Unit 6 - Lesson 6
Comparing Strings

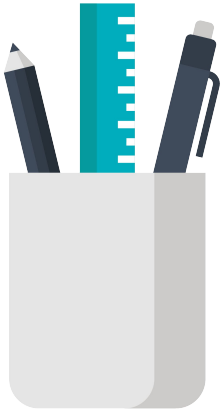
Computer Science A

Warm Up

CSA Bingo

📍 **You should have:**

- a CS Bingo card
- pen / pencil / marker



CS Bingo

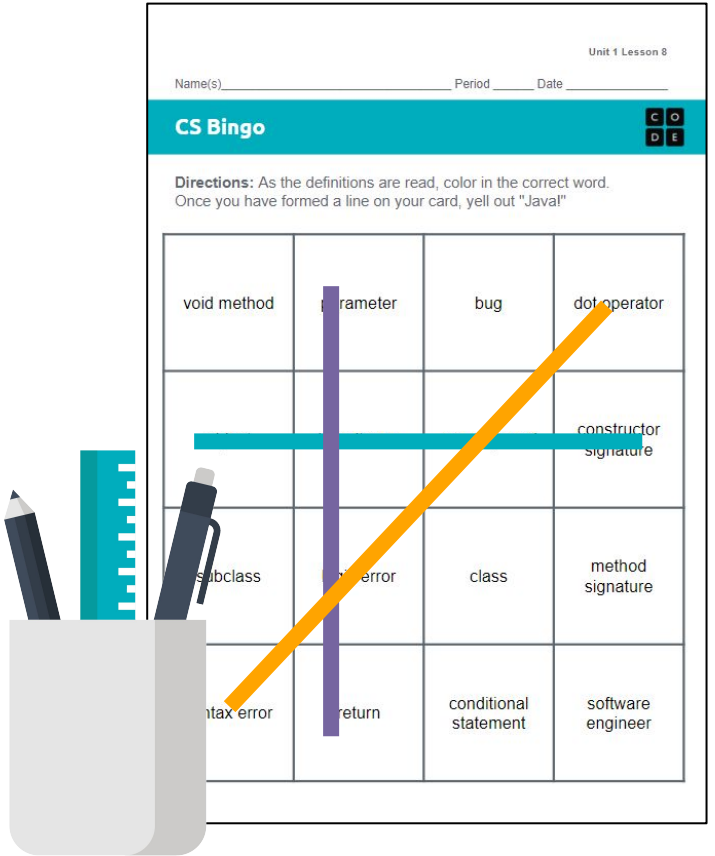
Directions: As the definitions are read, color in the correct word. Once you have formed a line on your card, yell out "Java!"

mutable	pixel	generic type	static data structure
ArrayList	underflow error	inner array	overflow error
RGB	Javadocs	2D array	column
HTML	outer array	documentation	dynamic data structure

CSA Bingo

As the definitions are read, **color in the correct word** on your **CS Bingo card**.

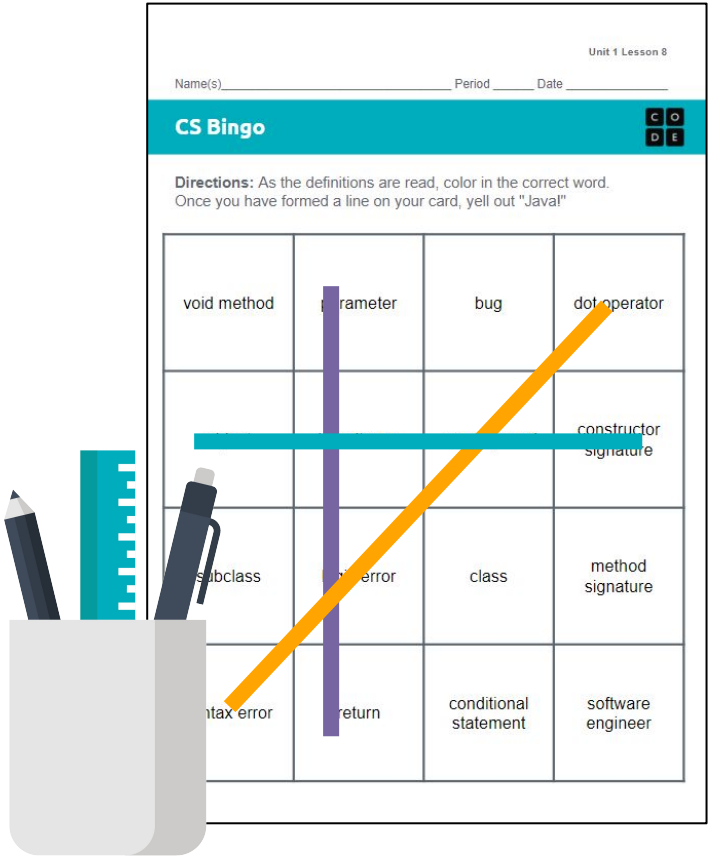
Once you have **formed a line** on your card, yell out **"Java!"**



CS Bingo!

As the definitions are read, **color in the correct word** on your **CS Bingo card**.

Once you have **formed a line** on your card, yell out **"Java!"**



Activity

🎯 Lesson Objectives

By the end of this lesson, you will be able to . . .

- Use the **`compareTo()`** method to check if one **`String`** object comes before or after another **`String`** object
- Implement algorithms using methods in the **`String`** class

🚀 Question of the Day

How can I determine if a list of **`String`** objects are in alphabetical order?

Predict and Run

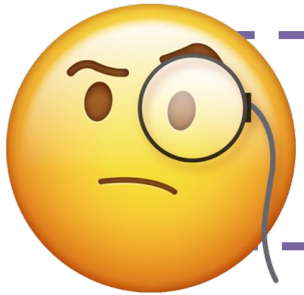
Navigate to Lesson 6, Level 1

✅ Do This:

1. Predict the output of the program

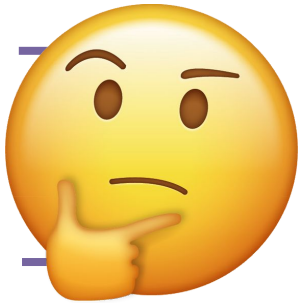
   There are no wrong answers!

2. Run it to compare your prediction with the results



What did you notice about the code in this program?

What do you wonder about the code in this program?



Lexicographical order means placing words in alphabetical order.

Investigate and Modify

Navigate to Lesson 6, Level 2

✅ Do This:

1. Investigate the code on **Levels 2 through 4**
2. Make changes as prompted and observe the results

What did you discover from the modifications you made to the code?

📝 Unit 6 Guide

`int compareTo(String anotherString)` returns a **negative** integer if ***this*** **`String`** comes *before* the argument **`String`**, a **positive** integer if ***this*** **`String`** comes *after* the argument **`String`**, and **`0`** if the **`String`** objects contain the *same characters*.

```
String firstWord = "Hello";
String secondWord = "HELLO";
String thirdWord = "Java";
System.out.println(firstWord.compareTo(thirdWord));
System.out.println(firstWord.compareTo(secondWord));
System.out.println(thirdWord.compareTo(firstWord));
```

```
-2
32
2
```

The **`compareTo()`** method is case-sensitive, so **`"Hello"`** and **`"HELLO"`** are not considered equal.

values that are returned are based on the ode value of each character in the **`String`**.

Self Check

What will be printed after this code segment is executed?

```
String str1 = "apple";
String str2 = "banana";
System.out.println(str1.compareTo(str2));
```

A `0`

B `1`

C `-1`

D `"apple"`

E `"banana"`

🎉 Practice

Navigate to Lesson 6, Level 5

✅ Do This:

1. **Level 5 –** Check for Understanding
2. **Level 6 –** Practice using the `compareTo()` method
3. **Level 7 –** Practice using methods in the `String` class

Wrap Up

Three W's

1. What did we learn today?
2. So what?
3. Now what?

🎉 Today, you learned about . . .

- Using the **`compareTo()`** method to check if one **`String`** object comes before or after another **`String`** object
- Implementing algorithms using methods in the **`String`** class

🚀 Question of the Day

How can I determine if a list of **`String`** objects are in alphabetical order?

💼 Key Vocabulary

- **lexicographical order:** placing words in alphabetical order