Recap: CokeEmpire

- What does this return?

```java
myMachines.getColaMachine(1).getCansSold();
```

Recap: CokeEmpire

- Understanding when and how to use
  - 2D arrays

Objectives

- Understanding when and how to use
  - 2D arrays

Arrays of Arrays

- In any given array, all data must be of same type
- In any given array, all data must be of same type
- All arrays in array of arrays must be of same type
Arrays of Arrays

Two-Dimensional Arrays

- In Java, 2D array implemented internally as array of arrays
- but externally syntax of 2D array may seem easier to use

Arrays of Arrays

- In any given array, all data must be of same type
- All arrays in array of arrays must be of same type
- So easier to use a two-dimensional array!

Arrays of Arrays

- Typial control structure for computing with 2D array is nested loop
- loop within another loop
- Let's write program to
- load array with values shown
- print contents of array

Two-Dimensional Arrays

In Java, 2D array implemented internally as array of arrays but externally syntax of 2D array may seem easier to use.
Two-Dimensional Arrays

A two-dimensional array is an array of arrays. For example, the following is a two-dimensional array:

```
int[][] scores = {
    {95, 82, 13, 96},
    {51, 68, 63, 57},
    {73, 71, 86, 78},
    {50, 50, 50, 50},
    {99, 70, 32, 12}
};
```

Sorting

Computers are essential for keeping track and finding large quantities of data! Finding data when necessary is much easier when data is sorted in some way:
- computer people think a lot about how to sort things:
  - finding medical records
  - banking information
  - income tax returns
  - driver's license information...
  - even names in a phone book...
- all depend on the information being sorted

Selection sort

Let's say want to sort array values in increasing order:
- one way to approach problem is to use algorithm called selection sort

Example: Per-Student Averages

Example: Per-Quiz Averages

Example: Per-Student Averages

Example: Per-Student Averages

Sorting

Example: Per-Student Averages
Selection sort
- Let's say want to sort array values in increasing order
  - one way to approach problem is to use algorithm called selection sort
- Start by setting pointer to first element in array
  - this is where smallest value in array will be placed
- Then look at every value in this unsorted array
  - find minimum value

The smallest value so far is 3
Its index is 1

Selection sort
- Let's say want to sort array values in increasing order
  - one way to approach problem is to use algorithm called selection sort
- Start by setting pointer to first element in array
  - this is where smallest value in array will be placed
- Then look at every value in this unsorted array
  - find minimum value

At this point we know smallest number in array is in first element

In other words, do everything again to unsorted part of array
- in this case, all but first element
- start by setting pointer to first element in array
  - this is where smallest value in array will be placed
- then look at every value in this unsorted array
  - find minimum value

Selection sort
- Let's say want to sort array values in increasing order
  - one way to approach problem is to use algorithm called selection sort
- Start by setting pointer to first element in array
  - this is where smallest value in array will be placed
- Then look at every value in this unsorted array
  - find minimum value

Once we've found the minimum value
  - swap that value with the one we selected at beginning

Selection sort
- Let's say want to sort array values in increasing order
  - one way to approach problem is to use algorithm called selection sort
- Start by setting pointer to first element in array
  - this is where smallest value in array will be placed
- Then look at every value in this unsorted array
  - find minimum value

At this point we know smallest number in array is in first element

In other words, do everything again to unsorted part of array
- in this case, all but first element
- start by setting pointer to first element in array
  - this is where smallest value in array will be placed
- then look at every value in this unsorted array
  - find minimum value
Selection sort

- At this point we know:
  - smallest number in array is in first element (index 0)
  - first element is sorted
  - rest of array remains unsorted
- Now select second element of array to be location which will hold next smallest value
  - in other words, do everything again to unsorted part of array
  - in this case, all but first element

The smallest value so far is 8
Its index is 3

Now first two elements of array are sorted
Select third element of array to be location of next smallest value
Search unsorted portion of array for that value, just like before

The smallest value so far is 16
Its index is 3

Again, swap values
Selection sort

- Now first two elements of array are sorted
- Select third element of array to be location of next smallest value
- Search unsorted portion of array for that value, just like before
- Again, swap values

Again, swap values

Then do whole thing again

The smallest value so far is 16
Its index is 3

Selection sort

- Now first two elements of array are sorted
- Select third element of array to be location of next smallest value
- Search unsorted portion of array for that value, just like before
- Again, swap values

Selection sort

- Now first two elements of array are sorted
- Select third element of array to be location of next smallest value
- Search unsorted portion of array for that value, just like before
- Again, swap values

Selection sort

- Now first two elements of array are sorted
- Select third element of array to be location of next smallest value
- Search unsorted portion of array for that value, just like before
- Again, swap values

Selection sort

- Now first two elements of array are sorted
- Select third element of array to be location of next smallest value
- Search unsorted portion of array for that value, just like before
- Again, swap values

Selection sort

- Showed arrows moving down array
- Red arrow on left represents one array index variable
- Yellow arrow on right represents different one
- Consider variables being controlled by loop
- Red arrow shows outer loop
- Yellow arrow shows inner loop inside outer loop
- Nested loop structure again
Selection sort

// selection sort
public class SelectionSort {
    public static void main(String[] args) {
        int[] numbers = {16, 3, 19, 8, 12};
        int max, temp;
        // find location of next smallest value
        for (int i = 0; i < numbers.length; i++) {
            max = i;
            // find the smallest value in the remainder of
            // the array to be sorted
            for (int j = i + 1; j < numbers.length; j++)
                if (numbers[j] < numbers[max])
                    max = j;
            // swap two values in the array
            temp = numbers[i];
            numbers[i] = numbers[max];
            numbers[max] = temp;
        }
        System.out.println("sorted array:");
        for (int i = 0; i < numbers.length; i++)
            System.out.print(numbers[i] + " ");
    }
}

Selection sort

// selection sort
public class SelectionSort {
    public static void main(String[] args) {
        int[] numbers = {16, 3, 19, 8, 12};
        int max, temp;
        // find location of next smallest value
        for (int i = 0; i < numbers.length; i++) {
            max = i;
            // find the smallest value in the remainder of
            // the array to be sorted
            for (int j = i + 1; j < numbers.length; j++)
                if (numbers[j] < numbers[max])
                    max = j;
            // swap two values in the array
            temp = numbers[i];
            numbers[i] = numbers[max];
            numbers[max] = temp;
        }
        System.out.println("sorted array:");
        for (int i = 0; i < numbers.length; i++)
            System.out.print(numbers[i] + " ");
    }
}

Selection sort

// selection sort
public class SelectionSort {
    public static void main(String[] args) {
        int[] numbers = {16, 3, 19, 8, 12};
        int max, temp;
        // find location of next smallest value
        for (int i = 0; i < numbers.length; i++) {
            max = i;
            // find the smallest value in the remainder of
            // the array to be sorted
            for (int j = i + 1; j < numbers.length; j++)
                if (numbers[j] < numbers[max])
                    max = j;
            // swap two values in the array
            temp = numbers[i];
            numbers[i] = numbers[max];
            numbers[max] = temp;
        }
        System.out.println("sorted array:");
        for (int i = 0; i < numbers.length; i++)
            System.out.print(numbers[i] + " ");
    }
}

Selection sort

// selection sort
public class SelectionSort {
    public static void main(String[] args) {
        int[] numbers = {16, 3, 19, 8, 12};
        int max, temp;
        // find location of next smallest value
        for (int i = 0; i < numbers.length; i++) {
            max = i;
            // find the smallest value in the remainder of
            // the array to be sorted
            for (int j = i + 1; j < numbers.length; j++)
                if (numbers[j] < numbers[max])
                    max = j;
            // swap two values in the array
            temp = numbers[i];
            numbers[i] = numbers[max];
            numbers[max] = temp;
        }
        System.out.println("sorted array:");
        for (int i = 0; i < numbers.length; i++)
            System.out.print(numbers[i] + " ");
    }
}
Selection sort

```java
public static void main(String[] args) {
    int[] numbers = {16, 3, 19, 8, 12};
    int temp, min;
    // find the smallest value in the remainder of the array to be sorted
    for (int i = 0; i < numbers.length - 1; i++) {
        min = i;
        for (int j = i + 1; j < numbers.length; j++) {
            if (numbers[j] < numbers[min]) {
                min = j;
            }
        }
        // swap two values in the array
        temp = numbers[i];
        numbers[i] = numbers[min];
        numbers[min] = temp;
        System.out.println(numbers[i]);
    }
}
```

Selection sort

```java
public static void main(String[] args) {
    int[] numbers = {16, 3, 19, 8, 12};
    int temp, min;
    // find the smallest value in the remainder of the array to be sorted
    for (int i = 0; i < numbers.length - 1; i++) {
        min = i;
        for (int j = i + 1; j < numbers.length; j++) {
            if (numbers[j] < numbers[min]) {
                min = j;
            }
        }
        // swap two values in the array
        temp = numbers[i];
        numbers[i] = numbers[min];
        numbers[min] = temp;
        System.out.println(numbers[i]);
    }
}
```

Selection sort

```java
public static void main(String[] args) {
    int[] numbers = {16, 3, 19, 8, 12};
    int temp, min;
    // find the smallest value in the remainder of the array to be sorted
    for (int i = 0; i < numbers.length - 1; i++) {
        min = i;
        for (int j = i + 1; j < numbers.length; j++) {
            if (numbers[j] < numbers[min]) {
                min = j;
            }
        }
        // swap two values in the array
        temp = numbers[i];
        numbers[i] = numbers[min];
        numbers[min] = temp;
        System.out.println(numbers[i]);
    }
}
```
// Selection sort
public class SortTest1
{
    public static void main(String[] args)
    {
        int[] numbers = {16,3,19,8,12};
        int min, temp;
        // select location of each sorted value
        for (int i = 0; i < numbers.length-1; i++)
        {
            min = i;
            // find the smallest value in the remainder of the array to be sorted
            for (int j = i+1; j < numbers.length; j++)
            {   // numbers[j] < numbers[min]
                min = j;
            }
            // swap two values in the array
            temp = numbers[i];
            numbers[i] = numbers[min];
            numbers[min] = temp;
            System.out.println("Printing sorted result");
            for (int k = 0, i < numbers.length; i++)
            {   System.out.print(numbers[i] + " ");
            }
        }
    }
}

// Selection sort
public class SortTest1
{
    public static void main(String[] args)
    {
        int[] numbers = {16,3,19,8,12};
        int min, temp;
        // select location of each sorted value
        for (int i = 0; i < numbers.length-1; i++)
        {
            min = i;
            // find the smallest value in the remainder of the array to be sorted
            for (int j = i+1; j < numbers.length; j++)
            {   // numbers[j] < numbers[min]
                min = j;
            }
            // swap two values in the array
            temp = numbers[i];
            numbers[i] = numbers[min];
            numbers[min] = temp;
            System.out.println("Printing sorted result");
            for (int k = 0, i < numbers.length; i++)
            {   System.out.print(numbers[i] + " ");
            }
        }
    }
}

// Selection sort
public class SortTest1
{
    public static void main(String[] args)
    {
        int[] numbers = {16,3,19,8,12};
        int min, temp;
        // select location of each sorted value
        for (int i = 0; i < numbers.length-1; i++)
        {
            min = i;
            // find the smallest value in the remainder of the array to be sorted
            for (int j = i+1; j < numbers.length; j++)
            {   // numbers[j] < numbers[min]
                min = j;
            }
            // swap two values in the array
            temp = numbers[i];
            numbers[i] = numbers[min];
            numbers[min] = temp;
            System.out.println("Printing sorted result");
            for (int k = 0, i < numbers.length; i++)
            {   System.out.print(numbers[i] + " ");
            }
        }
    }
}
Selection sort

// selection sort
public class SortTest1
{
    public static void main(String[] args)
    {
        int[] numbers = {16,3,19,8,12};
        int max, temp;
        //Initial location of each sorted value
        for (int i = 0; i < numbers.length-1; i++)
        {
            min = i;
            //Find the smallest value in the remainder of
            //the array to be sorted
            for (int j = i; j < numbers.length; j++)
            {
                if (numbers[j] < numbers[min])
                {
                    min = j;
                }
            }
            //swap the values in the array
            temp = numbers[i];
            numbers[i] = numbers[min];
            numbers[min] = temp;
            System.out.print("Printing sorted result:");
            for (int k = 0; k < numbers.length-1; k++)
            {
                System.out.print(numbers[k] + " ");
            }
            System.out.print(numbers[numbers.length-1] + "");
            System.out.println();
            System.out.println();
        }
    }
}

Selection sort

// selection sort
public class SortTest1
{
    public static void main(String[] args)
    {
        int[] numbers = {16,3,19,8,12};
        int max, temp;
        //Initial location of each sorted value
        for (int i = 0; i < numbers.length-1; i++)
        {
            min = i;
            //Find the smallest value in the remainder of
            //the array to be sorted
            for (int j = i; j < numbers.length; j++)
            {
                if (numbers[j] < numbers[min])
                {
                    min = j;
                }
            }
            //swap the values in the array
            temp = numbers[i];
            numbers[i] = numbers[min];
            numbers[min] = temp;
            System.out.print("Printing sorted result:");
            for (int k = 0; k < numbers.length-1; k++)
            {
                System.out.print(numbers[k] + " ");
            }
            System.out.print(numbers[numbers.length-1] + "");
            System.out.println();
            System.out.println();
        }
    }
}

Selection sort

// selection sort
public class SortTest1
{
    public static void main(String[] args)
    {
        int[] numbers = {16,3,19,8,12};
        int max, temp;
        //Initial location of each sorted value
        for (int i = 0; i < numbers.length-1; i++)
        {
            min = i;
            //Find the smallest value in the remainder of
            //the array to be sorted
            for (int j = i; j < numbers.length; j++)
            {
                if (numbers[j] < numbers[min])
                {
                    min = j;
                }
            }
            //swap the values in the array
            temp = numbers[i];
            numbers[i] = numbers[min];
            numbers[min] = temp;
            System.out.print("Printing sorted result:");
            for (int k = 0; k < numbers.length-1; k++)
            {
                System.out.print(numbers[k] + " ");
            }
            System.out.print(numbers[numbers.length-1] + "");
            System.out.println();
            System.out.println();
        }
    }
}
Selection sort
// selection sort
public class SortTest1
{
  public static void main(String[] args)
  {
    int[] numbers = [12, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1];
    int min, temp;
    //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);
    } //find the smallest value in the remainder of
    //the array to be sorted
    for (int i = 0; i < numbers.length-1; i++)
      if (numbers[i] < numbers[min])
        min = i;
      } //swap two values in the array
    swap(numbers[i], numbers[min]);
    System.out.println(numbers[i] + " swapped with " + numbers[min] + " at index " + i);