* Int a[2][3]={{1,2,3},{4,5,6}}; -initialisation
* Row value is optional.
* Int a[][3]= {{1,2,3},{4,5,6}};
* Int a[][3]= {{1,2},{4,5,6}}; -prints matrix of 2\*3: 1,2,0; 4,5,6
* If not initialised, all values are 0.
* Initialise with 0: int b[10][10]={{0}};
* a+(i\*c+j)\*4; a-base address, c=total column
* Find element in 2d array: binary search when array is sorted. Can apply in each row and column in 2d array in n log n time.

VECTOR: Inbuilt dynamic array

//vector<int> \* vp = new vector<int>(); -DYNAMIC

vector<int> v; - STATIC CREATION

v.push\_back(10);

v.ar(2); - gives position at 2

v.size();

v.pop\_back(); --deletes last element

v.capacity(); --doubles size

vector<int> v(5,2);

for(int i = 0; i < v.size(); i++){

cout << v[i] << " ";

}

Op: 2 2 2 2 2 vector<int> v(5,2) - This creates a vector with 5 elements and value of every element is 2

vector<int> v(4,12);

v.push\_back(15);

cout << v.size() <<" "<<v.capacity();

op: 5 8 vector<int> v(4,12) - creates a vector with size 4 and capacity 4 and value of every element is 12. After this next element is inserted, which increases the size by 1 and capacity of vector is doubled.