

# Arrays – 2D

---



**Colorado State University**  
Department of Computer Science

Slides Originally Created by Albert Lionelle (Albert.Lionelle@colostate.edu),  
updated by Marcia Moraes (marcia.moraes@colostate.edu)

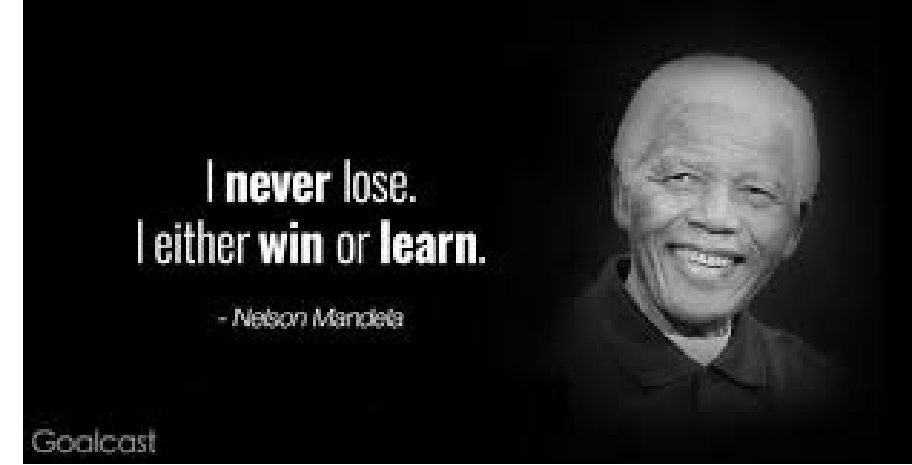
# Announcements

TODO Reminders:

Readings are due **before** lecture

- Reading 27 (zybooks) – you should have already done that 😊
- Lab 18
- Reading 28 (zyBooks)
- Lab 19
- Practical Project Part 2
- RPA 14

Keep practicing your RPAs in a spaced and mixed manner 😊



## Help Desk

Day	Time : Room
Monday	12 PM - 2 PM : CSB 120
Tuesday	6 PM - 8 PM : Teams
Wednesday	3 PM - 5 PM : CSB 120
Thursday	6 PM - 8 PM : Teams
Friday	3 PM - 5 PM : CSB 120
Saturday	12 PM - 4 PM : Teams
Sunday	12 PM - 4 PM : Teams

# Arrays – Recall and Group Code Activity

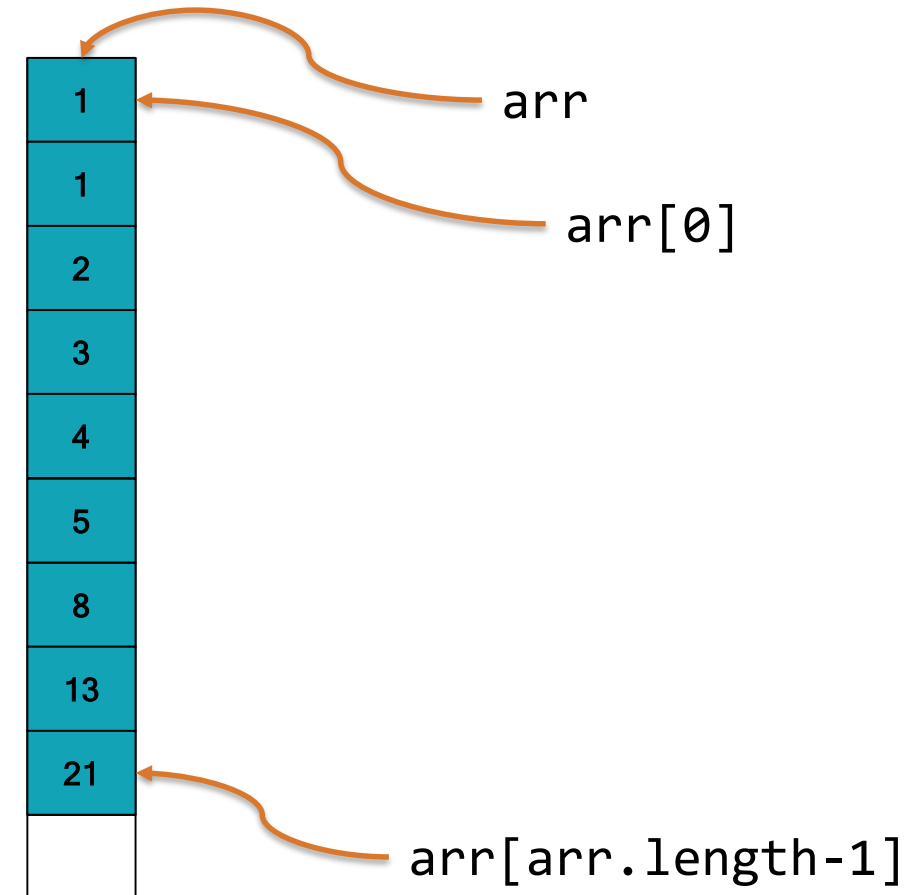
- Brainstorm in your group and write what you all can remember about arrays.

# Arrays - Reminder

- Ways to store
  - Variables in order
  - index from 0..N
- Arrays are
  - a type themselves
  - the value of the array
    - reference to **memory** location!
  - .length gives us total memory allocated
- Arrays can
  - be any size – as long as you allocate it
  - Store any valid type
    - primitives and objects

Loops are common ways to  
access elements!

```
int[] arr = {1,1,2,3,5,8,13,21};
```



# Arrays – easy access with For-Each

- For-Each Loops
  - Specialized for loops
  - Perfect for arrays or other collections
- Loops through every value (0..N)
  - Stores it in a temp variable
- Same as some very common for loops!

```
public static String[] foreachExample() {  
    String[] values = new String[2]; // string array!  
    values[0] = "Fib:";  
    values[1] = "Fib:";  
    int[] arr = {1,1,2,3,5,8,13,21};  
  
    for(int i =0; i<arr.length; i++) {  
        int ar = arr[i];  
        values[0] += ar;  
    }  
  
    for(int ar : arr) {  
        values[1] += ar;  
    }  
    return values;  
}
```

Equivalent!

[Fib:1123581321, Fib:1123581321]

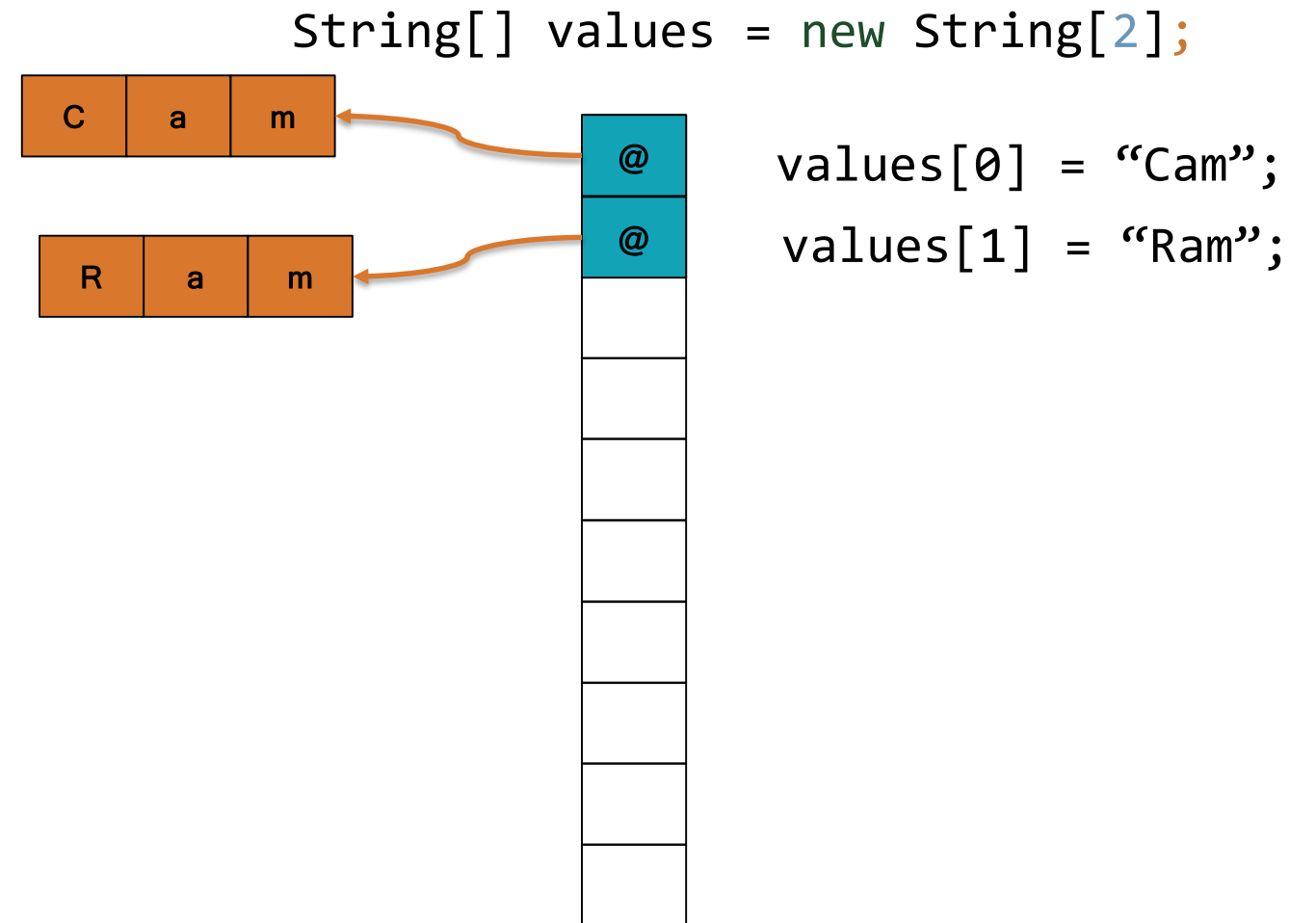
# Arrays with objects?

- primitives – values are stored
- objects – references to values

```
Box[] values = new Box[10];
```

```
MyObject[] values = new MyObject[5];
```

- Can you have an arrays of arrays?
- Arrays have type
  - Anything with type can be an array!



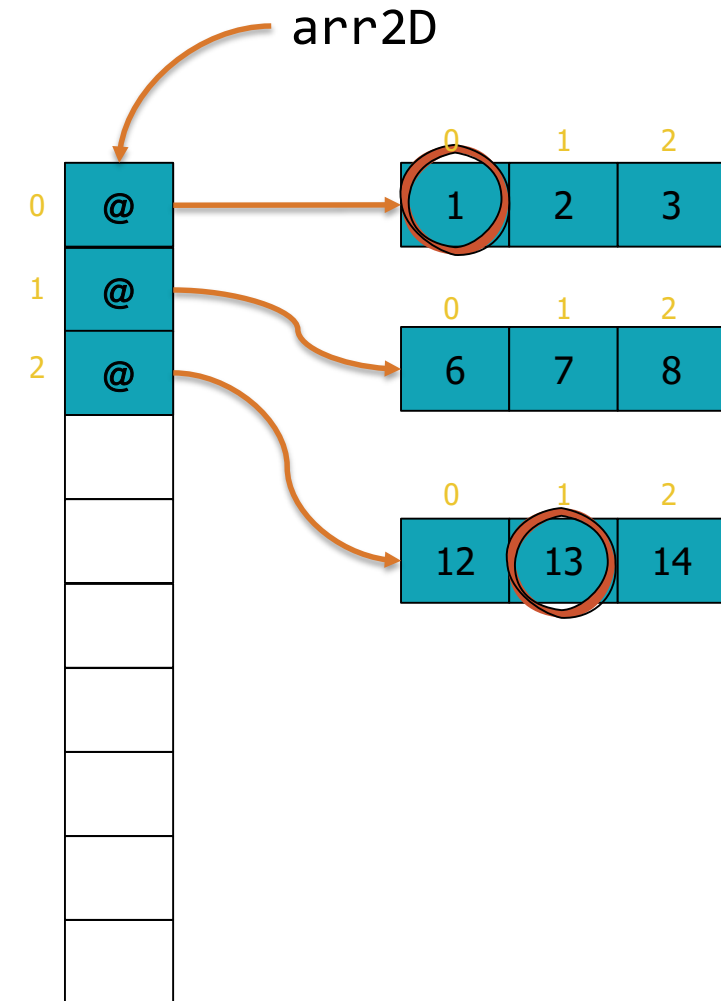
# 2D Arrays!

- 2D arrays
  - Array of arrays
  - Same type
- very common
  - enough we have a shorthand notation

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

```
System.out.println(arr2D[0][0]);
```

```
System.out.println(arr2D[2][1]);
```



# 2D Arrays - Declaring

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

```
print2D(matrix);
```

0	0	0
0	0	0
0	0	0

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



# 2D Arrays – Irregular/Ragged Arrays

- You can have arrays of variable length within an array
- Those are called “irregular or ragged” arrays

```
int ragged [][] = new int[3][];
```

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

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

```
int ragged2 [][] = new int[3][];
```

```
irregular(ragged2);
```

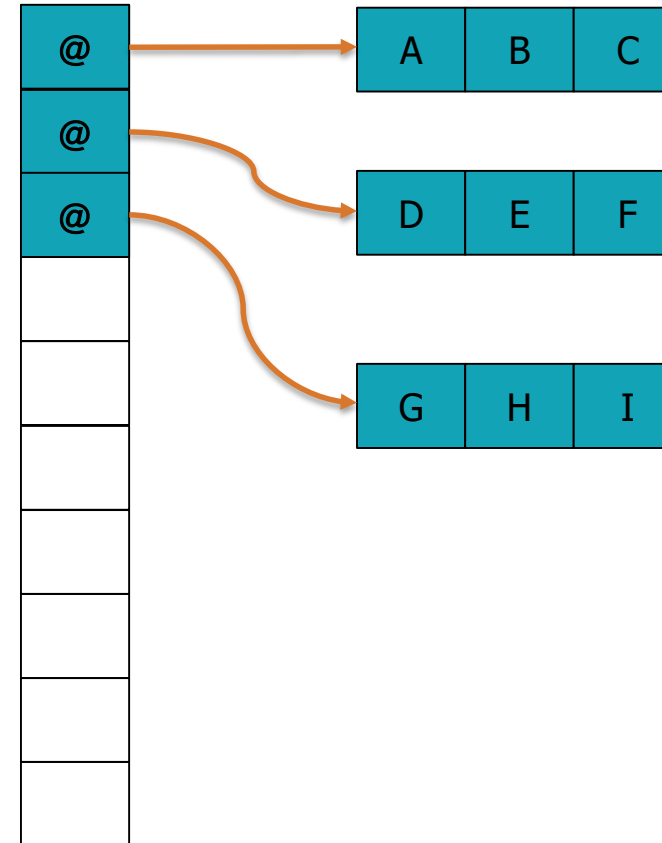
```
print2D(ragged2);
```

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

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

# 2D Arrays – Common With Nested Loops

```
public static void twoDArrays(int size) {  
    char[][] matrix = new char[size][size];  
    char start = 'A';  
    for(int i = 0; i < size; i++) {  
        for(int j = 0; j < size; j++) {  
            matrix[i][j] += (char) (start+j);  
        }  
        start = (char)(matrix[i][size-1]+1);  
    }  
    for(char[] row : matrix) {  
        for(char col : row) {  
            System.out.printf("%4d", col);  
        }  
        System.out.println();  
    }  
}  
  
public static void main(String[] args) {  
    twoDArrays(3);  
}
```



# 2D Arrays –Group Code Activity

- Write a `Array2DInt` class that:
- builds an array of 2D of int of a specific size;
- add user entered values to that 2D array;
- builds a `toString` method that return a String with the 2D array as a matrix format.

Challenge: what do you need to do to be able to read the numbers from a file instead of reading from the console?

```
public static void twoDArrays(int size) {  
    char[][] matrix = new char[size][size];  
    char start = 'A';  
    for(int i = 0; i < size; i++) {  
        for(int j = 0; j < size; j++) {  
            matrix[i][j] += (char) (start+j);  
        }  
        start = (char)(matrix[i][size-1]+1);  
    }  
    for(char[] row : matrix) {  
        for(char col : row) {  
            System.out.print(col);  
        }  
    }  
}
```

```
public static void main(String[] args) {  
    twoDArrays(3);  
}
```