Strings and 2D Arrays

→ Non-Primitive

#

```
{
    char c='c';
    String str = "abcdefgh";
    System.out.println(str.charAt(1));
    System.out.println(str.length());
    System.out.println(str.substring(1,6));
    System.out.println(str.substring(3));
    System.out.println(str);
    System.out.println(str.substring(0));
}
```

OUTPUT

```
b
8
bcdef
defgh
abcdefgh
abcdefgh
```

```
_____(str.contains("cde"));
___ ___ (___ ("cdf");
```

true

false

~~Public static~~

```
Public static void print(String s){
    for(int i=0; i<s.length() ; i++) {
        System.out.println(s.charAt(i));
    }
}

Public static void main(String[] args){
    String str="abcdefgh";
    print(str);
}
```

str.substring()

#

```
String str1 = "Java is Amazing"
//Implict construction via
  string literal

String str2 = new String("Java is cool");
// Explicit construction via
                    new
```

Count words

this is a sample string

Check Palindrome

i=2
j=2
abcba → Palindrome

① Reverse the string
And then compare

```
#
{
  int i=0;
  int j=str.length()-1;
  while(i<=j) {
    if(str.charAt(i) != str.charAt(j)) {
      return false;
    }
    else {
      i++;
      j--;
    }
  }
  return true;
}
```

words = space + 1;

```
j=str.length-1;
for(i=0; i<-  ; i++)
{

}
```

abba i=1, j=2 — ③

abccba i=2, j=3 ⑥

(j-i=1)

COUNT WORDS —X—

```
#
{
  if(str.length()==0) {
    return 0;
  }
  int count=0;
  for(int i=0; i<str.length(); i++){
    if(str.charAt(i)==' '){
      count++;
    }}
  return (count+1);
}
```

2.

```
String str = "abcdefgh";
str = "abc";
Print(str);
}
```

OUTPUT
abc

setcharAt(1) → No function like this
They are immutable

```
String s = "abc";
String t = "abc";
String to = "abc";
```

Arrays to content compare nhi hoga Address hoga.

arr1 == arr2

```
{ str = "abc";
// Print(str);
  str = str + "def";
  System.out.println(str);
  System.out.println(str.concat("ghi"));
  System.out.println(str);
}
```

abcdef
abcdefghi
abcdef

(str1.equals(str2)) → To compare their CONTENT

```
{String str = "abcde";
 for(int i=0; i<str.length(); i++) {
    system.out.println(str.substring(0, i+1));
 }
}
```

OUTPUT
a
ab
abc
abcd
abcde

Print All substrings ——x——

abc

```
Public class solution {
  Public static void printsubstrings(String str) {
     int val=0;
   While(val<str.length()) {
   for(int j=val ; j<str.length() ; j++){
     System.out.println(str.substring(val, j+1));
   }
     val++;
  }}
}
```

a
ab
abc
b
bc
c

0
01
012
1
12
2

(150)

* Reverse String word Wise ——x——

→ Reverse the string
→ Reverse the word

Always Indent your code
→ Code your Indent Always

```
{ int curr = input.length()-1;
  int per = input.length();
  String output = "";
  While(curr>0) {
    if(input.charAt(curr)==' '){
    output += input.substring(curr+1, prev) + " ";
      prev = curr;
    }
    curr--;
  } output += input.substring(curr, prev);
    return output;
```

100%
good

2D Arrays

int arr[] = new int[10]; → Syntax of 1D Array

int arr2D[][] → Rows, Column

	00	01	02
00	1	2	3
10	4	5	6

```
# Public class Arrayuse {
    Public static void main(String[] args) {
        int arr2d[][] = new int[3][4];
        System.out.println(arr2d[1][2]);
        arr2d[2][0] = 15;
        System.out.println(arr2d[2][0]);
        int arr2[][] = {1,2,3} {4,5,6};
    }
}
```

OUTPUT

0
15

```
# {  Scanner s = new Scanner(System.in);
     System.out.println("Enter no. of rows");
     int rows = s.nextInt();
     System.out.println("Enter no. of column");
     int cols = s.nextInt();
     int input[][] = new int[rows][column];
     for(int i=0; i<rows; i++){
        for(int j=0; j<cols; j++) {
           system.out.println("Enter element at "+i+" row "+j+" column");
           input[i][j] = s.nextInt();
        }
     }
     for(int i=0; i<rows; i++) {
        for(j=0; j<cols; j++) {
           System.out.print(input[i][j] + " ");
```

~~[illegible]~~ —x—

System.out.println();

}

}

}

[[I@7852c922

Two boxes means 2D Array

Integer Type

ROW WISE SUM —x—

Two dimensional Integer array (N x M)

00 + 01 + 02

Public class solution {

Public static void rowWisesum(int[][] mat) {

for(int i=0; i<mat.length ; i++) {

int sum=0;

for(int j=0;j<mat.[0].length ;j++) {

sum= sum + mat[i][j] ;

}

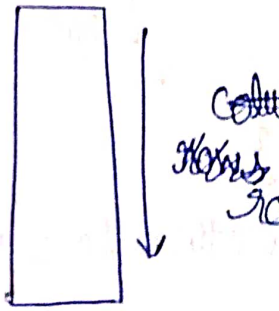
System.out.print(sum+" ");

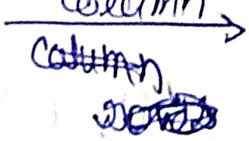
}

}

}

ROW WISE





LARGEST Column SUM

for(int i=0; i<mat.length ; i++) {

max= mat[i,0]

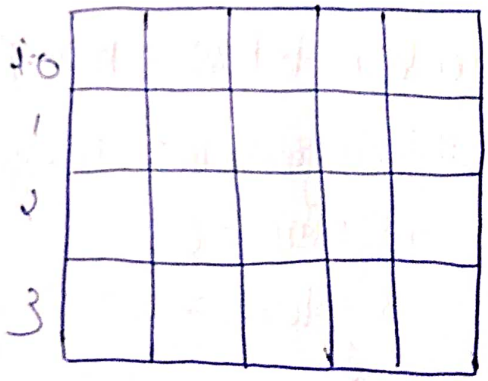
for(int j=0 . . .) {

if(max < i . . .)

{

}

. . . .out.print(max)



LARGEST Column Sum

```
# Public static int largestColSum(int input[][])
{
    int largest = Integer.MIN_VALUE;
    int rows = input.length;
    int cols = input[0].length;
    for(int j=0; j<cols; j++){
        sum=0;
        for(int i=0; i<rows; i++){
            sum= sum+ input[i][j];
        }
        if(sum>largest){
            largest=sum;
        }
    }
    return largest;
}
```

1	2	3
4	5	6

~~00 01 02~~

00	01	02
10	11	12
20	21	22

LARGEST ROW or column

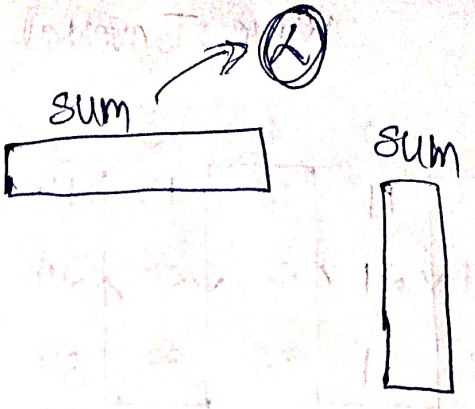
```
F {
    boolean isRow = true;
    int largestSum = Integer.MIN_VALUE;
    int num = 0;
    int nRows = mat.length;
    if(nRows==0){
        System.out.println("row" + num + " " + largestSum);
        return;
    }
    int nCols = mat[0].length;
```

(2,2)

1	1
1	1

3	6	9
1	4	7
2	8	9

```
for(int i=0 ; i<nRows ; i++) {
    int rowsum = 0 ;
    for(int j=0 ; j<mcols ; j++){
        rowsum = rowsum + mat[i][j] ;
    }
    if(rowsum > largestsum) {
        largestsum = rowsum ;
        num = i ;
    }
}
for(int j=0 ; j< mCols ; j++){
    int colsum = 0 ;
    for(int i=0 ; i< nrows ; i++) {
        colsum = colsum + mat[i][j] ;
    }
    if(colsum > largestsum) {
        largestSum = colsum ;
        num = j ;
        isRow = false ;
    }
}
if(isRow) {
    System.out.println("row" +"num"+" " + largestsum) ;
}
else {
    System.out.println("column" + num + " " + largestsum) ;
}
}
```



< is larger than

Wave Traversal

	0	1	2	3
0	11	12	13	14
1	21	22	23	24
2	31	32	33	34

column ↑

even — row ↑

odd — row ↓

```
{
if(arr.length==0) {
    return ;
}
```

#

```
for(int J=0 ; J<arr[0].length ; J++)
{ if(J%2==0){
  for(int I=0; I<arr.length ; I++) {
   System.out.println(arr[I][J]) ;
  }
else {
for(int I= arr.length-1 ; I>=0 ; I--) {
 System.out.println(arr[I][J]);
}
}
```

SPIRAL TRAVERSAL

minr, minc

11	12	13	14	15	16	17
21	22	23	24	25	26	27
31	32	33	34	35	36	37
41	42	43	44	45	46	47
51	52	53	54	55	56	57

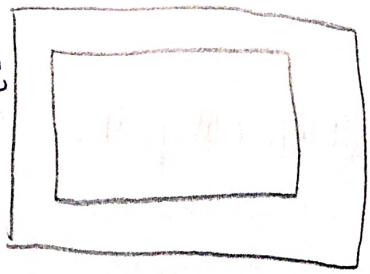
(minr, maxc)

(maxr, minc)

maxr, maxc

ROW

Column



#

```
int minr= 0;
int minc =0;
int maxr= arr.length -1;
int maxc = arr[0].length -1; int tne= n*m ; int cnt = 0;
while (cnt<tne){
for(int i = minr, j=minc; i<=maxr && cnt<tne ; i++)
{ System.out.println(arr[i][j]);
  cnt++;
} minc++;
for(int i = maxr; j= minc ; j<=maxc && cnt <tne ; j++)
{ system.out.println(arr[i][j]);
  cnt++;
} maxr--;
for(int i = maxr, j= maxc ; i>= minr && cnt<tne ; i--)
{ System.out.println(arr[i][j]);
  cnt++;
} maxc--;

for(int i = minr , j= maxc ; j>= minc && cnt< tne ; j--)
{ System.out.println(arr[i][j]);
  cnt++;
} minr++;

}
```

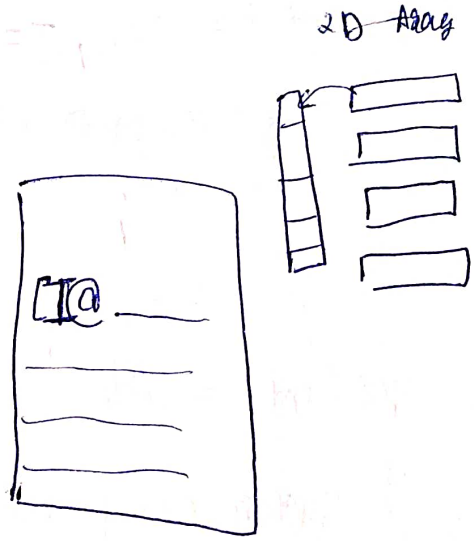
JAGGED ARRAYS

#
```
{
    int arr[][] = new int[3][5];
    System.out.println(arr[0]);

    int arr2[][] = new int[4][];
    System.out.println(arr2[0]);
    for(int i=0; i<arr2.length; i++){
        System.out.println(arr2[i]);
    }
}
```

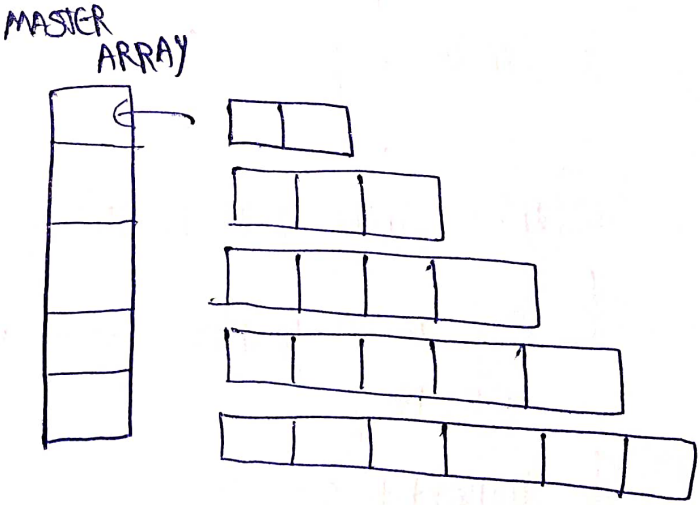
OUTPUT

```
[I@7852e922
[[I@4e25154f
null
null
null
null
```

```
for(int i=0; i<arr.length; i++) {
    arr2[i] = new int[2];
    System.out.println(arr2[i]);
}
```




```
{ int arr2[][] = new int[4][0];
  System.out.println(arr2);
for(int i=0; i<arr2.length; i++){
    arr2[i] = new int[i+2];
}
```




```
for(int i=0; i<arr2.length; i++) {
    for(int j=0; j<arr2[i].length; j++) {
        System.out.print(arr2[i][j] + " ");
    }
    System.out.println();
}
```

OUTPUT

```
[[I@
0 0
0 0 0
0 0 0 0
0 0 0 0 0
```

Check Permutation

str1	str2
sinrtg	strlng

- str1 and str2 Both have equal no. of characters

```
#  {
   if(str1.length() != str2.length())
   { return false;
   }
  else {
   char s1[] = str1.toCharArray();
   char s2[] = str2.to.CharArray();
Arrays.sort(s1);
Arrays.sort(s2);
for(int i=0; i<s1.length; i++)
{ if(s1[i] != s2[i]) {
    return false;
   }
 }
 return true;
}
```

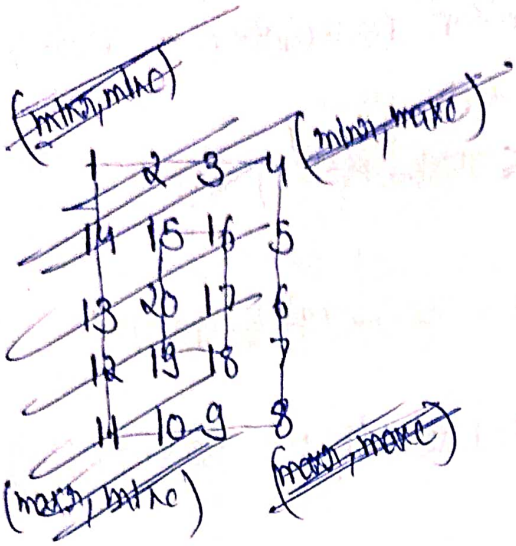
boolean Permutation = TRUE

abedee abccde

```
# ( count=0;
   boolean Permutation = True;
     z= str1.length-1;
for(i=0; i<str1.length(); i++) {

for(j=0; j< str2.length(); j++) {

   if(str1.charAt(i) == str2charAt(j))
   {
      count++;

   }
 }
if(count == z) {
   return true;
  }
else {
   return false;
   }
```

~~Remove Consecutive Duplicates.~~

~~STRAX Print~~



"aaaa"

→ ~~ab~~

aabccbaa

abcba

str

REMOVE CONSECUTIVE DUPLICATES

```
String s = " ";
for(int i=0; i<str.length; i++)
{ for(int j=i+1; j<str.length; j++)
  { if(str[i] != str[j])
    s = s + str.substring(i);
```

~~for(int i=0; i<str.length; i++) {~~

```
# String n = " ";
  char prevchar = ' ';
for(int i=0; i<s.length(); i++)
{ if(prevchar != s.charAt(i))
      n = n + s.charAt(i);
    prevchar = s.charAt(i);
}
return n;
```

charAt(i) ✗
s.charAt(i) ✓

Reverse Each WORD

~~int count~~

Hello, I am Aadil!

"olleH I ma !ldaA"

```
{
    String ans = "";
    int i = 0; currentWord = 0;
    for( ; i < str.length(); i++ ) {
      if(str.charAt(i) == ' ') {
        // Reverse current word
        int currentWordEnd = i-1;
        String reversedWord = "";
        for(int j = currentWordStart; j <= currentWordEnd; j++){
          reverseWord = str.charAt(j) + reverseWord;
        }
        ans = ans + reverseWord + " ";
        currentWordStart = i+1;
      }
    }
    int currentWordEnd = i-1;
    String reversedWord = "";
    for(int j = currentWordStart; j <= currentWordEnd; j++){
      reversedWord = str.charAt(j) + reversedWord;
    }
    ans += reversedWord;
    return ans;
}
```

HOW TO REV A STRING

Remove character.

Hello

abc def ghi

cba fed ihg

```
String ans = "";
int i, start = 0;
for(i=0; i<str.length; i++) {
    End = i; String reverse = "";
{ if(str.charAt(i)==' ') {
    for(int j=start; j<=End; j++) {
      reverse = str.charAt(j) + reverse;
    }

    ans = ans + reverse + " ";
    start = i+1
}

start =
   End = j-1; str.reverse = "";
for(j=start; j<= End; j++)
{
    reverse = str.charAt(j) + reverse;
}
ns :
```

aabccbaa

STRING IS IMMUTABLE

```
for(int j=0; i<str.length(); j++)
{ if(str.charAt(i) == 'x')
  {
    str.charAt(i) = str.charAt(j+1)
  }
    j++;
}

if(str.charAt(str.length()-1) == x)
{

   REMOVE CHARACTER —
#
{
   String output = "";
   for(int i=0; i<str.length(); i++)
   { if(str.charAt(i) != ch)
     {
     output = output + str.charAt(i)
     }
   } return output;
}
```

Highest Occuring Character

"abcdeapapqarr"

a = 3
b = 1

#
```
{
  int frequency[] = new int[256];
  int max = 0;
  char index = ' ';
  for(int i=0; i<256; i++)
  {
    frequency[i] = 0;
  }

  for(int i=0; i<str.length(); i++)
  {
    char ch = str.charAt(i);
    int a = (int) ch;
    frequency[a]++;
  }

  for(int i=0; i<256; i++)
  {
    if(freq[i] > max)
    {
      max = frequency[i];
      index = (char) i;
    }
  }
  return index;
}
```

REMOVE SPACES FROM A STRING

```
{
  String s = "this is demo abc
  s = s.replaceAll("\\s", "");
  System.out.println(s);
}
```

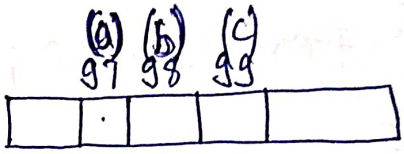
OUTPUT
thisisdemoabc

#

```
{
    int [] arr = new int [128] ;

    for(int i=0 ; i<s.length() ; i++)
    {
      arr[s.charAt(i)] = arr[s.charAt(i)]+1 ;
    }

    int max = -1 ;
    char c = ' ' ;

for(int i=0; i<s.length() ; i++)
{
    if(max < arr[s.charAt(i)])
    { max = arr[s.charAt(i)] ;
      c = s.charAt(i) ;
    }
}

return c ;
```

arr[b] ↕ arr[98]

APNA AAP ASCII MA CONVERT KAR LA GA:



COMPRESS A STRING

aaab

~~count = 2~~

a

Prev = a a a

count = ~~2~~ 3

~~a + 3~~

a3b

prev.char = charAt(str)

aaaabbdddghha

a4b2d3g h2a

```
for(int i=0 ; i<str.length() ; i++)
  if(prevchar = char.
```

#
```
{ int count = 1;
  str = str + (1 "";
  String s = "";
  char prev = str.charAt(0);
  for(int i = 1; i < str.length(); i++) {
   if(prev == str.charAt(i)) {
       count ++;
   }
   else if(i == str.length()-1)
   {
     if(count != 1) {
       s = s + prev + count;
     }
     else {
       s = s + prev;
     }
  }
  else {
    if(count != 1) {
      s = s + prev + count;
    }
    else {
      s = s + prev;
    }
    count = 1
  }
  prev = str.charAt(i);
} return s;
```