size

② It's internally work on the base of Red - black - Tree (advance data struct)

Inbuit operation

③ You can take any data type / utility class in it like pair, List etc.

④ It key is always unique and sorted

Page No.
Date

- unordered_map → It is same like unordered map all the function are same almost excep few

Differenc: ~~[illegible]~~ are ↓

InBuilt implemention → It work on ~~hashm~~ hashtable
Time complexity → O(1) linear time
valid key - dat tipes → you can't use complex data structue in it like vector, pair, list... etc..

→ key is not in sorted form (unordered)
→ used unordered map if you don't want sorted key because it's faster than map

Page No.
Date

Q Give N strings and Q queries.
In each query you are given a string
print frequency of that string.

```
void Solution ()
{
    unordered_map <~~string~~ , int > A;
    int N;
    cin >> N ;
    for(int i=0; i<N; i++)
    {
        Int S ;
        cin >> S ;
        A [S] ++;
    }

    int Q ;
    cin >> Q ;
    while (Q--)
    {
        int S ;
        cin >> S;
        cout << A[S] ;
        cout << endl;
    }
```

Input	Query	Output
8		
abc	abc	4
DEF	GHI	2
abc		
GHI		
abc		
GHI		
LMN		
abc		

- <u>Multi-map</u>
  multi-map is same like map but difference is in map key are unique but in multimap key is not unique we can' stor same multi value / key