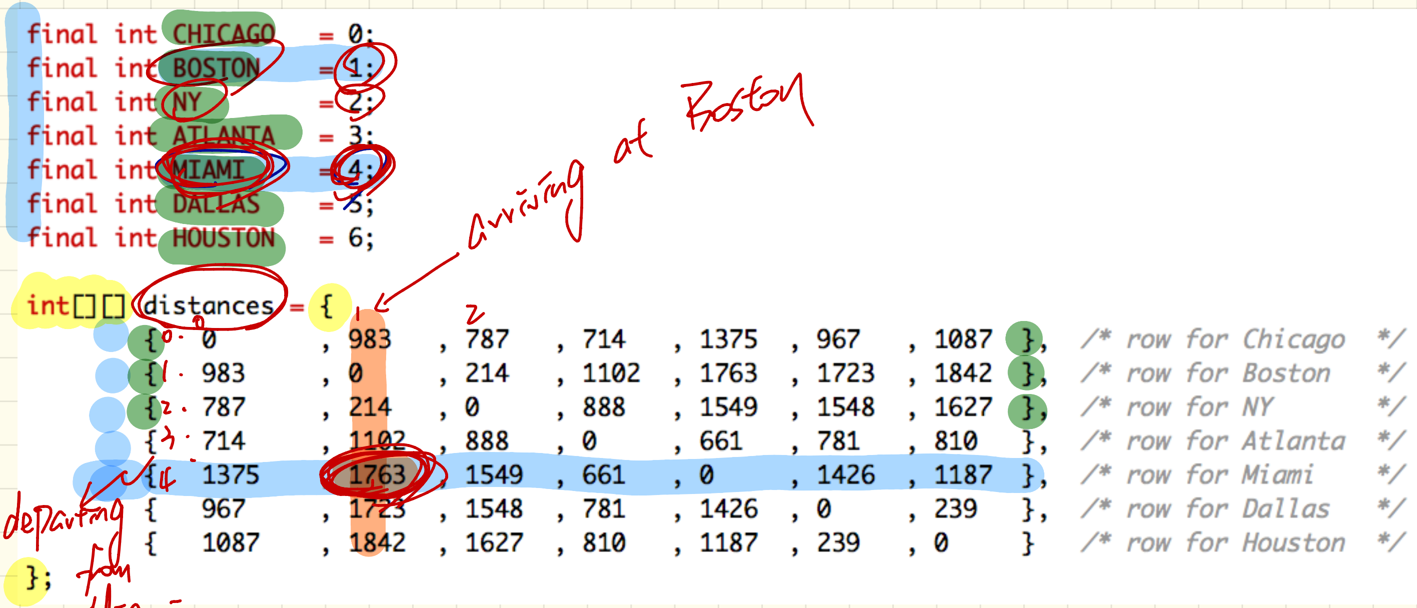
Monday March 19

Lecture 10

Motivating Example Solution : Part I



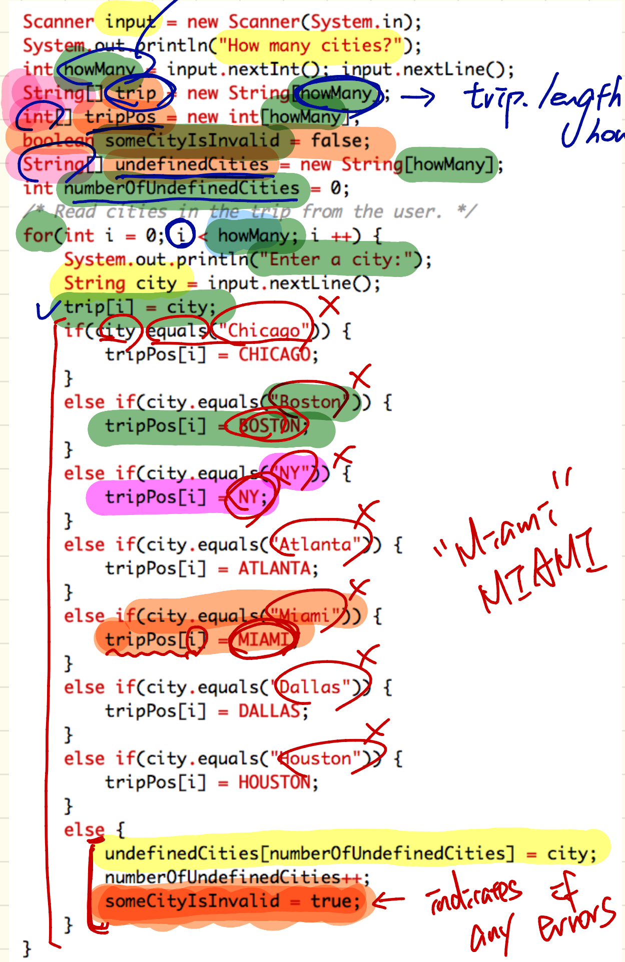
```
final int CHICAGO  = 0;
final int BOSTON   = 1;
final int NY       = 2;
final int ATLANTA  = 3;
final int MIAMI    = 4;
final int DALLAS   = 5;
final int HOUSTON  = 6;

int[][] distances = {
    {    0  , 983  , 787  , 714  , 1375 , 967  , 1087 }, /* row for Chicago  */
    {  983  , 0    , 214  , 1102 , 1763 , 1723 , 1842 }, /* row for Boston   */
    {  787  , 214  , 0    , 888  , 1549 , 1548 , 1627 }, /* row for NY       */
    {  714  , 1102 , 888  , 0    , 661  , 781  , 810  }, /* row for Atlanta  */
    { 1375  , 1763 , 1549 , 661  , 0    , 1426 , 1187 }, /* row for Miami    */
    {  967  , 1723 , 1548 , 781  , 1426 , 0    , 239  }, /* row for Dallas   */
    { 1087  , 1842 , 1627 , 810  , 1187 , 239  , 0    }  /* row for Houston  */
};
```

Q1. How to look up distance between "Miami" to "Boston"? 1763

Q2. How to calculate distances for itinerary {"Miami", "Boston", "NY"}?

? ?

Motivating Example Solution : Part 2



```
Scanner input = new Scanner(System.in);
System.out.println("How many cities?");
int howMany = input.nextInt(); input.nextLine();
String[] trip = new String[howMany];
int[] tripPos = new int[howMany];
boolean someCityIsInvalid = false;
String[] undefinedCities = new String[howMany];
int numberOfUndefinedCities = 0;
/* Read cities in the trip from the user. */
for(int i = 0; i < howMany; i ++) {
    System.out.println("Enter a city:");
    String city = input.nextLine();
    trip[i] = city;
    if(city.equals("Chicago")) {
        tripPos[i] = CHICAGO;
    }
    else if(city.equals("Boston")) {
        tripPos[i] = BOSTON;
    }
    else if(city.equals("NY")) {
        tripPos[i] = NY;
    }
    else if(city.equals("Atlanta")) {
        tripPos[i] = ATLANTA;
    }
    else if(city.equals("Miami")) {
        tripPos[i] = MIAMI;
    }
    else if(city.equals("Dallas")) {
        tripPos[i] = DALLAS;
    }
    else if(city.equals("Houston")) {
        tripPos[i] = HOUSTON;
    }
    else {
        undefinedCities[numberOfUndefinedCities] = city;
        numberOfUndefinedCities++;
        someCityIsInvalid = true;
    }
}
```

3

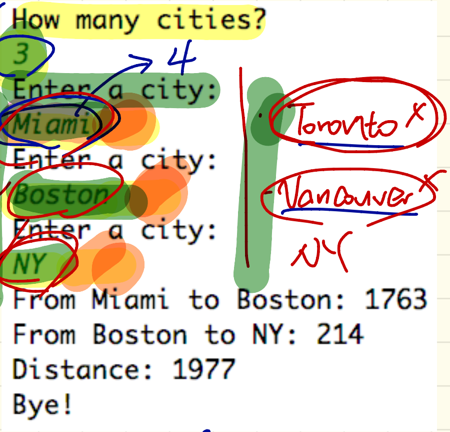
→ trip.length == howMany

"Miami"

MIAMI

indicates if any errors

Console:



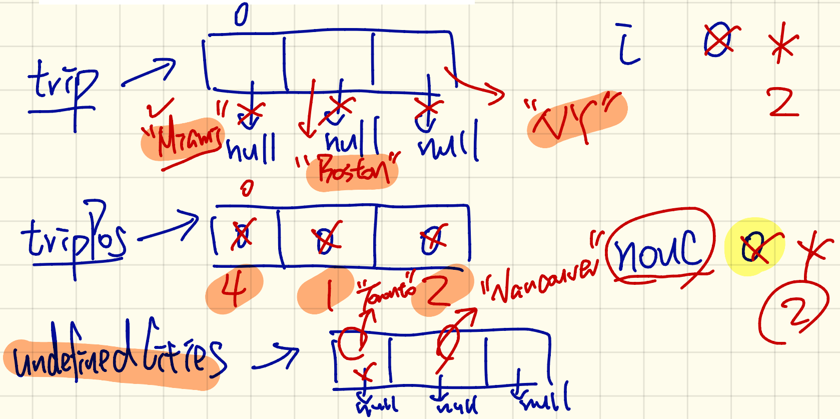
```
How many cities?
3
Enter a city:
Miami
Enter a city:
Boston
Enter a city:
NY
From Miami to Boston: 1763
From Boston to NY: 214
Distance: 1977
Bye!
```

4

Toronto ✗

Vancouver ✗

NY



Motivating Example Solution : Part 3

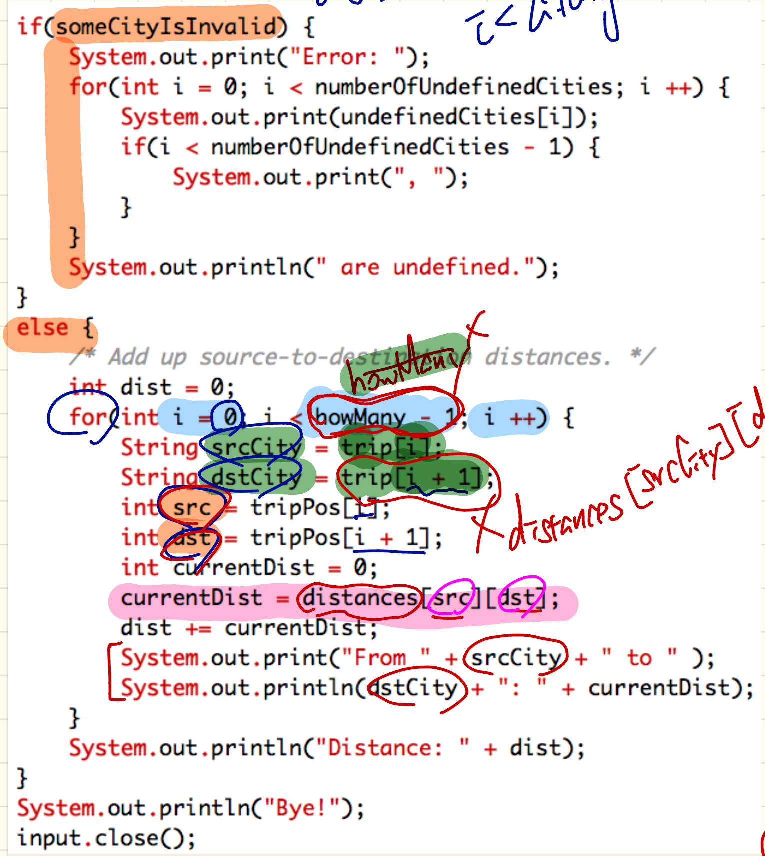
isSorted

a[i] <= a[i+1]

i < a.length-1

i	srcCity	dstCity	src	dst
0	"Miami"	"Boston"	4	1

1763

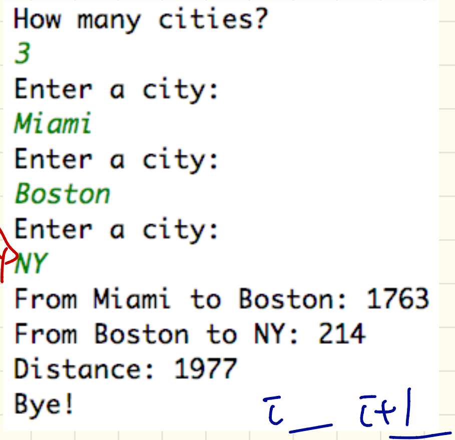


```
if(someCityIsInvalid) {
    System.out.print("Error: ");
    for(int i = 0; i < numberOfUndefinedCities; i ++) {
        System.out.print(undefinedCities[i]);
        if(i < numberOfUndefinedCities - 1) {
            System.out.print(", ");
        }
    }
    System.out.println(" are undefined.");
}
else {
    /* Add up source-to-destination distances. */
    int dist = 0;
    for(int i = 0; i < howMany - 1; i ++) {
        String srcCity = trip[i];
        String dstCity = trip[i + 1];
        int src = tripPos[i];
        int dst = tripPos[i + 1];
        int currentDist = 0;
        currentDist = distances[src][dst];
        dist += currentDist;
        System.out.print("From " + srcCity + " to " );
        System.out.println(dstCity + ": " + currentDist);
    }
    System.out.println("Distance: " + dist);
}
System.out.println("Bye!");
input.close();
```

howMany

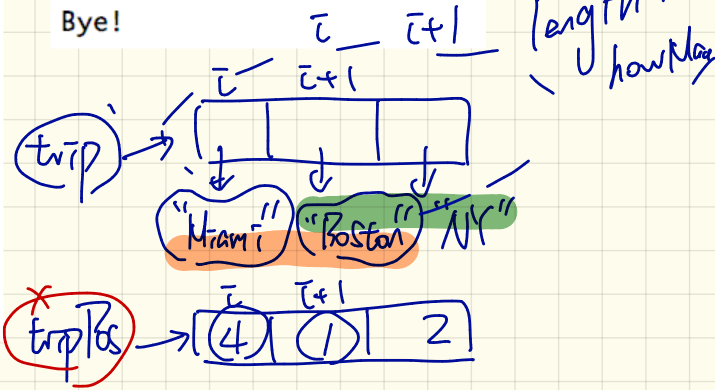
distances[srcCity][dstCity]

Console:



```
How many cities?
3
Enter a city:
Miami
Enter a city:
Boston
Enter a city:
NY
From Miami to Boston: 1763
From Boston to NY: 214
Distance: 1977
Bye!
```

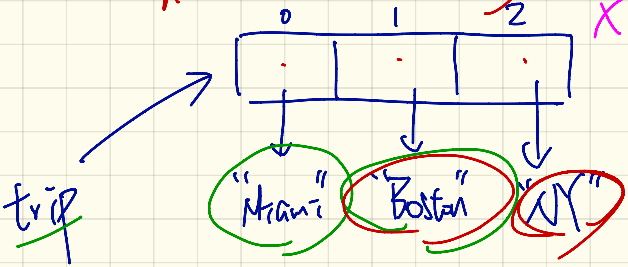
length: howMany



```
String[] trip = new String[howMany];
  ⋮
                          x         howMany - 1
for( int i = 0;  i < howMany;  i++) {
    String srcCity = trip[i];
    String dstCity = trip[i+1];
      ⋮
}
```

howMany == trip.length

3

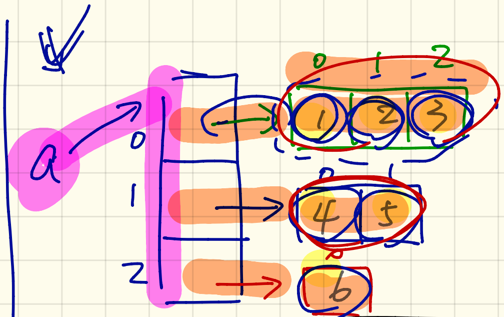


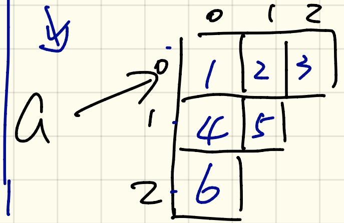
i	i+1	srcCity	dstCity
0	1	"Miami"	"Boston"
x 1	2	"Boston"	"NY"
2	3	"NY"	trip[3]

IndexOutOfB.-E.

✓ ✓

```
int[][] a = {
    {1, 2, 3},
    {4, 5},
    {6}
};
```





a~~[r]~~.length

```
for(int r=0; r< ____ ;r++){
    for(int c=0; c< ____ ; c++){
    }
}
```

a[r].length

# of rows: a.length 3

a[0][0] a[0][1] a[0][2]

a[1][0] a[1][1]

a[2][0]

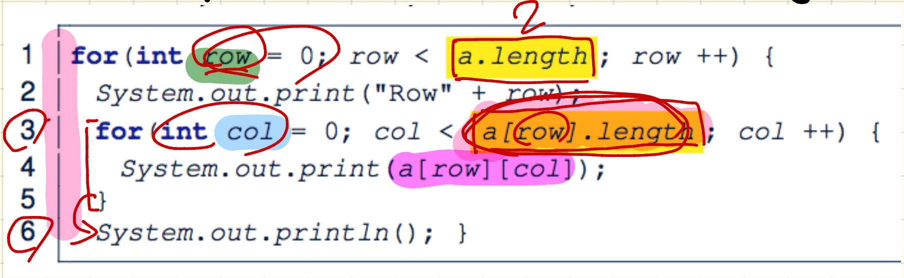
# of columns:

a[0].length 3

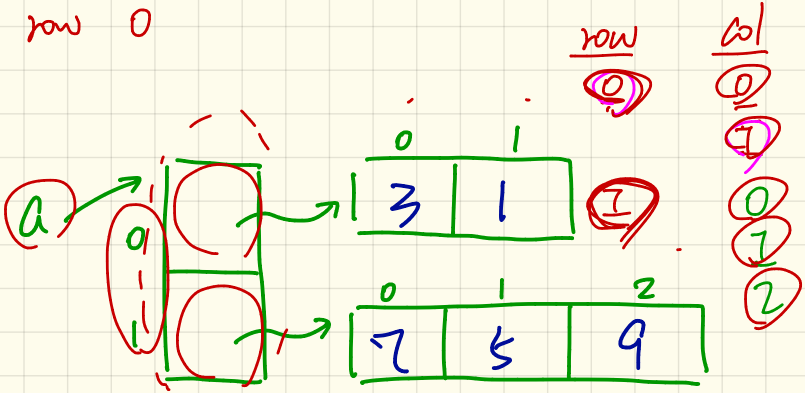
a[1].length 2

a[2].length 1

Example 1: Print 2D Array

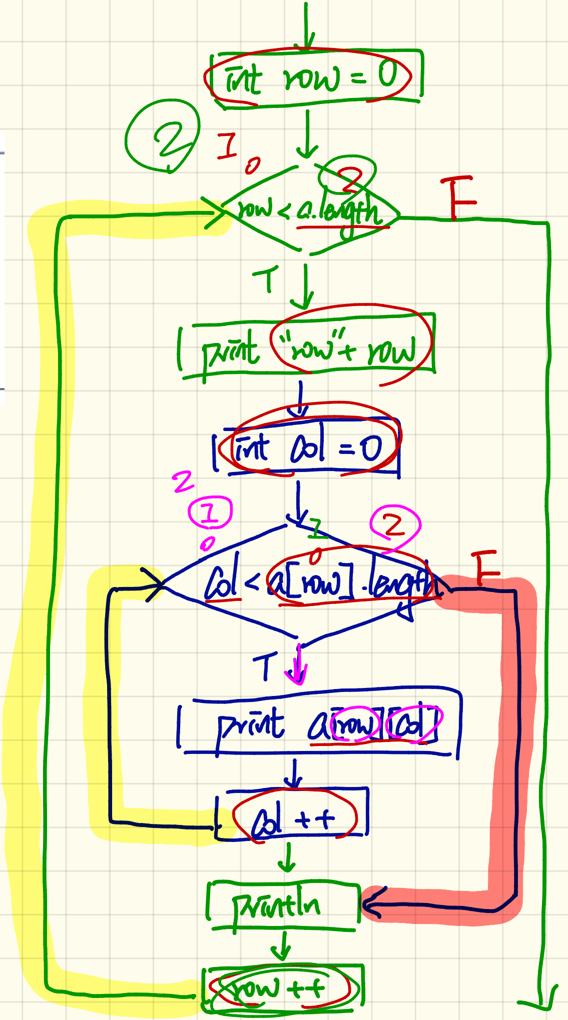


```
for (int row = 0; row < a.length; row ++) {
 System.out.print("Row" + row);
 for (int col = 0; col < a[row].length; col ++) {
   System.out.print(a[row][col]);
 }
 System.out.println(); }
```

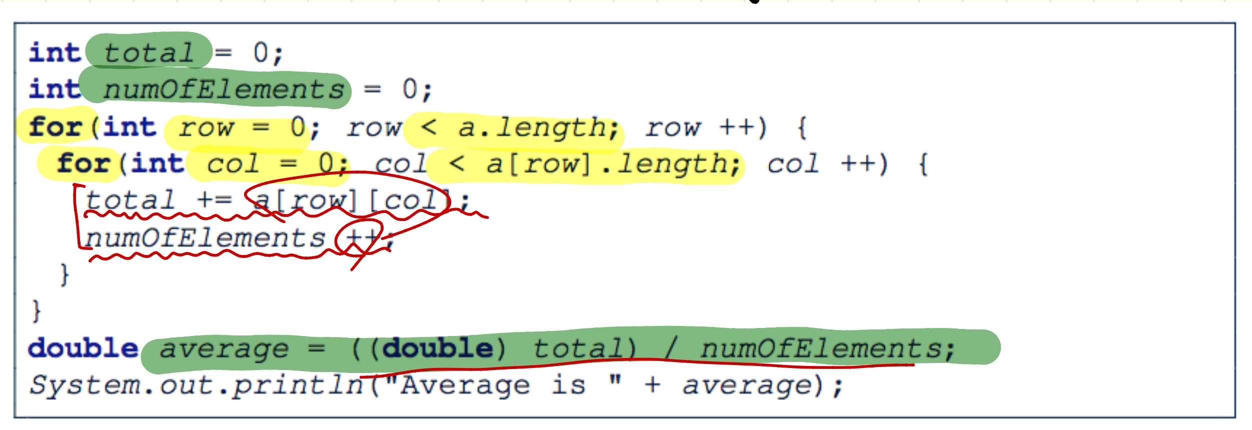


a[0].length 2

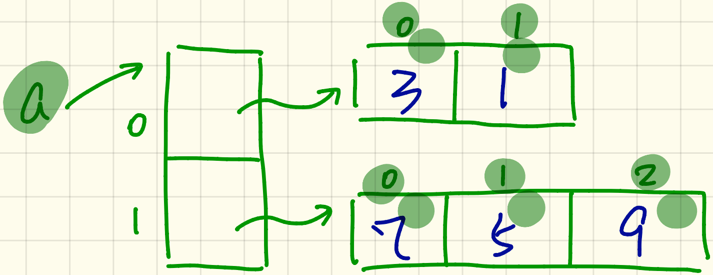
a[1].length 3



Example 2: Calculate Average



```
int total = 0;
int numOfElements = 0;
for(int row = 0; row < a.length; row ++) {
  for(int col = 0; col < a[row].length; col ++) {
    total += a[row][col];
    numOfElements ++;
  }
}
double average = ((double) total) / numOfElements;
System.out.println("Average is " + average);
```



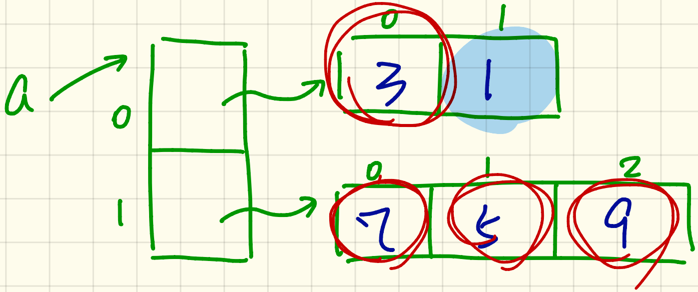
total += a[0][0]

total += a[0][1]

— += a[1][0]

— += a[1][1]

— += a[1][2]

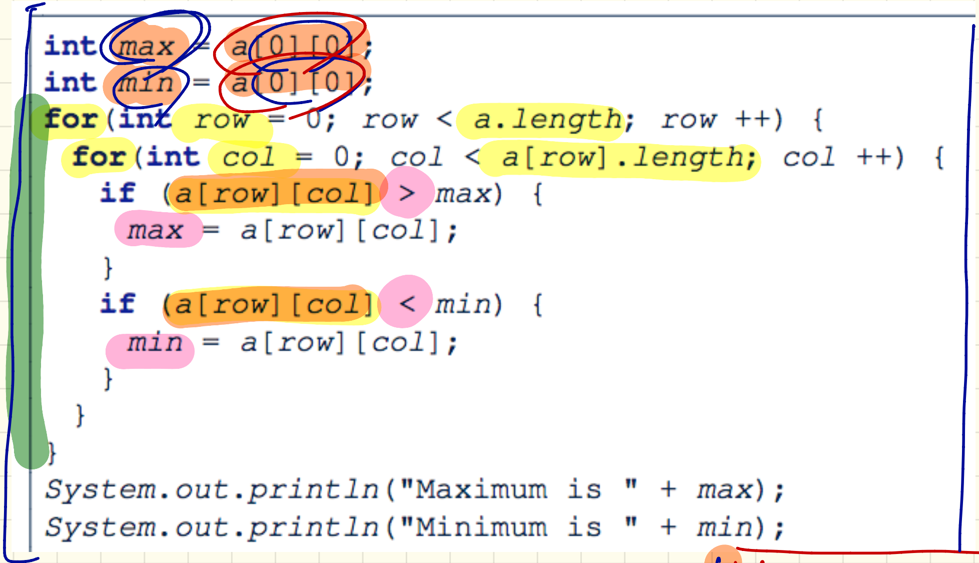


a[0][0] 3

max ~~3~~ ~~7~~ 9

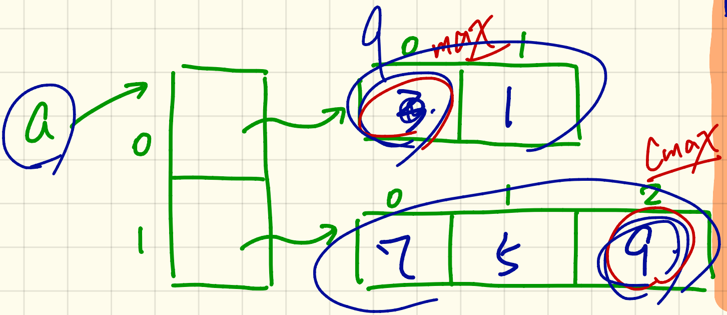
min ~~3~~ 1

Example 3: Calculat Max and Min



```
int max = a[0][0];
int min = a[0][0];
for(int row = 0; row < a.length; row ++) {
  for(int col = 0; col < a[row].length; col ++) {
    if (a[row][col] > max) {
      max = a[row][col];
    }
    if (a[row][col] < min) {
      min = a[row][col];
    }
  }
}
System.out.println("Maximum is " + max);
System.out.println("Minimum is " + min);
```

```
class Utilities {
  int maxOf(int[] a){
    ;
  }

  int maxOf(int[][] a){

  }
}
```



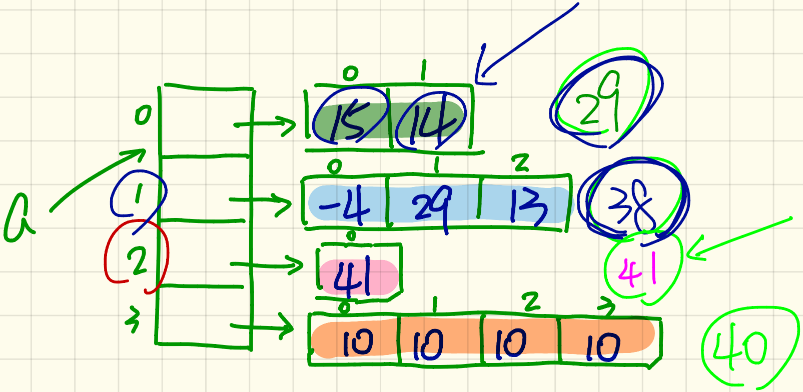
Assume: int min ~~max~~Of (int[] a)

```
int max = maxOf( a[0] );
for( int r = 1; r < a.length; r++){
  int cmax = maxOf( a[r] );
  if( cmax > max ) { max = cmax; }
}
```

3 9 9

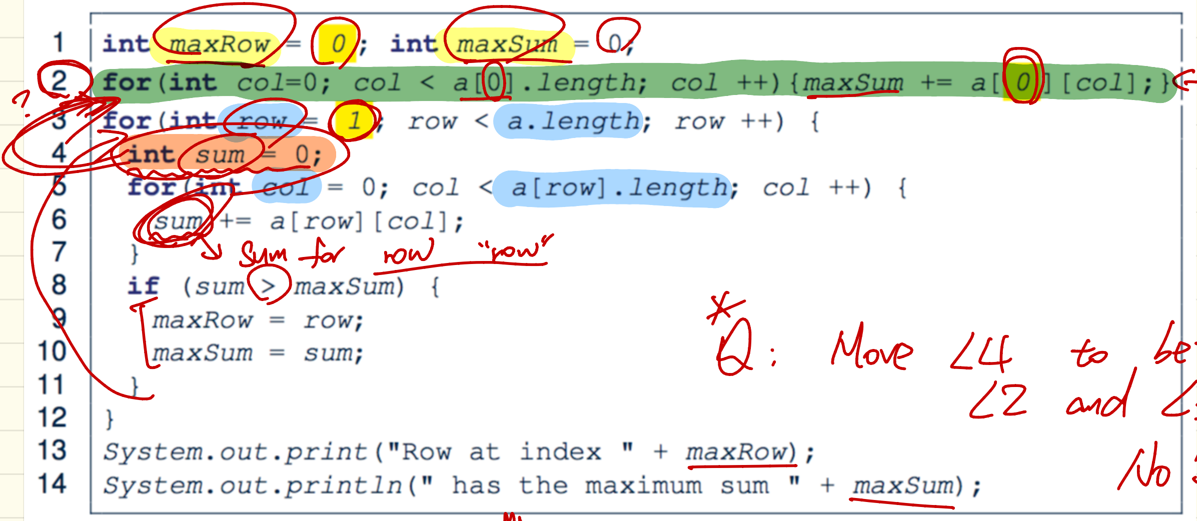
int maxSum = ?;

int maxRow = ?;



Row 2 has max sum 41

Example 4: Calculat Row with Max Sum



```
1   int maxRow = 0; int maxSum = 0;
2   for(int col=0; col < a[0].length; col ++){maxSum += a[0][col];}
3   for(int row = 1; row < a.length; row ++) {
4     int sum = 0;
5     for(int col = 0; col < a[row].length; col ++) {
6       sum += a[row][col];
7     }
8     if (sum > maxSum) {
9       maxRow = row;
10      maxSum = sum;
11    }
12  }
13  System.out.print("Row at index " + maxRow);
14  System.out.println(" has the maximum sum " + maxSum);
```

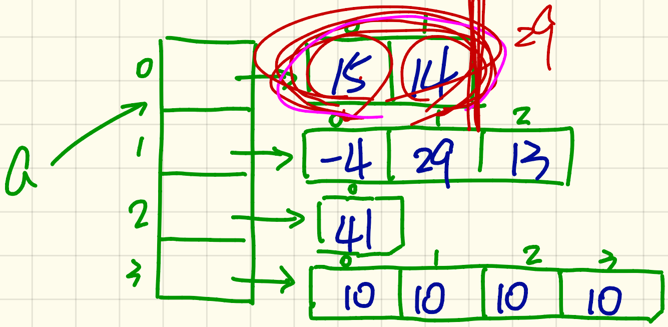
?

treating Row 0 to be containing the max sum.

sum for row "row"

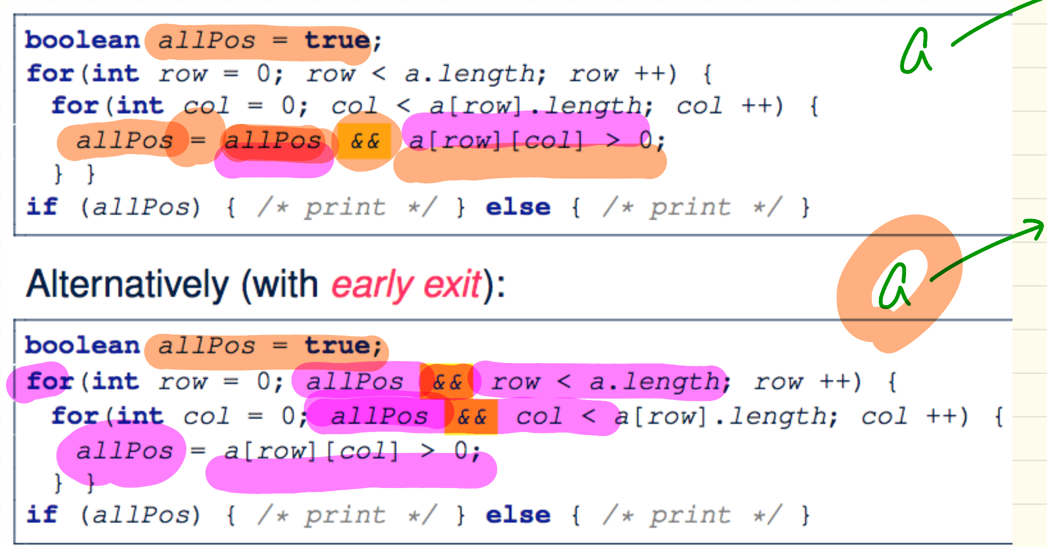
*Q: Move L4 to between L2 and L3?

No! sum will be summing up all elements of 2D array



	maxRow	maxSum
L1	0	0
L2	0	29

Example 5: all positive?

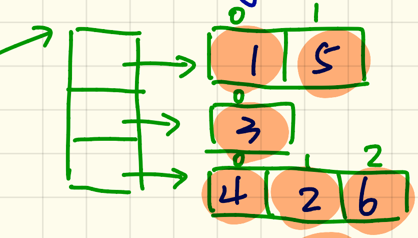


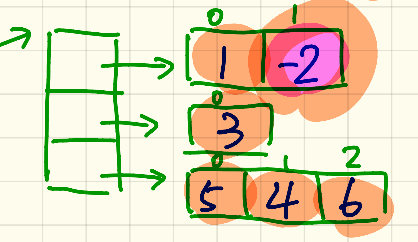
```
boolean allPos = true;
for(int row = 0; row < a.length; row ++) {
  for(int col = 0; col < a[row].length; col ++) {
    allPos = allPos && a[row][col] > 0;
  } }
if (allPos) { /* print */ } else { /* print */ }
```

Alternatively (with *early exit*):

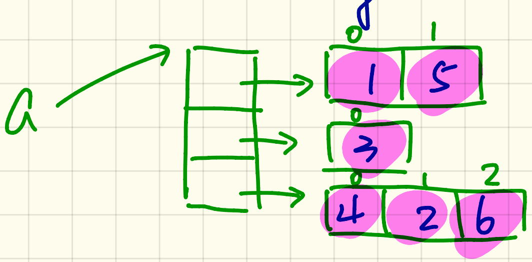
```
boolean allPos = true;
for(int row = 0; allPos && row < a.length; row ++) {
  for(int col = 0; allPos && col < a[row].length; col ++) {
    allPos = a[row][col] > 0;
  } }
if (allPos) { /* print */ } else { /* print */ }
```

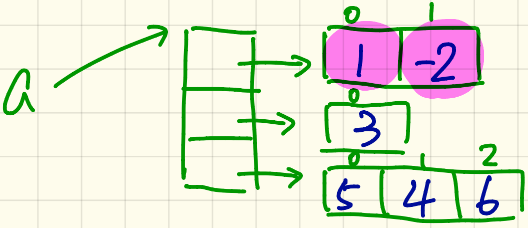
Version of No Early Exit





Version with Early Exit


a
0
1
1
5
0
3
0
1
2
4
2
6


a
0
1
1
-2
0
3
0
1
2
5
4
6