Lecture 30

- Covers
  - Multi-dimensional arrays

- Reading: Savitch 6.5

Example

- We may want to store in an array the total monthly rainfall for one year
- Using an array, we could store each month in each successive element

0	1	2	3	4	5	6	7	8	9	10	11
12	0	10	20	35	47	40	58	93	68	47	29

```
int[ ] months = new int[12];
```

Example

- What if we want to store the average monthly rainfall for 10 years?
- We do not want to use 10 arrays of 12 months
- Instead, we create an array of arrays

Example

months

years

	0	1	2	3	4	5	6	7	8	9	10	11
0	12	0	10	20	35	47	40	58	93	68	47	29
1	5	8	16	31	45	49	50	52	68	79	42	18
2	20	10	9	19	28	33	48	61	76	84	32	13
3	6	12	14	23	36	39	40	55	82	65	28	9
4	15	15	12	29	31	49	55	60	89	71	44	31
5	22	35	42	35	52	56	99	89	92	76	53	33
6	18	12	22	27	34	41	56	78	87	62	44	27
7	13	0	5	19	33	38	41	66	78	64	39	18
8	0	2	0	12	28	19	22	35	45	55	42	23
9	14	12	3	4	19	22	31	42	51	53	37	21

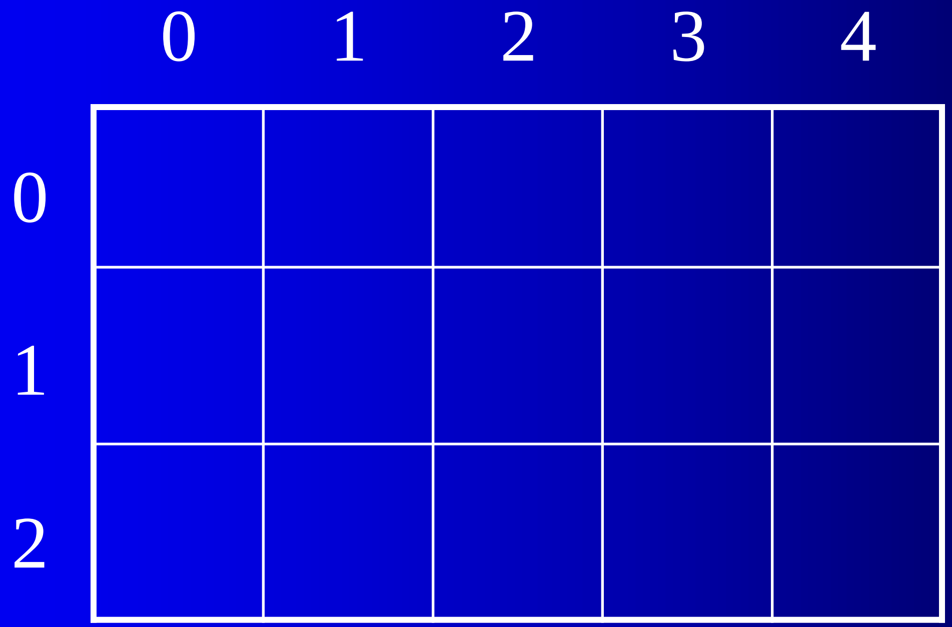
Multi-dimensional arrays

- Arrays with more than one index
- Arrays with 2 indexes are referred to as two-dimensional arrays
- A two-dimensional array is an array containing arrays
- A three-dimensional array is an array containing two-dimensional arrays

Two-dimensional arrays

- Can be thought of as a two-dimensional table
- Declaration

```
int[ ][ ] matrix = new int[3][5];
```



Two-dimensional arrays

- Accessing individual elements

```
matrix[2][1] = 12;
int val = matrix[2][1];
```

	0	1	2	3	4
0					
1					
2		12			

Two-dimensional arrays

- Initialising arrays

```
int[ ][ ] matrix = { {1, 2, 3, 4, 5} ,
                     {6, 7, 8, 9, 10},
                     {11, 12, 13, 14, 15} };
```

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15

Two-dimensional arrays

- Iterating through the array

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

Example

- Creating an array of 10 years of total monthly rainfall

```
int[ ][ ] rainfall = new int[10][12];
```

- Or

```
int[ ][ ] rainfall = { {12,   0, 10, 20, 35, 47, 40, 58, 93, 68, 47, 29},
                       { 5,   8, 16, 31, 45, 49, 50, 52, 68, 79, 42, 18},
                       {20, 10,   9, 19, 28, 33, 48, 61, 76, 84, 32, 13},
                       { 6, 12, 14, 23, 36, 39, 40, 55, 82, 65, 28,  9 },
                       {15, 15, 12, 29, 31, 49, 55, 60, 89, 71, 44, 31},
                       {22, 35, 42, 35, 52, 56, 99, 89, 92, 76, 53, 33},
                       {18, 12, 22, 27, 34, 41, 56, 78, 87, 62, 44, 27},
                       {13,   0,   5, 19, 33, 38, 41, 66, 78, 64, 39, 18},
                       { 0,   2,   0, 12, 28, 19, 22, 35, 45, 55, 42, 23},
                       {14, 12,   3,   4, 19, 22, 31, 42, 51, 53, 37, 21} };
```

Example

months

years	0	1	2	3	4	5	6	7	8	9	10	11
0	12	0	10	20	35	47	40	58	93	68	47	29
1	5	8	16	31	45	49	50	52	68	79	42	18
2	20	10	9	19	28	33	48	61	76	84	32	13
3	6	12	14	23	36	39	40	55	82	65	28	9
4	15	15	12	29	31	49	55	60	89	71	44	31
5	22	35	42	35	52	56	99	89	92	76	53	33
6	18	12	22	27	34	41	56	78	87	62	44	27
7	13	0	5	19	33	38	41	66	78	64	39	18
8	0	2	0	12	28	19	22	35	45	55	42	23
9	14	12	3	4	19	22	31	42	51	53	37	21

Example

- How much rain fell in the first year?

rainfall[0][0] + rainfall[0][1] + rainfall[0][2] + rainfall[0][3] + rainfall[0][4] + rainfall[0][5] + rainfall[0][6] + rainfall[0][7] + rainfall[0][8] + rainfall[0][9] + rainfall[0][10] + rainfall[0][11]

Example

- How much rain fell in the first year?

```
int yearOne = 0;
for (int j = 0; j < rainfall[ 0 ].length; ++j)
{
    yearOne += rainfall[ 0 ][ j ];
}
System.out.println("Rainfall - first year: " + yearOne);
```

Example

- Output how much rain fell in each of the ten years?

```
int currentYear;
for (int i = 0; i < rainfall.length; ++i)
{
   currentYear = 0;
   for (int j = 0; j < rainfall[i].length; ++j)
   {
      currentYear += rainfall[ i ][ j ];
   }
   System.out.println("Year " + (i+1) + ": " + currentYear);
}
```

Example

- What is the average June rainfall?

```
(rainfall[0][5]  + rainfall[1][5]  + rainfall[2][5]  +
 rainfall[3][5]  + rainfall[4][5]  + rainfall[5][5]  +
 rainfall[6][5]  + rainfall[7][5]  + rainfall[8][5]  +
 rainfall[9][5])  / 10.0
```

Example

- What is the average June rainfall?

```
 int juneRainfall = 0;
for (int i = 0; i < rainfall.length; i++)
{
   juneRainfall += rainfall[i][5];
}
System.out.println("Average June Rainfall: " +
                    juneRainfall / (double)rainfall.length);
```

Example

- Display the average rainfall for each month over the 10 years

```
int currentMonthRainfall;
for (int j = 0; j < rainfall[0].length; ++j)
{
    currentMonthRainfall = 0;
    for (int i = 0; i < rainfall.length; i++)
    {
        currentMonthRainfall += rainfall[i][j];
    }
    System.out.println("Average rainfall for month " + (j+1)
       + ": " + currentMonthRainfall / (double)rainfall.length);
}
```

Example

- Which month of which year had the highest rainfall? How much rain fell in that month?
- Algorithm

```
LOOP FOR each year
    LOOP FOR each month
        IF the rainfall in that month of that year is greater
        than the current max THEN
            Update the current max
        ENDIF
    ENDLOOP
ENDLOOP
```

Example

```
int currentMax = rainfall[0][0];
int yearOfCurrentMax = 0;
int monthOfCurrentMax = 0;
for (int i = 0; i < rainfall.length; ++i)
{
   for (int j = 0; j < rainfall[i].length; ++j)
   {
      if (rainfall[i][j] > currentMax)
      {
         currentMax = rainfall[ i ][ j ];
         yearOfCurrentMax = i;
         monthOfCurrentMax = j;
      }
   }
}
System.out.println("The most rain fell in month " +
                                (monthOfCurrentMax + 1) + " of year " +
                                (yearOfCurrentMax + 1) + "\n" + currentMax +
                                "mm of rain fell in this month");
```

Example

- On average which is the driest month?
- Algorithm

```
LOOP FOR each month
    Calculate the average rainfall
    IF average < driest month so far THEN
        Update driest month
    ENDIF
ENDLOOP
Output driest month
```

* *To what do we initialise the minimum?*

Example

```
int currentMonthlyMin = 0;
int monthOfCurrentMin = 0;
int currentMonthRain;
int currentAverage;
for (int j = 0; j < rainfall[0].length; ++j)
{
    currentMonthRain = 0;
    for (int i = 0; i < rainfall.length; i++)
    {
        currentMonthRain += rainfall[i][j];
    }
    currentAverage = currentMonthRain / (double)rainfall.length;
    if (j == 0 || currentAverage < currentMonthlyMin)
    {
        monthOfCurrentMin = j;
        currentMonthlyMin = currentAverage;
    }
}
System.out.println("On average, the driest month is " +
    (monthOfCurrentMin + 1) + " with an average rainfall of " +
    currentMonthlyMin + "mm");
```

Declaring and initialising two-dimensional arrays

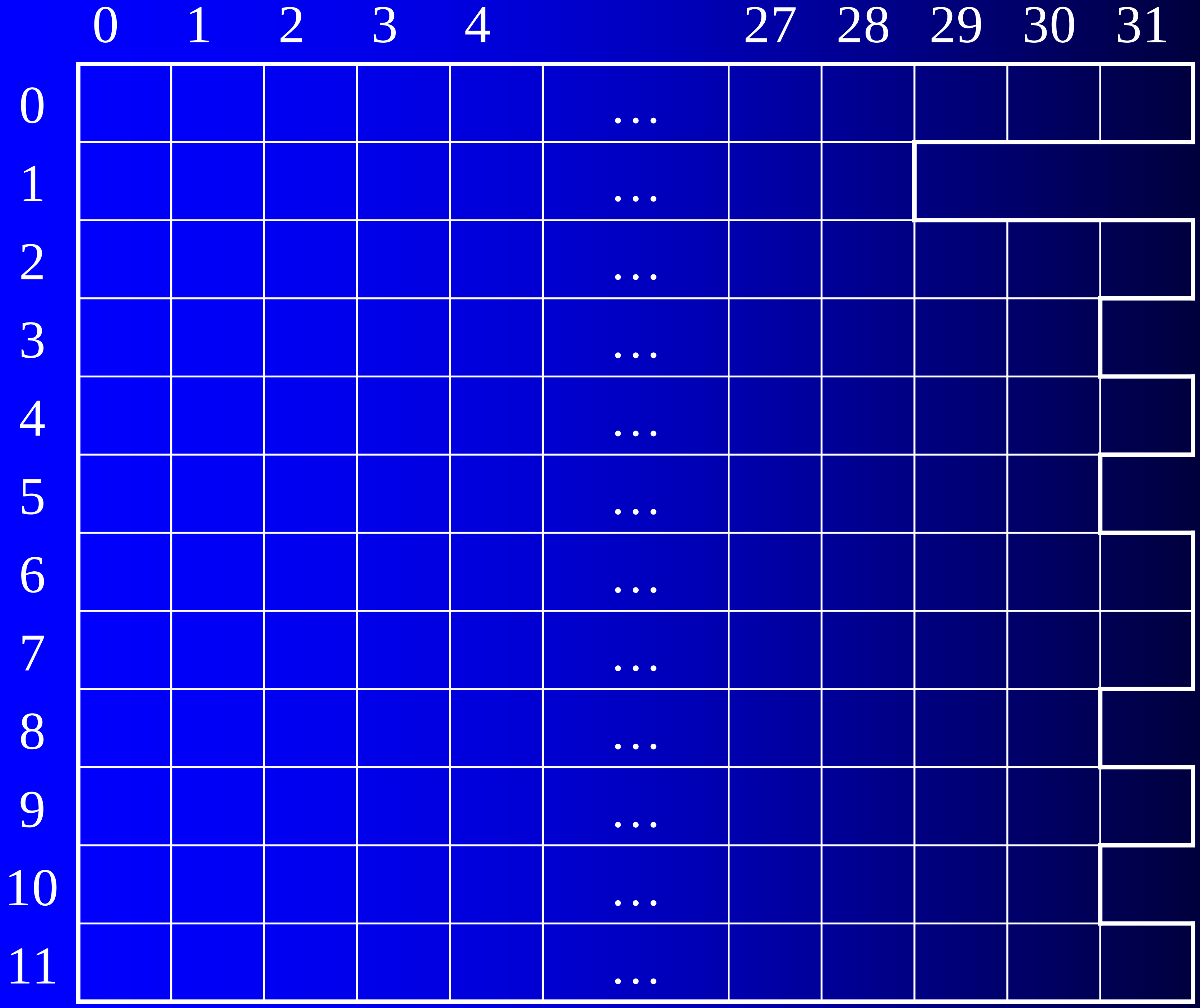
```
int [ ] [ ] a;
int [ ] [ ] a = new int [5][ ];
int [ ] [ ] a = new int [5][10];
int [ ] [ ] a = { {1,2,3}, {4,5,6} };
int [ ] [ ] a = { {1,2,3}, {4,5} };
```

Ragged arrays

- Sometimes we wish to create an array of arrays where each array element may be of a different size
- This is possible in Java because the array of arrays contains references to separate array objects, and the length of each array object is defined in that object

Example

- For example, we may want to store the daily maximum temperatures for a year
- If we want to index them by day and month, the length of a month is not the same for each


0
1
2
3
4
27
28
29
30
31
0
1
2
3
4
5
6
7
8
9
10
11
. . .

Ragged arrays

- Declaration

```
int[ ][ ] temperature  = new int[12][ ];
temperature[0]  = new int[31];
temperature[1]  = new int[28];
temperature[2]  = new int[31];
…
```

Multi-dimensional arrays

- So far we have looked at two multi-dimensional arrays that have two indexes
- We can create multi-dimensional arrays with more than two indexes

```
String[ ][ ][ ] s = new String[3][5][12];
double[ ][ ][ ][ ] d = new Double[5][10][12][31];
```

Class exercise

- Problem
  - What is the result of the following code?

```
int[ ][ ] myArray = new int[4][4];
int index1, index2;

for (index1 = 0; index1 < 4; index1++)
    for (index2 = 0; index2 < 4; index2++)
        myArray[index1][index2] = index2;

for (index1 = 0; index1 < 4; index1++)
{
    for (index2 = 0; index2 < 4; index2++)
        System.out.print(myArray[index1][index2] + " ");
    System.out.println( );
}
```

Class exercise

- Problem
  - What is the result of the following code?

```
char[ ][ ] myArray = new char[3][5];
int index, index1, index2;
for (index = 0; index < myArray.length * myArray[0].length; index++)
{
     myArray[index/myArray[0].length][index%myArray[0].length] =
           (char)('A' + index);
}
for (index1 = 0; index1 < 3; index1++)
{
     for (index2 = 0; index2 < 5; index2++)
          System.out.print(myArray[index1][index2] + " ");
     System.out.println( );
}
```

Example: Drunkard's Walk

- A man leaves the pub to walk home
- Every step he takes on the way home is in one of the four directions: north, south, east or west
- The direction in which he takes each step is random
- Given the location of the pub and his home, calculate how many steps he takes to get home
- Keep track of which coordinates he has visited on the way home

Example: Drunkard’s Walk

```
*****************
*...............*
*...............*
*..P............*
*...............*
*...............*
*...............*
*...............*
*...............*
*...............*
*............H..*
*****************
```

Town with pub and home

```
*****************
*...............*
*...............*
*.._............*
*..@............*
*...............*
*...............*
*...............*
*...............*
*...............*
*............H..*
*****************
```

Leaving the pub

Example: Drunkard's Walk

```
*****************
*...............*
*...............*
*..-............*
*.@-............*
*...............*
*...............*
*...............*
*...............*
*...............*
*............H..*
*****************
```

Going east

```
*****************
*...............*
*...............*
*..-............*
*@--............*
*...............*
*...............*
*...............*
*...............*
*...............*
*............H..*
*****************
```

At city wall

Example: Drunkard's Walk

```
*****************
*------------...*
*.------------..*
*-------------..*
*-------------..*
*--------------.*
*--------------.*
*-------------..*
*-------------..*
*------------@..*
*...---------H..*
*****************
```

Getting close

```
*****************
*------------...*
*.------------..*
*-------------..*
*-------------..*
*--------------.*
*--------------.*
*-------------..*
*-------------..*
*-------------..*
*...---------@..*
*****************
```

```
Took 655 steps!
```

Drunkard’s Walk solution

```
public class DrunkardsWalk
{
    char[ ][ ] town;
    int currentXcoord;
    int currentYcoord;
    int numberOfSteps;


}
```

* Defining the class and attributes

Drunkard's Walk solution

```
DrunkardsWalk( )
{
   int numberOfSteps = 0;
   System.out.print("Enter size of town [height] [width]: ");
   int height = keyboard.nextInt( );
   int width = keyboard.nextInt( );

   town = new char[height][width];
   for (int i = 0; i < height; ++i)
      for (int j = 0; j < width; ++j)
          town[i][j] = '.';

   System.out.print("Enter coordintates of Pub [Y] [X]: ");
   int pubYcoord = keyboard.nextInt( );
   int pubXcoord = keyboard.nextInt( );

   town[pubYcoord][pubXcoord] = 'P';

   System.out.print("Enter coordintates of Home [Y] [X]: ");
   int homeYcoord = keyboard.nextInt( );
   int homeXcoord = keyboard.nextInt( );
   town[homeYcoord][homeXcoord] = 'H';

   currentXcoord = pubXcoord;
   currentYcoord = pubYcoord;
}
```

* Defining the constructor

Drunkard's Walk solution

```
public void displayTown( )
{
    for (int i = 0; i < town[0].length+2; ++i)
    {
        System.out.print("*");
    }
    System.out.println( );

    for (int x = 0; x < town.length; ++x)
    {
        System.out.print("*");
        for (int y = 0; y < town[x].length; ++y)
        {
            System.out.print(town[x][y]);
        }
        System.out.println("*");
    }

    for (int i = 0; i < town[0].length+2; ++i)
    {
        System.out.print("*");
    }
    System.out.println( );
}
```

* Display the town map

Drunkard's Walk solution

```
public void walkHome( )
{
   int nextXcoord = currentXcoord;
   int nextYcoord = currentYcoord;
   boolean bumpedIntoWall;

   while(town[nextYcoord][nextXcoord] != 'H')
   {
      ++numberOfSteps;
      bumpedIntoWall = false;
      int direction = (int) (Math.random() * 4);
      switch(direction)
      {
          case 0: // go north
              if (currentYcoord == 0)
             {
                 bumpedIntoWall = true;
             }
             else
             {
                 nextXcoord = currentXcoord;
                 nextYcoord = currentYcoord -1;
             }
             break;
```

* Walk home

Drunkard's Walk solution

```
case 1: // go south
    if (currentYcoord == town.length - 1)
    {
        bumpedIntoWall = true;
    }
    else
    {
        nextXcoord = currentXcoord;
        nextYcoord = currentYcoord +1;
    }
    break;
 case 2: // go west
    if (currentXcoord == 0)
    {
        bumpedIntoWall = true;
    }
    else
    {
        nextXcoord = currentXcoord -1;
        nextYcoord = currentYcoord;
    }
    break;
```

* Walk home

Drunkard's Walk solution

```
            case 3: // go east
                if (currentXcoord == town[0].length -1)
                {
                    bumpedIntoWall = true;
                }
                else
                {
                    nextXcoord = currentXcoord +1;
                    nextYcoord = currentYcoord;
                }
                break;
        }
        // update map if not home
        if (town[nextYcoord][nextXcoord] != 'H')
        {
            town[currentYcoord][currentXcoord] = '-'; // visited
            currentXcoord = nextXcoord;
            currentYcoord = nextYcoord;
            town[currentYcoord][currentXcoord] = '@';
            displayTown( );
        }
        keyboard.nextLine( );
    }
    System.out.println("Reached Home! it took " + numberOfSteps +
                            " steps");
}
```

* Walk home

Next lecture

- Inheritance