Day-12

Q) Print all subarrays of a given array

[1,3,1,5,8] N=5

[1], [3], [1], [5], [8] ⇒ subarrays of size 1 (5)

[1,3], [3,1], [1,5], [5,8] ⇒ subarrays of size 2 (4)

[1,3,1], [3,1,5], [1,5,8] ⇒ subarrays of size 3 (3)

[1,3,1,5] [3,1,5,8] → subarrays of size 4 (2)

[1,3,1,5,8] ⇒ subarray of size 5 (1)

Q) Given an array of size of N, how many subarrays does it have? [a0, a1, a2, ..., an-3, an-2, an-1]

No of subarrays = 1+2+3+4+5 = 15

$\frac{n(n+1)}{2} = \frac{5(6)}{2} = 15$ .....

Size N → 1
Size N-1 → 2
Size N-2 → 3
}
Size 1 → N

1) start_index & length
2) end_index & length
3) Start_index & end_index

Q) Print all subarrays of a given array

[1,3,1,5,8] N=5

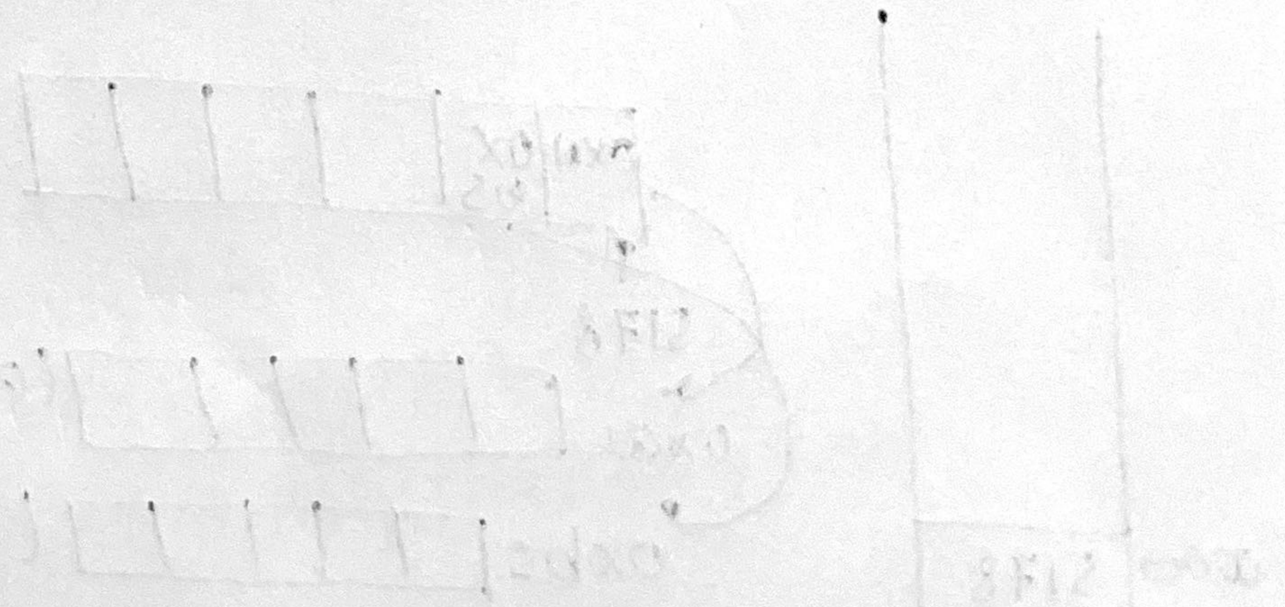
st=0 [1]en=0 [1,3]en=1 [1,3,1]en=2 [1,3,1,5]en=3 [1,3,1,5,8]en=4

st=1 [3]en=1 [3,1]en=2 [3,1,5]en=3 [3,1,5,8]en=4

st=2 [1]en=2 [1,5]en=3 [1,5,8]en=4

st=3 [5]en=3 [5,8]en=4

st=4 [8]en=4



```
import java.util.Scanner;
class classA {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
// Taking I/P
        int N = sc.nextInt();
        int arr[] = new int[N];
        for(int i=0; i<N; i++) {
            arr[i] = sc.nextInt();
        }
// Print all the subarrays
        for (int st=0; st<N; st++) {
            for (int en=st; en<N; en++) {
                for (int i=st; i<=en; i++) {
                    S.o.pln(arr[i]+" ");
                } S.o.pln();
            }
        }
    }
}
```

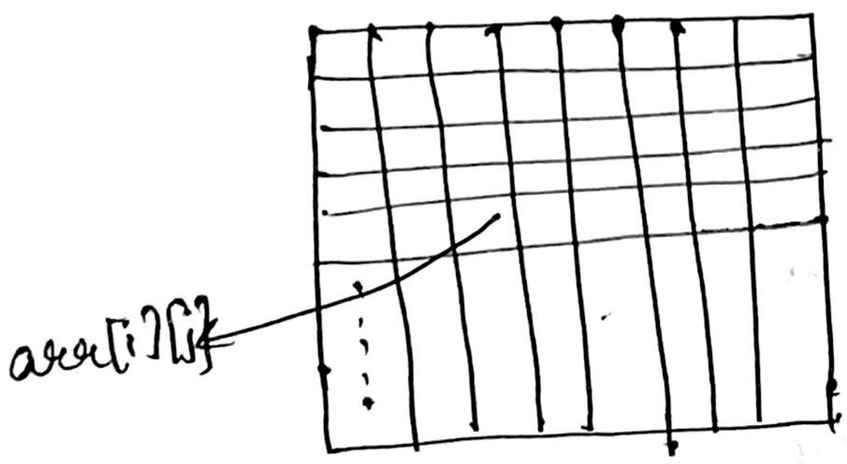
I/P

5
1 3 1 5 8

O/P

```
1
1 3
1 3 1
1 3 1 5
1 3 1 5 8
3
3 1
3 1 5
3 1 5 8
1
1 5
1 5 8
5 8
8
```

*MultiDimensional Arrays



Declaration

<datatype> <array_name>[][] = new
<datatype> [<rows>][<cols>]

→ Declaring and initializing at the same time

```
int arr[][] = {
    {1,2,3,4},
    {5,6,7,8},
    {9,10,11,12}
};
```

i (row indices) ⇒ 0 to R-1
j (col indices) ⇒ 0 to C-1

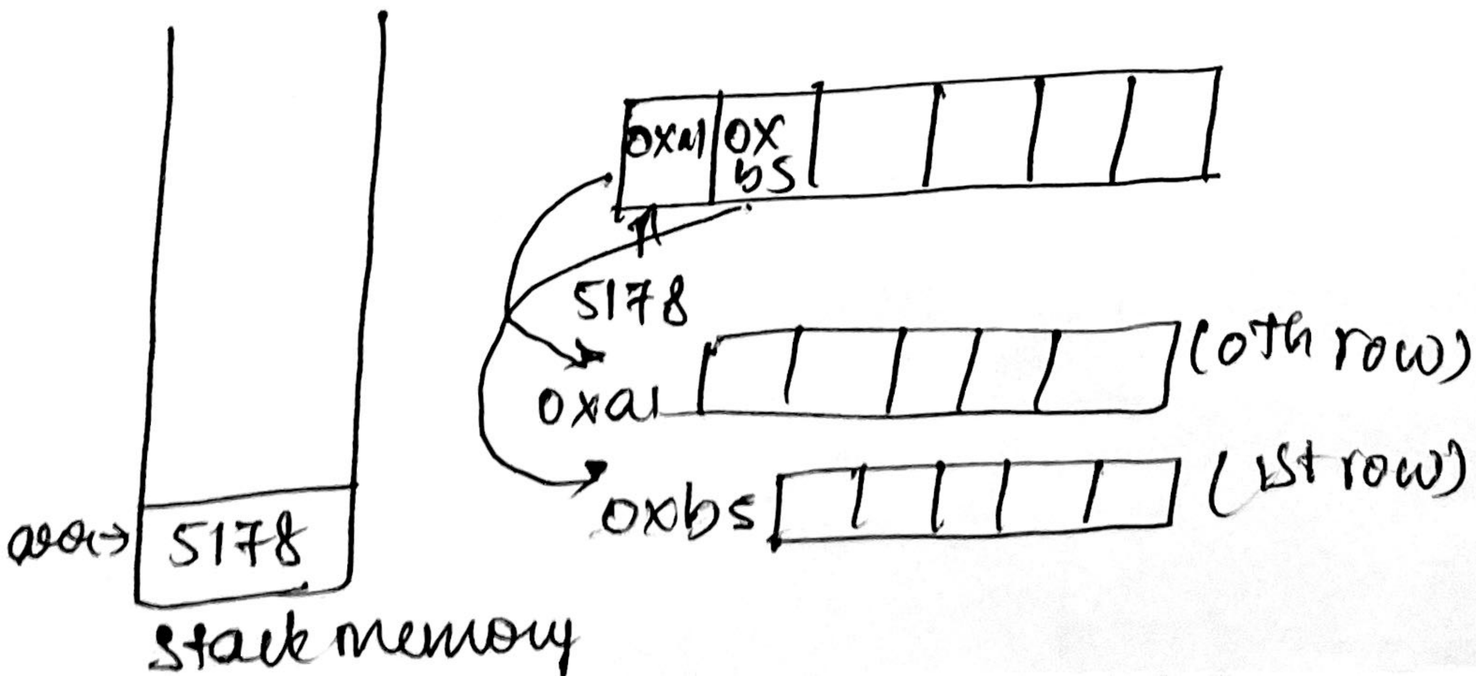
ith row, jth column ⇒ arr[i][j]

Accessing the elements

indices ⇒ 0 to n-1
Rows ⇒ R
Columns ⇒ C
no of elements ⇒ R*C

Can diff rows have diff sizes?
Yes

• A bit about how is memory allocated?



```
class classA {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
// Taking I/P
        int N = sc.nextInt();
        int arr[] = new int[N];
        for(int i=0; i<N; i++) {
            arr[i] = sc.nextInt();
        }
// Print all the subarrays
        for (int st=0; st<N; st++) {
            for (int en=st; en<N; en++) {
                for (int i=st; i<=en; i++) {
                    S.o.pln(arr[i] + " ");
                } } } S.o.pln();
```

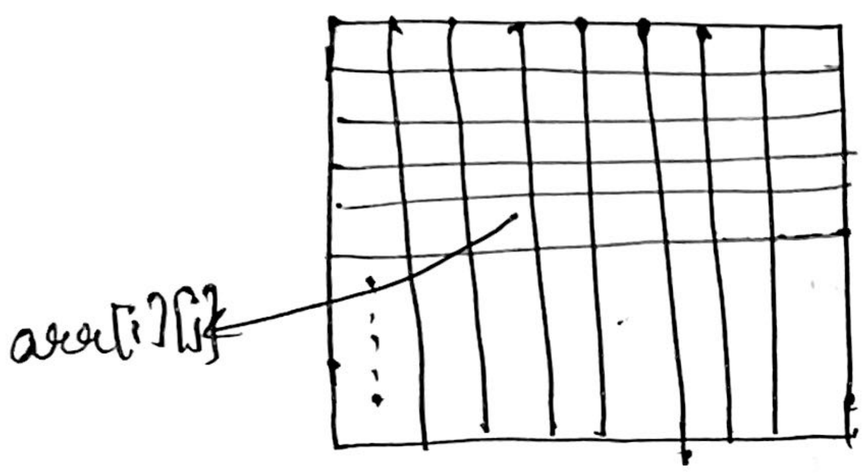
I/P

5
1 3 1 5 8

O/P

1
1 3
1 3 1
1 3 1 5
1 3 1 5 8
3
3 1
3 1 5
3 1 5 8
1
1 5
1 5 8
5 8
8

* Multi Dimensional Arrays



Declaration

<datatype> <array_name>[][] = new
<datatype> [<rows>][<cols

→ Declaring and initializing at the same time

```
int arr[][] = {
        {1,2,3,4},
        {5,6,7,8},
        {9,10,11,12}
};
```

i (row indices) ⇒ 0 to R-1
j (col indices) ⇒ 0 to C-1

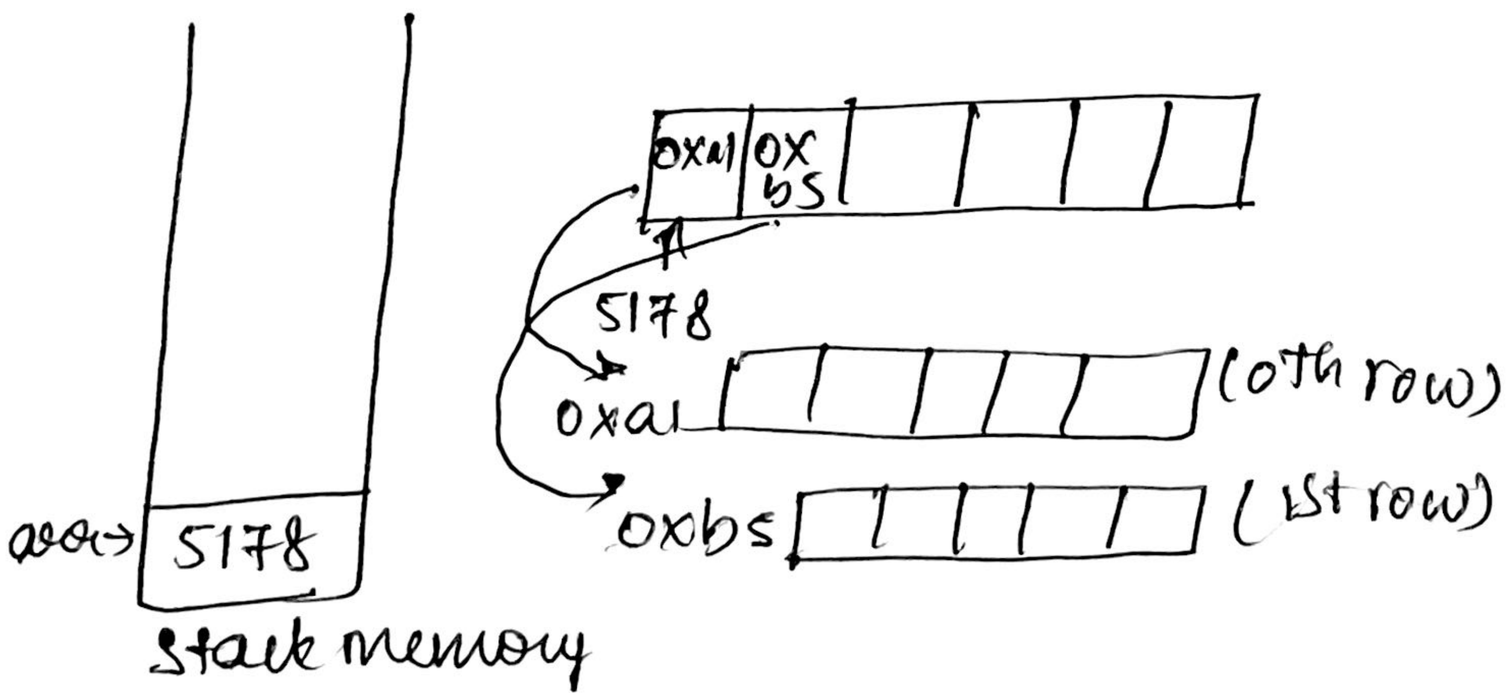
ith row, jth column ⇒ arr[i][j]

Accessing the elements
indices ⇒ 0 to n-1
Row C ⇒ R
Columns ⇒ C

no of elements ⇒ R * C

Can diff rows have diff sizes?
Yes

. A bit about how is memory allocated!



```
int arr[][] = new int[5][];
              arr[0] = new int[3];
              arr[1] = new int[4];
              arr[2] = new int[2];
                 :
                 ;
```

(Jagged Array)

Eg:
```
class ClassA {
    public static void main(String[] args) {
        int arr[][] = new int[6][3];
        S.o.pln(arr.length + "rows.");
        S.o.pln(arr[0].length + "colcemns.");

int brr[][] = {
        {1,2,34},
        {4,5,6},
        {4,5,6,10}
    };      {7,8,9,10}   // Row-wise traversal
for(int i=0; i<brr.length; i++){
    for(int j=0; j< brr[i].length; j++){
        S.o.pln(brr[i][j] + " ");
        S.o.pln();
    }
S.o.pln(brr[1][2]);
}
}
```

o/p
6 rows
3 cols
1 2 3
4 5 6

```
// Jagged array using new keyword
int jag[][] = new int[3][];
    jag[1] = new int[5];
    S.o.pln(jag[0]);
    S.o.pln(jag[1] + " " + jag[1].length);
    S.o.pln(jag[2]);
```

```
int crr[][] = {
    {1,2,3,4},
    {5,6,7,8},
    {9,10,11,12}
};
    // Column-wise traversal
for(int j=0; j< crr[0].length; j++){
    for(int i=0; i< crr.length; i++){
        S.o.pln(crr[i][j] + " ");
    }
}
S.o.pln();
}
}
```

int arr[][] = new int[5][];

arr[0] = new int[3];
arr[1] = new int[4];
arr[2] = new int[2];
:
;

(Jagged Array)

Eg: class ClassA {
public static void main(String[] args) {
int arr[][] = new int[6][3];
S.o.pln(arr.length + "rows.");
S.o.pln(arr[0].length + "colcemns.");

int brr[][] = {
{1,2,34},
{4,5,6},
{7,8,9,10}
}; // Row-wise traversal

for(int i=0; i<brr.length; i++) {
for(int j=0; j< brr[i].length; j++) {
S.o.pln(brr[i][j] + " ");
S.o.pln();
}
S.o.pln(brr[1][2]);

}
}

o/p
6 rows
3 cols
1 2 3
4 5 6

// Jagged array using new keyword
int jag[][] = new int[3][];
jag[1] = new int[5];
S.o.pln(jag[0]);
S.o.pln(jag[1] + " " + jag[1].length);
S.o.pln(jag[2]);

int crr[][] = {
{1,2,3,4},
{5,6,7,8},
{9,10,11,12}
};

// Column-wise traversal
for(int j=0; j< crr[0].length; j++) {
for(int i=0; i< crr.length; i++) {
S.o.pln(crr[i][j] + " ");
}
S.o.pln();
}
}