

# DAY 11

06 May 2024 01:01 PM

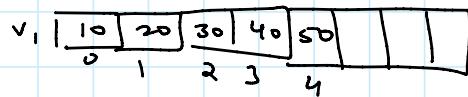
## DSA

STL ✓ vector | ✓ Array | ✓ Stack | ✓ Queue | ✗ List | Set | Map

Set :-

vector :- Dynamic Array

```
vector<int> v1;  
v1.push_back(10);  
v1.push_back(20);  
v1.push_back(30);  
v1._____ (40)  
v1._____ (50)
```

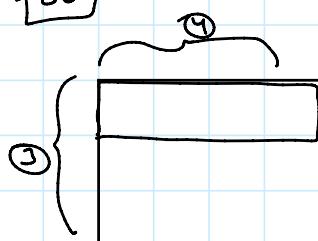
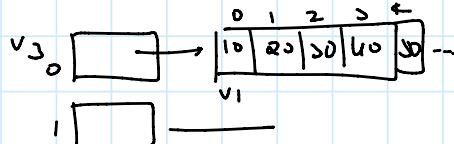


size [ 5 ]  
capacity [ 8 ]

vector<float> v2;

vector<vector<int>> v3;

```
v3.push_back(v1);  
cout << v3[0][2];
```



vector<vector<int>> v4(3, vector<int>(4));

state | array array ← static  
 ↓  
 array <int, 10> a;  
 ↴

int a[10] ✓

set :→ duplicate elements not allowed

#include <set>

set <int> s1;

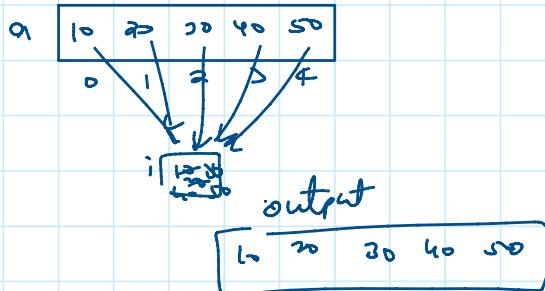
s1.insert(10);  
 s1.insert(50);  
 s1.insert(20);  
 X s1.insert(10); X

s1 [10 20 50]

int a[] = {10, 20, 30, 40, 50};

for each loop

for (int i: a)  
cout << i;



int a[5] = {10, 20, 30};

for (int i: a) ] → 10 20 30 0 0  
cout << i;

```
#include<iostream>
#include<set>
using namespace std;
```

```
void output(set<int> s1)
{
    for(int i:s1)
        cout<<i<<" ";
    cout<<endl;
}
int main()
{
    set<int> s1;
    s1.insert(10);
    s1.insert(50);
    s1.insert(20);
    s1.insert(40);
    s1.insert(10);
    s1.insert(20);
    output(s1);
    if(s1.find(200)==s1.end())
        cout<<"Not found\n";
    else
        cout<<"Found\n";
    s1.erase(10);
    output(s1);
}
```

W.A.P to Remove duplicate element from an Array

נ

```

for (i=0 ; i<n ; i++) {
    flag = true;
    for (j=i+1 ; j<n ; j++) {
        if (arr[i] == arr[j]) {
            flag = false;
            break;
        }
    }
    if (flag)
        arr[<--k] = arr[i];
}

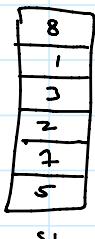
```

line  $\rightarrow O(n^2)$   
space  $\rightarrow O(n)$

```
j = 0  
for(i=0; i < n; i++)  
{  
    if(c[i] < 0) c[i] = 0;  
}
```

int arr<sub>2</sub>[n]

arr<sub>2</sub> [5|7|2|3|1|8] ← x  
j+ → j → j → j → j  
set<int> s<sub>1</sub>;



Time (n)

Space (x n)

for(i=0; i < n; i++)

{ if(s<sub>1</sub>.find(arr<sub>2</sub>[i]) == s<sub>1</sub>.end())

s<sub>1</sub>.insert(arr<sub>2</sub>[i]);

arr<sub>2</sub>[j++] = arr<sub>2</sub>[i];

Multiset :- duplicate elements allow

#include<set>

multiset<int> s<sub>1</sub>;

s<sub>1</sub>.insert(10);  
s<sub>1</sub>.insert(20);  
s<sub>1</sub>.insert(10);

s<sub>1</sub> [10|10|20]

```
#include<iostream>
#include<set>
using namespace std;
void output(multiset<int> s1)
{
    for(int i:s1)
        cout<<i<<" ";
    cout<<endl;
}
int main()
{
    multiset<int> s1;
    s1.insert(10);
    s1.insert(50);
    s1.insert(20);
    s1.insert(40);
    s1.insert(10);
    s1.insert(20);
    output(s1);
    if(s1.find(20)==s1.end())
        cout<<"Not found\n";
    else

```

```

cout<<"Found\n";
s1.erase(10);
output(s1);
}

→ template <typename T1, typename T2>
class pair
{
    T1 first;
    T2 second;
};

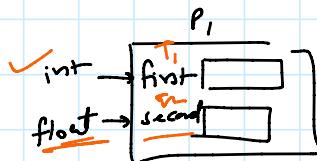
class date
{
public:
    int a;
};

map :-
```

map :-

pair

pair  $\langle \text{int}, \text{float} \rangle$   $P_i;$



`pair <float, string> p2;`

The diagram shows a variable `string` pointing to a memory location containing the text "first" and "second". Another variable `float` is shown pointing to the same memory location, indicating they both reference the same data.

```
class data  
{ public:  
    int a;  
    float b;
```

3

$$\begin{aligned} d_2 \cdot 9 &= 10 \\ \underline{d_1 \cdot 9} &= 10 \\ d_1 \cdot 6 &= 7.5 \end{aligned}$$

$$\begin{array}{r} d_1 \\ \hline 9 \boxed{10} \\ b \boxed{7.5} \end{array}$$

data d<sub>2</sub>;

A rectangle is divided into four quadrants by a horizontal and a vertical line. The top-left quadrant is labeled 'a', the top-right 'b', the bottom-left 'c', and the bottom-right 'd'. A large orange circle surrounds the entire rectangle.

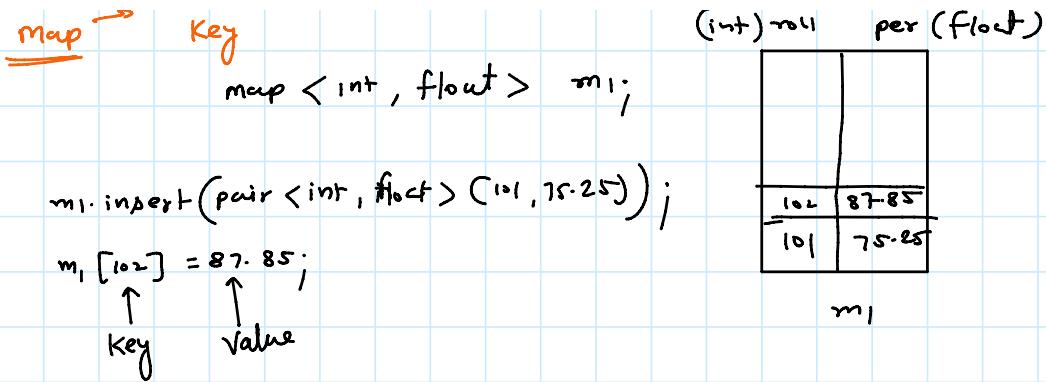
```
#include<iostream>
#include<set>
using namespace std;
void output(multiset<int> s1)
{
    for(int i:s1)
        cout<<i<<" ";
    cout<<endl;
}
int main()
{
    pair<int, int> p1;
    p1.first=101;
    p1.second=8005;
    cout<<p1.first<<' '<<p1.s
    pair<int, float> p2(1001,2
    cout<<p2.first<<' '<<p2.s
```

map → sort According to Key

key must  
be unique → (first) key

value (second)  
↓

(int) roll per (float)



```
#include<iostream>
#include<map>
using namespace std;
void output(map<int, float> m1)
{
    for(pair<int, float> i:m1)
        cout<<i.first<<" "<<i.second<<endl;
    cout<<endl;
}
int main()
{
    map<int, float> m1;
    m1.insert(pair<int, float>(101, 75.28));
    m1.insert(pair<int, float>(103, 95.25));
    m1.insert(pair<int, float>(101, 65.54));
    m1.insert(pair<int, float>(102, 74.14));
    m1[107]=85.14;
    m1[105]=45.54;
    m1[101]=35.25;      //edit
    output(m1);
    map<int, float>::iterator i=m1.find(101);
    if(i==m1.end())
        cout<<"Not found";
    else
        cout<<"Found Value = "<<i->second;
}
```

multi map:— duplicate values allowed

```
#include<iostream>
#include<map>
using namespace std;
void output(multimap<int, float> m1)
{
    for(pair<int, float> i:m1)
        cout<<i.first<<" "<<i.second<<endl;
    cout<<endl;
}
```

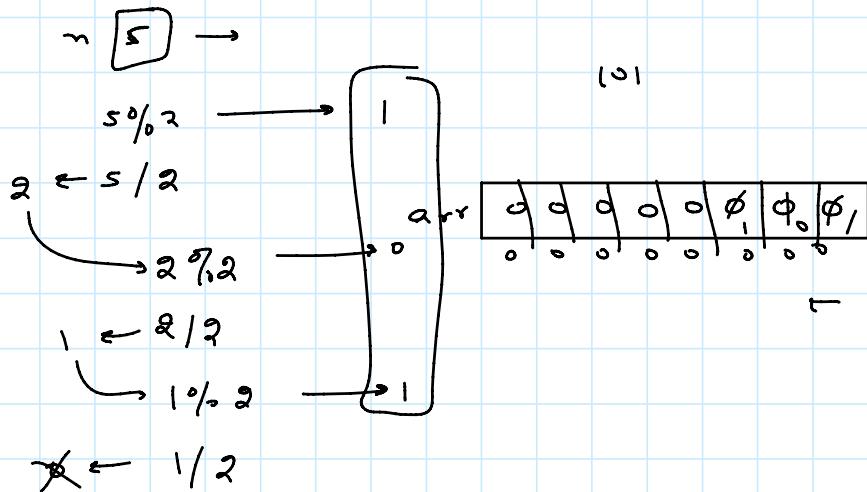
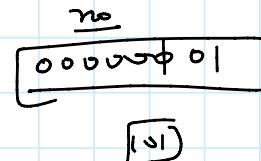
```

int main()
{
    multimap<int, float> m1;
    m1.insert(pair<int, float>(101, 75.28));
    m1.insert(pair<int, float>(103, 95.25));
    m1.insert(pair<int, float>(101, 75.28));
    m1.insert(pair<int, float>(102, 74.14));
    output(m1);
    map<int, float>::iterator i=m1.find(101);
    if(i==m1.end())
        cout<<"Not found";
    else
        cout<<"Found Value = "<<i->second;
}

```

$\ll \gg \leftarrow | \wedge \sim$

print Binary no of a decimal no

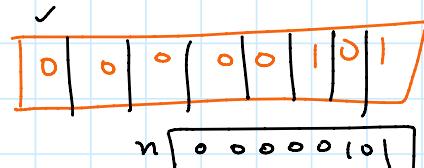
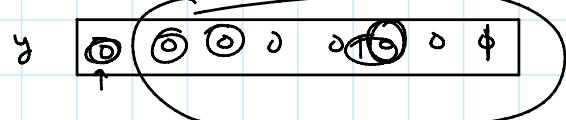
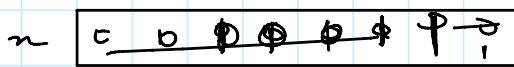
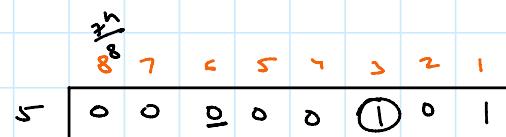


i  $\boxed{74}$   $\leftarrow$   
n  $\boxed{-}$

```

for(i=7; i>=0; i--)
{
    n = 1 << i;
    y = n & n;
    if(y == 0)
        cout << "0";
    else
        cout << "1";
}

```



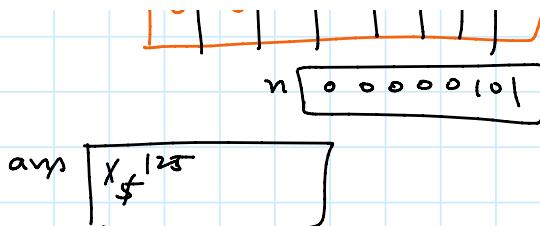
~~n~~  $\boxed{8}$   $p \boxed{3}$

first bit  $\boxed{1}$

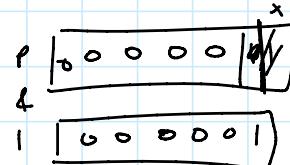
if (first-bit)  
 $\lceil ans = ans * n$

$\underline{n = n * n}$

$p = p \gg 1;$



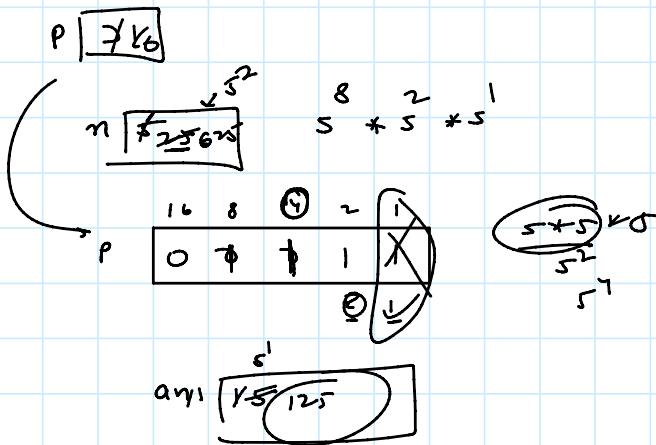
first bit =  $p \gg 1;$



8

first bit \_\_\_\_\_)

```
#include<iostream>
#include<map>
using namespace std;
int mypower(int n, int p)
{
    int ans=1;
    while(p)
    {
        int first_bit=p&1;
        if(first_bit)
        {
            ans *= n;
        }
        n=n*n; → next line
        p=p>>1;
    }
    return ans;
}
int main()
{
    int n=5, p=3;
    cout<<mypower(n,p);
}
```



Time Complexity :-

for ( $i = 1$  to  $n$ )

$n^{\log n}$

for ( i = 0 ;

$O(n)$

The diagram illustrates nested loops. The outer loop is labeled "for( i = n)" with curly braces {}, and the inner loop is labeled "for( i = n)" with curly braces {}, enclosed in a rounded rectangle.

۷۸

$O(n^2)$

The diagram illustrates the execution flow of nested loops. It shows three levels of nesting:

- The outermost loop is labeled `for (i = n)`.
- The middle loop is labeled `for (j = n)`.
- The innermost loop is labeled `for (k = n)`.

Each loop iteration is represented by a vertical segment. The total height of the segments is labeled  $n$ . The overall time complexity is indicated as  $O(n^3)$ .

```
for( i = 0; i < n; i++ )  
    for( j = 0; j < n; j++ )  
        . . .  
        . . .  
        . . .  
        . . .
```

$O(n^n)$

m 8

for( 1 —  $\log_2 n$ )  
{  
} =

$$O(\log_2 n)$$

$$n \boxed{8} \longrightarrow 3$$

$$8 = \underline{2}^P$$

$$P = \boxed{\log_2 8^n}$$

2  
1  
8  
16  
32

$\text{for } (1 \dots n)$   
 {  
    $\text{for } (1 \dots \log_2 n)$   
   {  
 }  
 }  
 ——————

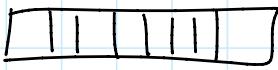
$O(n \log_2 n)$

$\text{if } ( \dots ) \longrightarrow O(1)$   
 ——————  
 else ——————

- $O(1)$
- $O(\log_2 n)$
- $O(n)$
- $O(n \log_2 n)$
- $O(n^2)$
- $O(n^2 \log_2 n)$
- $O(n^3)$
- $O(n^m)$

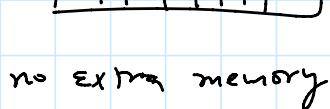
Space Complexity :-

problem  $\rightarrow$



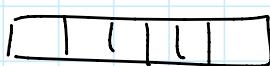
no extra memory

Space  
 $O(1)$



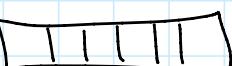
$O(1)$

prob —



$n \square$

arr<sub>1</sub>



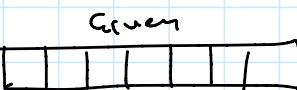
$n \square$

space

$O(n)$

problem

→



$n \square$

space  
 $O(n^2)$

arr<sub>2</sub> [n][n] ← Extra



space

$O(nm)$

for (i = 0)

{

  for (j = 0)

{

  }

time

$O(nm)$