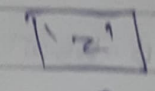
Character arrays and strings

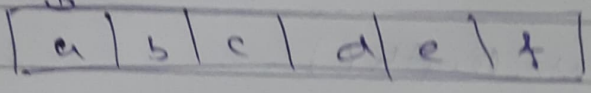
→ char a = 'z';



a

→ Strings in c++ → 1D char. Arrays.

char. Arrays.

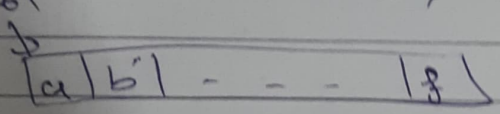
strings → 

→ Character Array

↓

char ch[10];

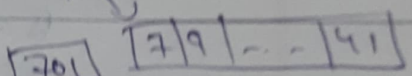
701



701

address

int arr[10]

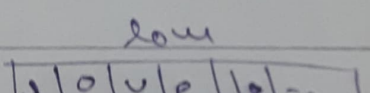


arr.

cin >> n

cin >> arr[i]

i/p → cin >> name;



NULL character used as terminator

↓

ASCII value 0.

o/p → cout << name;

↓

love.

cin → execution stop → space " "
enter "\n"
tab "\t"

string s;

string str = "Babbar";

B	a	b	b	a	r	\0

length → str.length()

str.push_back('c');

Babbarc

str.pop_back()

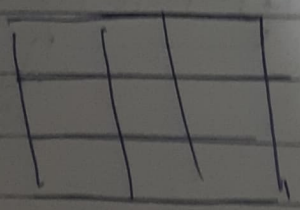
Babba

cin → one stop → space, tab, newline.

for that.

We have to use cin.getline(str, len);

2D Arrays



arr1

arr2

arr3

10 rows, 10 col

arr1

arr10

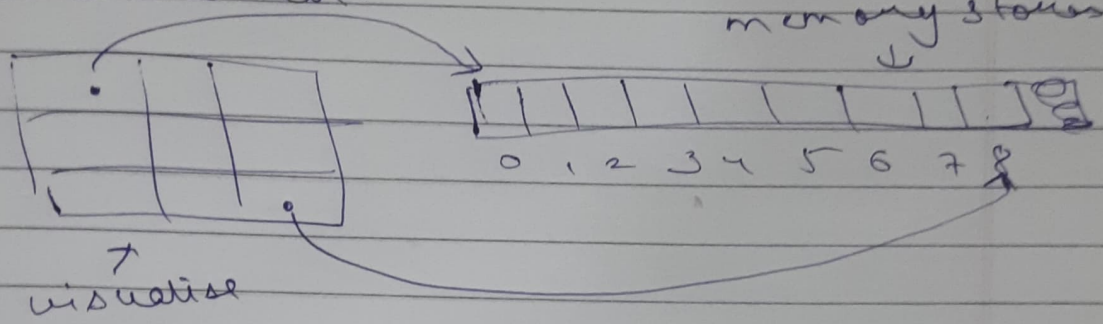
Date

How 2D Array stored in Memory:

DELTA Pg No.

1000 row, 1000 col: It is diff to cater make 1000 arrays. So we will make a linear array.

Like, for 3 row 3 col.



Mathematical formula: $c \times i + j$.

For 1 row 0 col, $= 3 \times 1 + 0$

$= 3$

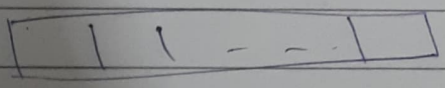
For 2 row, 1 col: $3 \times 2 + 1$

$= 7$.

Creation of 2D Arrays:

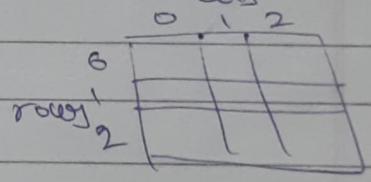
for linear,

1D → int arr[10];



rows cols

int arr[3][3];



input:

1D array:

cin >> arr[i]

2D array:

cin >> arr[i][j]

output:

1D → cout << arr[i]

2D:

cout << arr[i][j];