2D Arrays in Java

- In this lecture, we will cover
  - Multidimensional Arrays
    - An array of arrays
  - Keeping track of access

Your future in CS

I used to include this on my slides, but since these slides have changed - going to just leave it up here for every notebook. I get a lot of questions about more programming courses, the concentrations, and minors in computer science. Here is a brief reminder.

CS 165 – Next Course In Sequence, also consider CS 220 (math and stats especially)

- CO Jobs Report 2021 – 77% of *all* new jobs in Colorado require programming
- 60% of all STEM jobs requires *advanced* (200-300 level)
- 31% of all Bachelor of Arts degree titled jobs also required coding skills
- 2016 Report found on average jobs that require coding skills paid $22,000 more

- Concentrations in CS:

  - Computer science has a number of concentrations.
    - General concentration is the most flexible, and even allows students to double major or minor pretty easily.
    - Software Engineering
    - Computing Systems
    - Human Centered Computing
    - Networks and Security
    - Artificial Intelligence
    - Computer Science Education.
  - Minors:
    - Minor in Computer Science - choose your own adventure minor
    - Minor in Machine Learning - popular with stats/math, and engineering
    - Minor in Bioinformatics - Biology + Computer Science

Warmup Activity

- write a method that build an array of length N
- Adds random **int**s to the array until N
  - from 1 until and including *range* (another parameter passed in)
- returns the array

Reminder:

```
Random rnd = new Random();
int val = rnd.nextInt(6)+1;
```

generates a random number between 0-5, adds 1 to it, and stores it into val

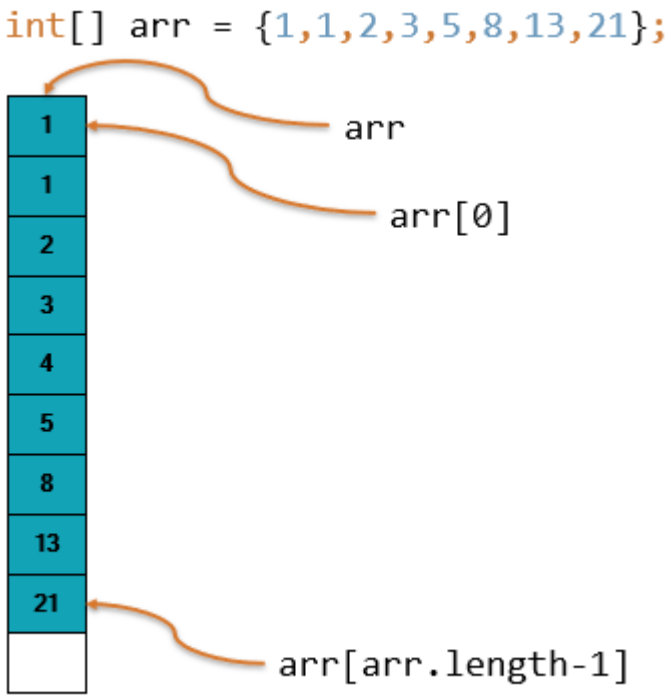
In [2]:
```
public static int[] randomArray(int n, int range) {
    Random rnd = new Random();
    int[] rtn = new int[n];
    for(int i = 0; i < n; i++) {
        rtn[i] = rnd.nextInt(range) + 1;
    }
    return rtn;
}

int[] random_one = randomArray(10, 12);
System.out.println(Arrays.toString(random_one));

int[] random_two = randomArray(10, 20);
System.out.println(Arrays.toString(random_two))
```

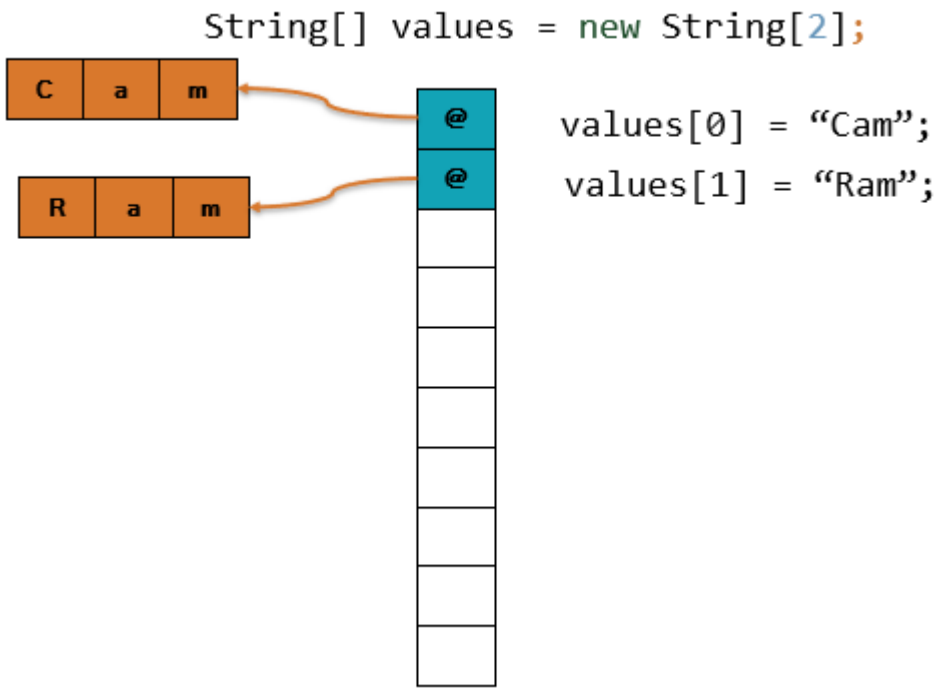
```
[3, 2, 10, 7, 10, 5, 6, 11, 10, 3]
[13, 15, 1, 17, 20, 20, 4, 1, 12, 16]
```

Array Review

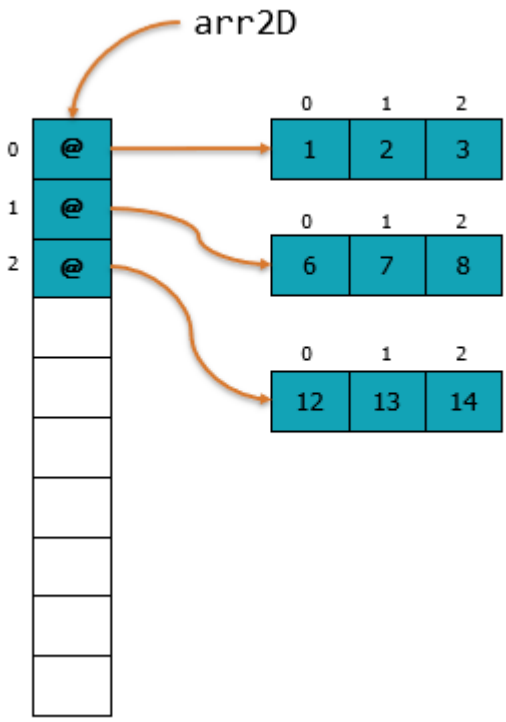


- Ways to store
  - Variables in order
  - index from 0..N
- Arrays are

    - a type themselves
    - the value of the array
        - reference to memory location!
        - Matters for parameters and return values!
    - .length gives us total memory allocated
- Arrays can
    - be any size – as long as you allocate it
    - Store any valid type
    - primitives and objects
    - and other arrays (since they are a type themselves)!



- An array of objects are always references to those objects
- So since arrays are types, we can also do the following



In [3]:
```
int[][] arr2D = {{1,2,3},{6,7,8},{12,13,14}};

int[] inner = arr2D[0];
System.out.println(inner[0]); // what is printed here?

// we also have a short hand notation that is *very* common

int val = arr2D[2][1];
System.out.println(val); // what is printed here?
```

```
1
13
```

The above array is known as 3x3 array.

Printing the array is also commonly done with nested for loops

In [5]:
```
for(int[] row : arr2D) {
    for(int col: row)
        System.out.printf("%4d", col);
    System.out.println();
}
```

```
   1   2   3
   6   7   8
  12  13  14
```

In [23]:
```
// just adding these so it will be easier later in our code
public static void print2D(Object[][] values) {
    for(Object[] row: values) {
        for(Object col: row)
            System.out.printf("%20s", col);
```

```
        System.out.println();
    }
    System.out.println();
}

public static void print2D(int[][] values) {
    for(int[] row: values) {
        for(int col: row)
            System.out.printf("%4d", col);
        System.out.println();
    }
    System.out.println();
}
```

However, if I need to modify the values, I need to use a standard for loop!

In [24]:
```
String[][] names = {{"Superman", "Batman"}, {"Lex Luther", "Joker"}};


print2D(names);


for(int row =0; row < names.length; row++) {
    for(int col = 0; col < names[row].length; col++) { // notice!!!
        names[row][col] = names[row][col].toUpperCase();
    }

}

print2D(names); // notice names is modified!
```

```
  Superman              Batman
Lex Luther               Joker

  SUPERMAN              BATMAN
LEX LUTHER               JOKER
```

Declaring Arrays

- There are multiple ways to declare arrays
- Shorthand using the curly brackets
- Declaring the entire array at once
- Irregular / declaring on the fly

In [27]:
```
int[][] matrix = new int[3][3];


print2D(matrix); // fully initialized with 0
```

```
   0   0   0
   0   0   0
   0   0   0
```

And then modify the values of the empty array.

In [28]:
```
for(int i  = 0; i < matrix.length; i++) {
    for(int j = 0; j < matrix[i].length; j++) {
        matrix[i][j] = j + (i*10) +1;
    }
}

print2D(matrix);
```

```
 1   2   3
11  12  13
21  22  23
```

Irregular / Ragged Arrays

- You can have arrays of variable length within an array
- These are often called 'ragged' or irregular arrays

In [32]:
```
int[][] ragged = new int[3][];

System.out.println(Arrays.toString(ragged));
```

```
[null, null, null]
```

Notice: It recrated an array of 3 null values, but it will require an `int[]` to be placed in each spot.

In [60]:
```
Random rnd  = new Random();

for(int i = 0; i < ragged.length; i++) {
    ragged[i] = new int[rnd.nextInt(6)+1];
    for(int j = 0; j < ragged[i].length; j++) {
        ragged[i][j] = j + (i*10);
    }
}

print2D(ragged);
```

```
 0   1   2   3
10  11  12  13  14  15
20  21
```

In class activity

- Write a method that
  - builds an NxM array (n and m are both parameters)
  - populates it with random double values between 0 and 1
    - ```
      Random rnd = new Random();
      rnd.nextDouble(); // gives between 0 and 1
      ```
  - returns that array
    - print out the array contents
- Expert level (if you finish with the first task)
  - Create a second level where M is random 1-M creating a ragged array.
      ```

In [65]:
```
public static double[][] matrixSeed(int n, int m) {
    Random rnd = new Random();
    double[][] seed = new double[n][m];
    for(int row = 0; row < seed.length; row++) {
        for(int col = 0; col < seed[row].length; col++) {
            seed[row][col] = rnd.nextDouble();
        }
    }
    return seed;
}


public static void printSeed(double[][] seed) {
    for(double[] row:  seed) {
        for(double col: row) {
            System.out.printf("%6.2f", col);
        }
        System.out.println();
    }
    System.out.println();
}


double[][] hidden_a = matrixSeed(10, 10);
double[][] hidden_b = matrixSeed(10, 5);

printSeed(hidden_a);
printSeed(hidden_b);
```

```
  0.20  0.61  0.12  0.42  0.26  0.69  0.24  0.75  0.86  0.96
  0.17  0.90  0.04  0.40  0.05  0.25  1.00  0.08  0.89  0.04
  0.16  0.74  0.51  0.95  0.75  0.16  0.59  0.65  0.85  0.04
  0.36  0.76  0.06  0.68  0.94  0.49  0.36  0.86  0.96  0.01
  0.19  0.06  0.00  0.07  0.01  0.05  0.61  0.84  0.70  0.17
  0.32  0.08  0.69  0.02  0.48  0.15  0.43  0.03  0.34  0.03
  0.09  0.71  0.55  0.67  0.53  0.17  0.49  0.54  0.34  0.89
  0.12  0.00  0.62  0.62  0.84  0.23  0.47  0.77  0.78  0.42
  0.45  0.44  0.13  0.33  0.29  0.07  0.83  0.90  0.89  0.20
  0.49  0.48  0.34  0.05  0.72  0.74  0.91  0.96  0.36  0.14

  0.12  0.45  0.77  0.00  0.31
  0.60  0.40  0.44  0.70  0.03
  0.37  0.41  0.33  0.81  0.52
  0.44  0.17  0.48  0.82  0.14
  0.16  0.91  0.90  0.76  0.28
  0.22  0.69  0.12  0.31  0.96
  0.95  0.84  0.25  0.99  0.44
  0.68  0.00  0.13  0.04  1.00
  0.33  0.14  0.47  0.83  0.66
  0.10  0.04  0.31  0.36  0.88
```

Why am I calling it a "seed"?

This is actually common to do when working with artificial neural networks or other machine learning applications. It is the 'starting point' for learning!

Overview

- Practice arrays!
- You can even make 3D arrays, just be adding on another layer
  - You are now dealing in 3D space..
- Data science and Machine Learning often deals with N-dimensional arrays!
  - Often, you don't know the N, so making sure you break it down is important!
- This is a topic we cover the least, that you will want to practice before CS 165.