Two Dimensonal Array

2D array

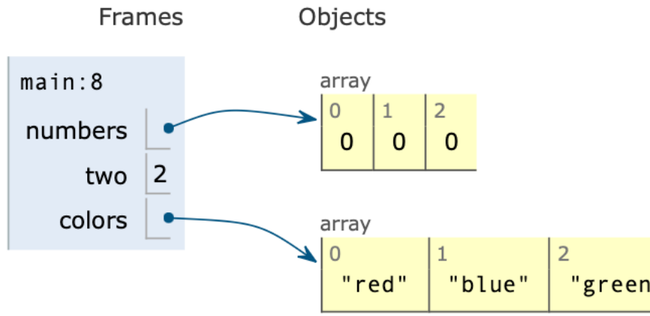
复习：一维数组的创建

- int[] a = new int[3]; // 创建默认值都是 0 的数组
- int[] b; b = new int[4]; // first declare b;then initialize;
- String[] colors = {“red”, “blue”, “green”}; // assign elments directly

- element of array could any time except void
  - primitive type
  - reference type
    - array is an reference type

Java 8
known limitations

```
1  public class YourClassNameHere {
2      public static void main(String[] args) {
3        int[] numbers = new int[3];
4
5        int two = 2;
6
7        String[] colors = {"red", "blue", "green"};
8      }
9  }
```



复习： index(indice) of array

- String[] colors = {“red”, “blue”, “green”}; // assign elments directly
- index start from zero(0)
- colors[2]
- colors[3]
- colors[5] // IndexOutOfBoundException

- int[] a;
- a[2]

```
int[] a;
System.out.println(a[2]);
```

Variable 'a' might not have been initialized
Initialize variable 'a' ⌥⇧⏎ More actions... ⌥⏎

```
int[] a
```

```
/Users/lxm/code/plugins/Hello2/src/ArrayAndForeach.java:23:28
java: variable a might not have been initialized
```

复习：length of array

- int[] arr = {1, 2, 3};
- length of array means how many elements in an array
- arr.length // 3

- String s = “helle”;
- s.length();

- ArrayList<String> list = new ArrayList<>();
- list.size();

复习：traverse array 遍历数组

- regular for-loop
  - loop variable is index of each elment

```java
int[] arr = {1, -1, 2};

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

- for-each loop
  - loop variable means element of an array

```java
int[] arr = {1, -1, 2};

for(int value : arr)
{
    System.out.println(value);
}
```

如果你不关心索引，只关心元素就用 for-each

```java
int i = 0;
System.out.println(arr[i]);
i += 1;
System.out.println(arr[i]);
i += 1;
System.out.println(arr[i]);
```

```java
int value = arr[0];
System.out.println(value);
value = arr[1];
System.out.println(value);
value = arr[2];
System.out.println(value);
```

复习：查找元素是否在数组中

- regular for-loop
  - loop variable is index of each elment

```java
int[] arr = {1, -1, 2};

int target = 3;
boolean found = false;

for (int i = 0; i < arr.length; i++)
{
    if (arr[i] == target)
    {
        found = true;
        break;
    }
}
if (found)
{
    System.out.println("found");
}
else
{
    System.out.println("not found");
}
```

```java
int[] arr = {1, -1, 2};

int target = 3;
boolean found = false;

for (int value : arr)
{
    if (value == target)
    {
        found = true;
        break;
    }
}
if (found)
{
    System.out.println("found");
}
else
{
    System.out.println("not found");
}
```

复习：查找元素是否在数组中(method)

- regular for-loop
  - loop variable is index of each elment

```java
int[] arr = {1, -1, 2};

int target = 3;
boolean found = false;

for (int i = 0; i < arr.length; i++)
{
    if (arr[i] == target)
    {
        found = true;
        break;
    }
}
if (found)
{
    System.out.println("found");
}
else
{
    System.out.println("not found");
}
```

```java
int[] arr = {1, -1, 2};

int target = 3;
boolean found = false;

for (int value : arr)
{
    if (value == target)
    {
        found = true;
        break;
    }
}
if (found)
{
    System.out.println("found");
}
else
{
    System.out.println("not found");
}
```

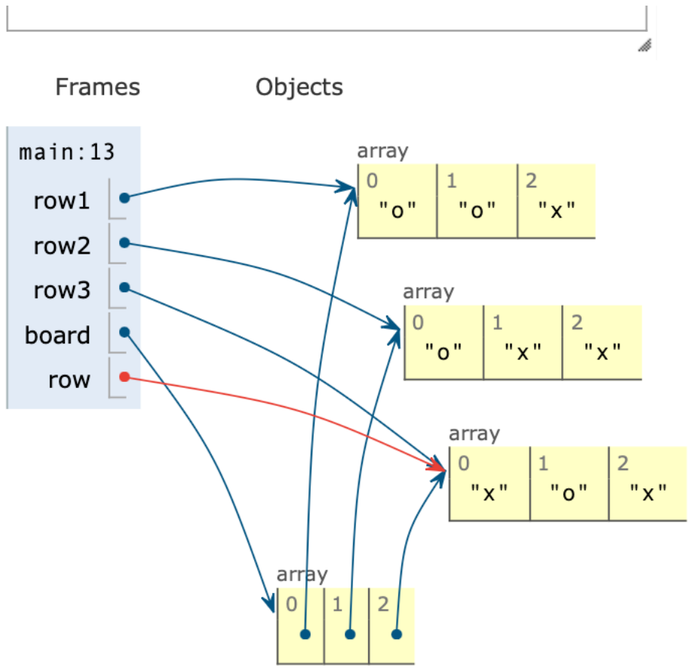
```java
public static boolean find(int[] arr, int target)
{
    for (int i = 0; i < arr.length; i++)
    {
        if (arr[i] == target)
        {
            return true;
        }
    }
    return false;
}
```

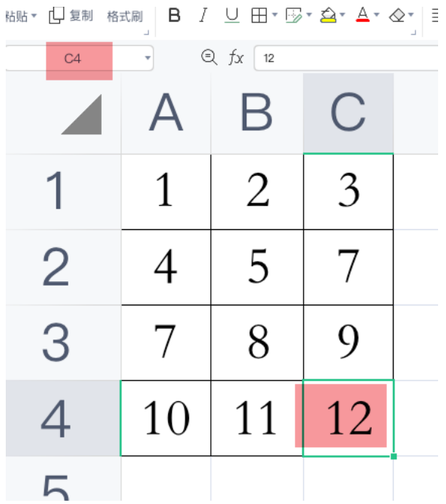
```java
public static boolean find(int[] arr, int target)
{
    for (int value : arr)
    {
        if (value == target)
        {
            return true;
        }
    }
    return false;
}
```

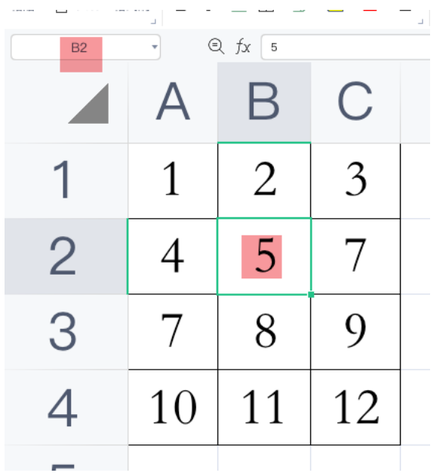
2d array

- element of array could be any time except void
- element of array could also be array

```
1  import java.util.Arrays;
2  public class YourClassNameHere {
3      public static void main(String[] args) {
4         String[] row1 = {"o", "o", "x"};
5          String[] row2 = {"o", "x", "x"};
6          String[] row3 = {"x", "o", "x"};
7
8          // element of array could be array
9          String[][] board = {row1, row2, row3};
10
11         String[] row = board[2];
12
13         System.out.println(Arrays.toString(row));
14
15     }
16 }
```







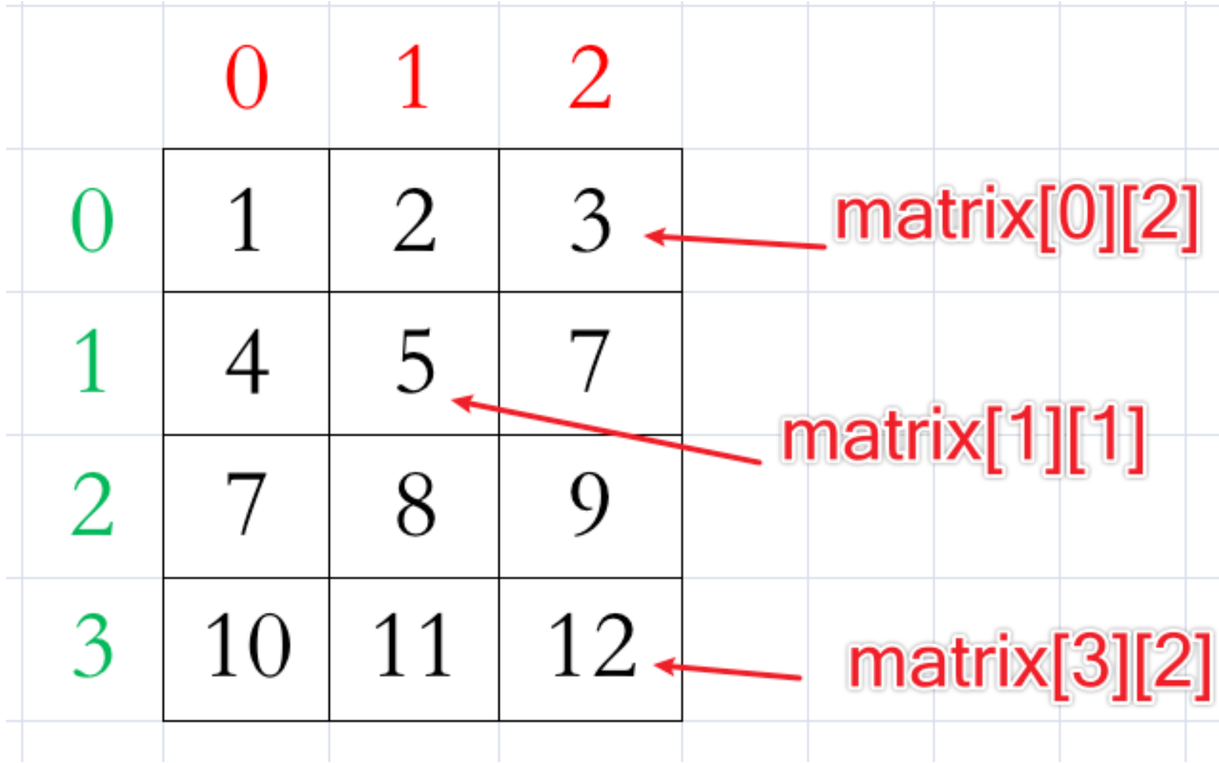
Create 2d array

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

int[][] arr2d;
arr2d = new int[2][6];

```
int[][] mat2 = new int[4][3];
mat2[0][0] = 1;
mat2[0][1] = 2;
mat2[0][2] = 3;

mat2[1][0] = 4;
mat2[1][1] = 5;
mat2[1][2] = 6;

mat2[2][0] = 7;
mat2[2][1] = 8;
mat2[2][2] = 9;

mat2[3][0] = 10;
mat2[3][1] = 11;
mat2[3][2] = 12;
```

访问二维数组的元素

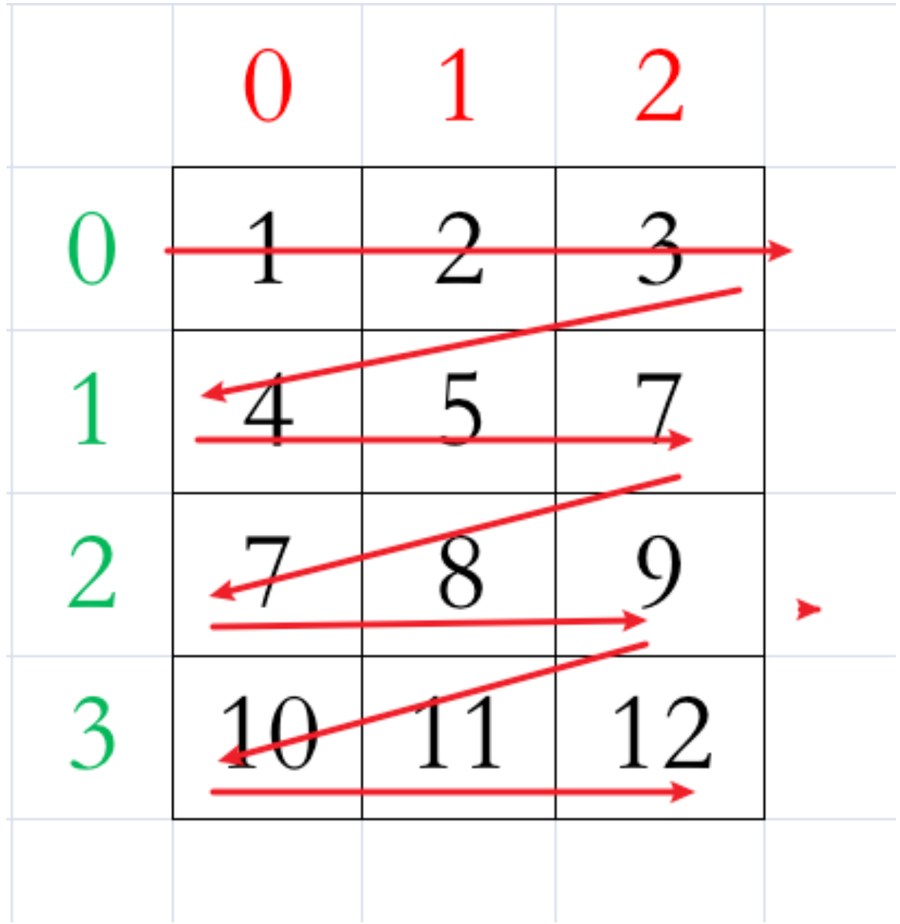



```
int[][] matrix =
 {
    {1, 2, 3},{4, 5, 6},
    {7, 8, 9}, {10, 11, 12}
 }
```

相当于在平面确定一个点的位置，需要横纵坐标两个数值。
访问二维数组中的元素，需要行索引和列索引。

二维数组的长度是有多少行（二维数组中有多少个数组）matrix.length
二维数组的列，任意一行有多少个元素。matrix[0].length

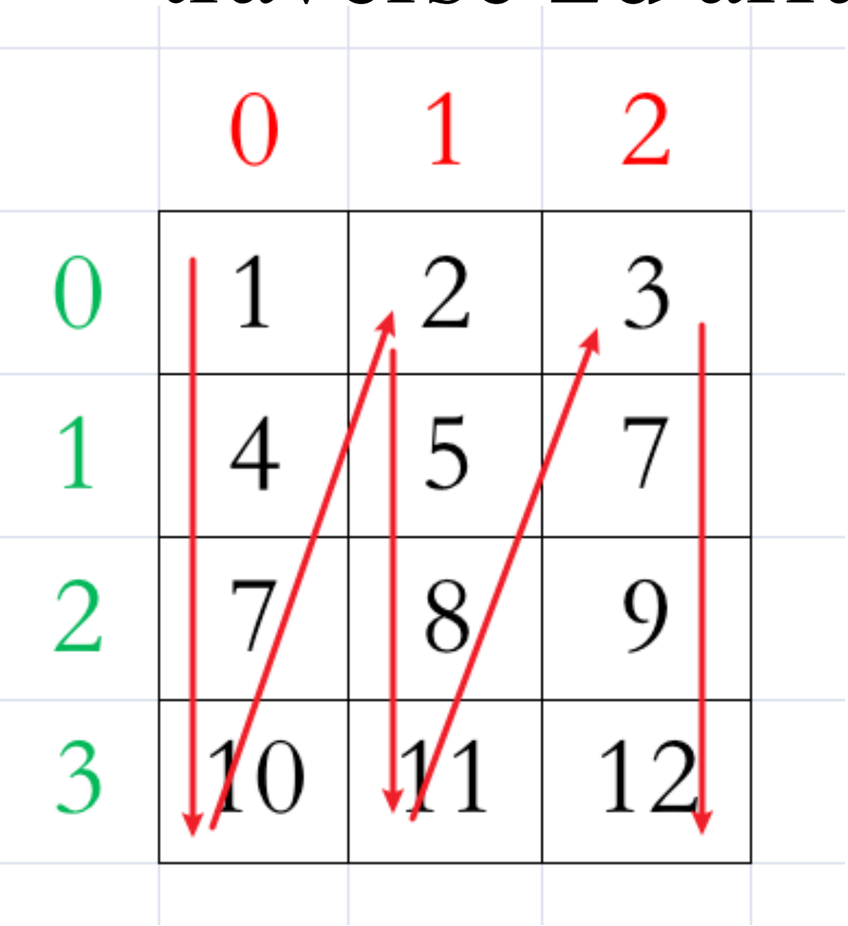
traverse 2d array in row-major




```java
// traverse 2d array using regular for-loop
for (int row = 0; row < mat.length; row++)
{
    for (int col = 0; col < mat[0].length; col++)
    {
        System.out.print(mat[row][col] + " ");
    }
    System.out.println();
}
```

```java
// traverse 2d array using for-each loop
for (int[] arr : mat)
{
    for (int value : arr)
    {
        System.out.print(value + " ");
    }
    System.out.println();
}
```

traverse 2d array in column maior

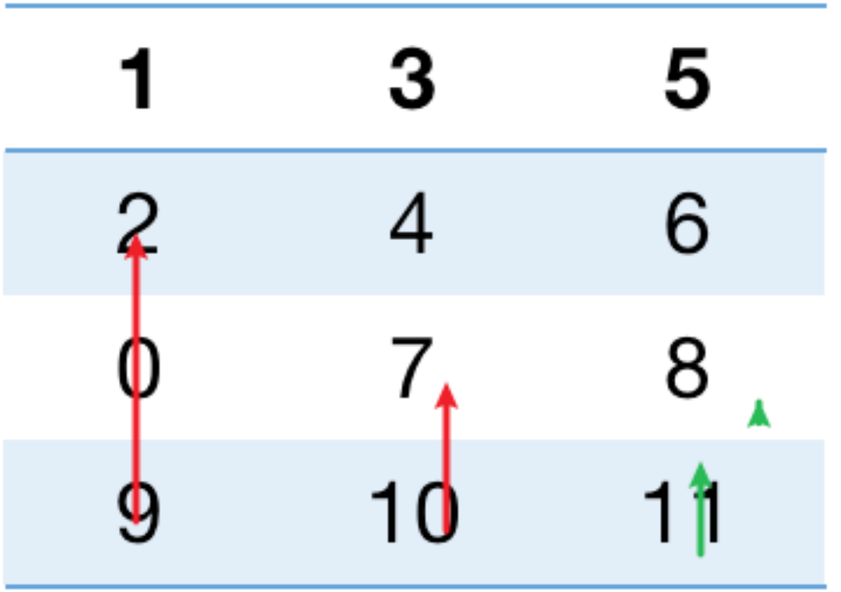



```java
int[][] mat = {
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9},
        {10, 11, 12}
};

// traverse 2d array in column-major
for (int col = 0; col < mat[0].length; col++)
{
    for (int row = 0; row < mat.length; row++)
    {
        System.out.print(mat[row][col] + " ");
    }
    System.out.println();
}
```

```
1 4 7 10
2 5 8 11
3 6 9 12
```

Consider the following code segment.

```
int[][] mat = {{1,3,5},
               {2,4,6},
               {0,7,8},
               {9,10,11}}

for (int col = 0; col < mat[0].length; col++)
    for (int row = mat.length - 1; row > col; row--)
        System.out.println(mat[row][col]);
```



When this code is executed, which will be the fifth element printed?

(A) 3

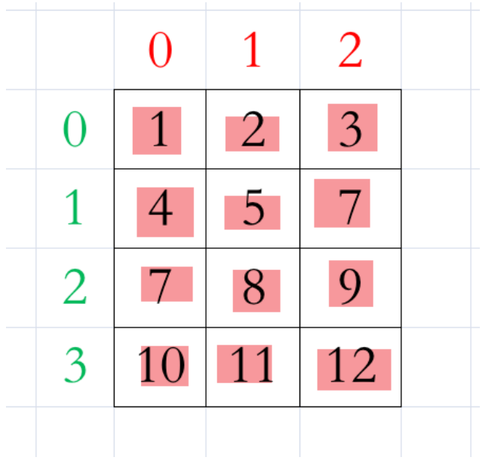
(B) 4

(C) 5

(D) 6

(E) 7

二维数组元素求和

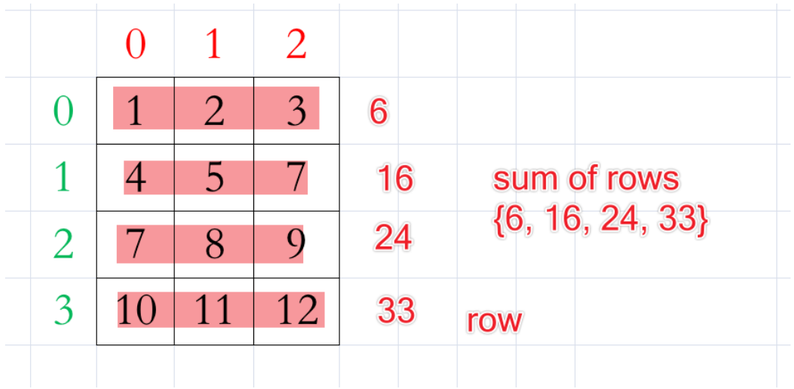


	0	1	2
0	1	2	3
1	4	5	7
2	7	8	9
3	10	11	12

```java
int sum = 0;
for (int row = 0; row < mat.length; row++)
{
    for (int col = 0; col < mat[0].length; col++)
    {
        sum += mat[row][col];
    }
}
System.out.println(sum);
```

```java
int sum = 0;
for (int[] row : mat)
{
    sum += rowSum(row);
}
System.out.println(sum);
```

sum of 2d array in row-major

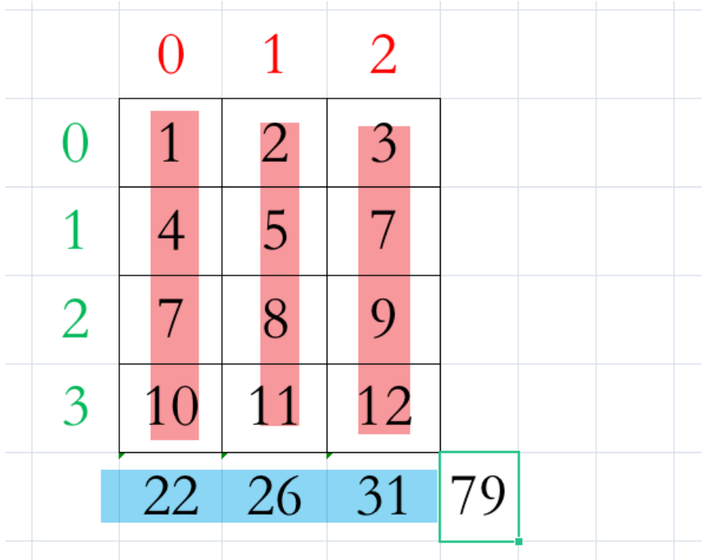


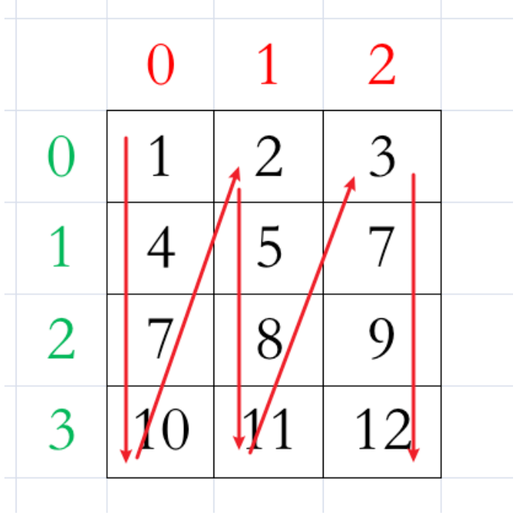

```
int sum = 0;
sum += rowSum(mat[0]);
sum += rowSum(mat[1]);
sum += rowSum(mat[2]);
sum += rowSum(mat[3]);
```

```java
public static int rowSum(int[] arr)
{
    int sum = 0;
    for (int value : arr)
    {
        sum += value;
    }
    return sum;
}
```

```java
    int sum = 0;
    for (int row = 0; row < mat.length; row++)
    {
//          for (int col = 0; col < mat[0].length; col++)
//          {
//              sum += mat[row][col];
//          }
        sum += rowSum(mat[row]);
    }
    System.out.println(sum);
```

Cal sum in column-major



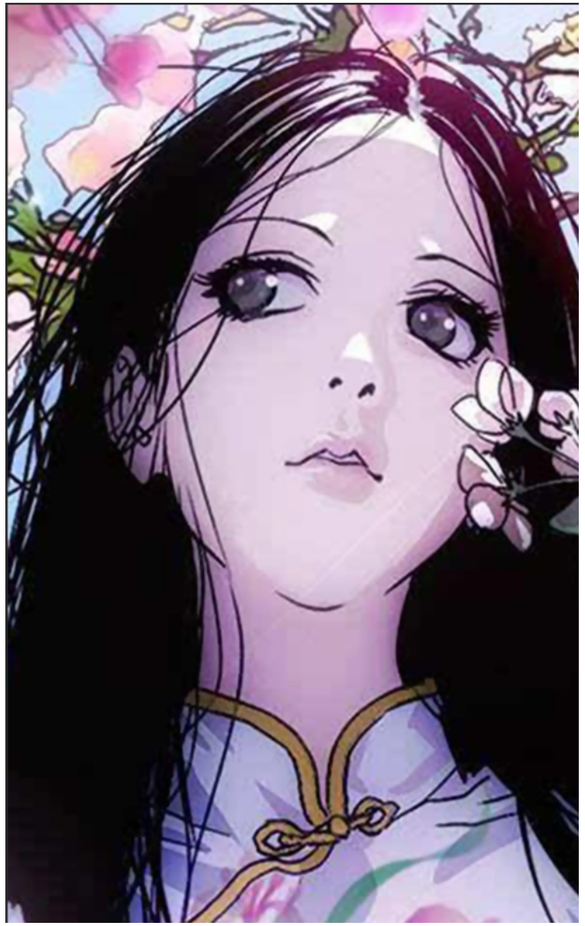



```java
public static int columnSum(int[][] arr2d, int column)
{



}
```

```java
public static int columnMajorSum(int[][] mat)
{
    int sum = 0;

    for (int col = 0; col < mat[0].length; col++)
    {
        sum += columnSum(mat, col);
    }
    return sum;
}
```

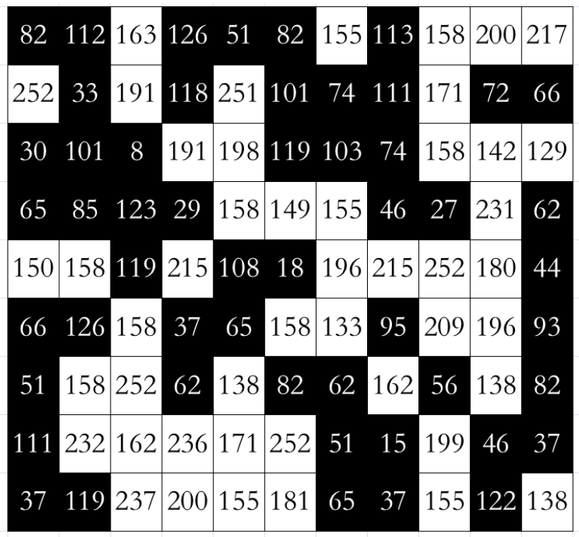
```java
int[][] mat = {
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9},
        {10, 11, 12}
};
columnMajorSum(mat);
```

图像的二值化Binary Image



Binary Image

82	112	163	126	51	82	155	113	158	200	217
252	33	191	118	251	101	74	111	171	72	66
30	101	8	191	198	119	103	74	158	142	129
65	85	123	29	158	149	155	46	27	231	62
150	158	119	215	108	18	196	215	252	180	44
66	126	158	37	65	158	133	95	209	196	93
51	158	252	62	138	82	62	162	56	138	82
111	232	162	236	171	252	51	15	199	46	37
37	119	237	200	155	181	65	37	155	122	138




```java
/**
 * 如果像素值大于 127，则将像素值设置为 255
 * 否则设置为 0
 * @param pixels
 */
no usages
public static void binaryImage(int[][] pixels)
{
    for (int row = 0; row < pixels.length; row++)
    {

    }
}
no usages
```

Flip Image

Flip Image

- reverse array
  - [1, 2, 3, 5, 6]
  - [6, 5, 3, 2, 1]

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

// create a new array having same length
int[] reversed = new int[arr.length];
// put each element of arr to new array
// using for loop
// visit element of arr for last to first
for (int index = 0; index < arr.length; index++)
{
    reversed[0] = arr[arr.length - 1 - 0];
}
```

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

for (int index = 0; index < arr.length / 2; index++)
{
    int tmp = arr[index];
    arr[index] = arr[arr.length - 1 - index];
    arr[arr.length - 1 - index] = tmp;
}
```

Flip Image

```
public static void flipRow(int[] rowPixels)
{
    // reverse array rowPixels
    for (int index = 0; index < rowPixels.length / 2; index++)
    {
        int tmp = rowPixels[index];
        rowPixels[index] = rowPixels[rowPixels.length - 1 - index];
        rowPixels[rowPixels.length - 1 - index] = tmp;
    }
}
}
```

no usages

```
public static void flipImage(int[][] pixels)
{
    // reverse each row
    for (int[] row : pixels)
    {
        flipRow(row);
    }
}
```