

Unit 4 - Lesson 12

BingoCaller FRQ



Warm Up



FRQ Strategy

- **Step 1:** Annotate
- **Step 2:** Pseudocode
- **Step 3:** Implement





BingoCaller FRQ

 **You should have:**

- BingoCaller FRQ
- scratch paper
- pen / pencil



Name(s) _____ Period _____ Date _____

Activity Guide - BingoCaller FRQ

C O
D E

BingoCaller FRQ

In a game called Bingo, players mark off numbers on cards as the numbers are called randomly by a "caller." Each bingo card has 25 spaces arranged in a square of 5 rows and 5 columns. The columns are labeled from left to right with the letters "B", "I", "N", "G", "O".

The spaces in the card are assigned random values as follows:

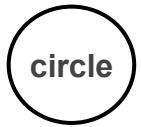
- Each space in the "B" column contains a number from 1 to 15, inclusive.
- Each space in the "I" column contains a number from 16 to 30, inclusive.
- Each space in the "N" column contains a number from 31 to 45, inclusive.
- Each space in the "G" column contains a number from 46 to 60, inclusive.
- Each space in the "O" column contains a number from 61 to 75, inclusive.

When a game of Bingo is played, each number can only be called once. A bingo call contains a column letter followed by a value (for example, "B7").

1

Do This:

Annotate the question using the following shapes and symbols.



conditions (look for keywords such as **if**, when, as long as something is **true** or **false** or within certain conditions, etc.)



information to return or output as a result of an action



variables / lists / methods used either to return / output or needed to perform an action



parameters or information needed to perform an action to produce a result

underline

words or concepts you are not sure of

 **Do This:**

Compare your **annotations**
with your **partner**.





Discuss:


What questions do you have about approaching this free response question?

Activity





Step 2: Pseudocode

 **Do This:** On your scratch paper, **plan** your solutions for Parts A, B, and C using **pseudocode**.



Name(s) _____ Period _____ Date _____

Activity Guide - BingoCaller FRQ

BingoCaller FRQ

In a game called Bingo, players mark off numbers on cards as the numbers are called randomly by a "caller." Each bingo card has 25 spaces arranged in a square of 5 rows and 5 columns. The columns are labeled from left to right with the letters "B", "I", "N", "G", "O".

The spaces in the card are assigned random values as follows:

- Each space in the "B" column contains a number from 1 to 15, inclusive.
- Each space in the "I" column contains a number from 16 to 30, inclusive.
- Each space in the "N" column contains a number from 31 to 45, inclusive.
- Each space in the "G" column contains a number from 46 to 60, inclusive.
- Each space in the "O" column contains a number from 61 to 75, inclusive.

When a game of Bingo is played, each number can only be called once. A bingo call contains a column letter followed by a value (for example, "B7").

1



Step 3: Implement

 **Do This:** Translate your **pseudocode** to **Java** on the **BingoCaller FRQ** activity guide.



Name(s) _____ Period _____ Date _____

Activity Guide - BingoCaller FRQ C O
D E

BingoCaller FRQ

In a game called Bingo, players mark off numbers on cards as the numbers are called randomly by a "caller." Each bingo card has 25 spaces arranged in a square of 5 rows and 5 columns. The columns are labeled from left to right with the letters "B", "I", "N", "G", "O".

The spaces in the card are assigned random values as follows:

- Each space in the "B" column contains a number from 1 to 15, inclusive.
- Each space in the "I" column contains a number from 16 to 30, inclusive.
- Each space in the "N" column contains a number from 31 to 45, inclusive.
- Each space in the "G" column contains a number from 46 to 60, inclusive.
- Each space in the "O" column contains a number from 61 to 75, inclusive.

When a game of Bingo is played, each number can only be called once. A bingo call contains a column letter followed by a value (for example, "B7").

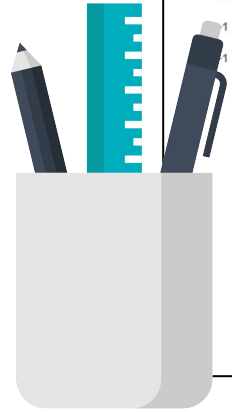
1



BingoCaller FRQ Scoring Guidelines

You should have:

- BingoCaller FRQ Scoring Guidelines
- pen / pencil



Name(s) _____ Period _____ Date _____

Activity Guide - BingoCaller FRQ Scoring Guidelines C O D E

Part (a) `hasBeenCalled()` **2 points**

- +1 Accesses the correct entry in `numbersCalled` (*no bounds error*)
- +1 Returns a `boolean` value

Part (b) `getRandomNumber()` **2 points**

- +1 Generates a random integer within the correct range based on the parameter `letter`
- +1 Returns an `int` value

Part (c) `makeCall()` **5 points**

- +1 Selects a random letter from `COLUMN_LETTERS`
- +1 Uses `getRandomNumber()` to get a random number based on the random letter chosen
- +1 Checks whether the bingo call has been previously made
- +1 Returns a `String` containing the correct result based on whether the bingo call has been previously made
- +1 Marks the corresponding bingo call as `true` in `numbersCalled` (*no bounds error*)

1



Do This:

Assess your solution using the **BingoCaller FRQ Scoring Guidelines** with your partner.

Complete the **reflection** on the **last page** of the **BingoCaller FRQ** activity guide.



Name(s) _____ Period _____ Date _____

Activity Guide - BingoCaller FRQ Scoring Guidelines

C
D
O
E

Part (a)	<code>hasBeenCalled()</code>	2 points
<ul style="list-style-type: none"> +1 Accesses the correct entry in <code>numbersCalled</code> (<i>no bounds error</i>) +1 Returns a <code>boolean</code> value 		
Part (b)	<code>getRandomNumber()</code>	2 points
<ul style="list-style-type: none"> +1 Generates a random integer within the correct range based on the parameter <code>letter</code> +1 Returns an <code>int</code> value 		
Part (c)	<code>makeCall()</code>	5 points
<ul style="list-style-type: none"> +1 Selects a random letter from <code>COLUMN_LETTERS</code> +1 Uses <code>getRandomNumber()</code> to get a random number based on the random letter chosen +1 Checks whether the bingo call has been previously made +1 Returns a <code>String</code> containing the correct result based on whether the bingo call has been previously made +1 Marks the corresponding bingo call as <code>true</code> in <code>numbersCalled</code> (<i>no bounds error</i>) 		

1

```
public static boolean hasBeenCalled() {  
    if (num < 1 || num > 75) {  
        return false;  
    }  
    else {  
        return numbersCalled[num - 1];  
    }  
}
```

hasBeenCalled()

Scoring Guidelines

Returns `false` if the given number is invalid (less than 0 or greater than 75)



```
public static boolean hasBeenCalled() {  
  
    if (num < 1 || num > 75) {  
        return false;  
    }  
    else {  
        return numbersCalled[num - 1];  
    }  
  
}
```

hasBeenCalled()

Scoring Guidelines

Returns the correct entry in
`numbersCalled` (*no bounds error*)



```
public static int getRandomNumber() {
    int randomValue;

    if (letter.equals("B")) {
        randomValue = (int)(Math.random() * 15) + 1;
    }
    else if (letter.equals("I")) {
        randomValue = (int)(Math.random() * 15) + 16;
    }
    else if (letter.equals("N")) {
        randomValue = (int)(Math.random() * 15) + 31;
    }
    else if (letter.equals("G")) {
        randomValue = (int)(Math.random() * 15) + 46;
    }
    else {
        randomValue = (int)(Math.random() * 15) + 61;
    }

    return randomValue;
}
```

getRandomNumber()

Scoring Guidelines

Generates a random integer within the correct range based on the parameter **letter**





```
public static int getRandomNumber() {  
    int randomValue;  
  
    if (letter.equals("B")) {  
        randomValue = (int)(Math.random() * 15) + 1;  
    }  
    else if (letter.equals("I")) {  
        randomValue = (int)(Math.random() * 15) + 16;  
    }  
    else if (letter.equals("N")) {  
        randomValue = (int)(Math.random() * 15) + 31;  
    }  
    else if (letter.equals("G")) {  
        randomValue = (int)(Math.random() * 15) + 46;  
    }  
    else {  
        randomValue = (int)(Math.random() * 15) + 61;  
    }  
    return randomValue;  
}
```

getRandomNumber()



Scoring Guidelines
Returns an int value



```
public static String makeCall() {  
    int randomIndex = (int)(Math.random() * COLUMN_LETTERS.length);  
    String randomLetter = COLUMN_LETTERS[randomIndex];  
    int randomNumber = getRandomNumber(randomLetter);  
    String result = "";  
  
    if (hasBeenCalled(randomNumber)) {  
        result = randomLetter + randomNumber + " has already been called";  
    }  
    else {  
        result = randomLetter + randomNumber;  
        numbersCalled[randomNumber - 1] = true;  
    }  
  
    return result;  
}
```



makeCall()



Scoring Guidelines

Selects a random letter from
COLUMN_LETTERS

```
public static String makeCall() {  
    int randomIndex = (int)(Math.random() * COLUMN_LETTERS.length);  
    String randomLetter = COLUMN_LETTERS[randomIndex];  
    int randomNumber = getRandomNumber(randomLetter);  
    String result = "";  
  
    if (hasBeenCalled(randomNumber)) {  
        result = randomLetter + randomNumber + " has already been called";  
    }  
    else {  
        result = randomLetter + randomNumber;  
        numbersCalled[randomNumber - 1] = true;  
    }  
  
    return result;  
}
```



makeCall()



Scoring Guidelines

Uses `getRandomNumber()` to get a random number based on the random letter chosen

```
public static String makeCall() {  
    int randomIndex = (int)(Math.random() * COLUMN_LETTERS.length);  
    String randomLetter = COLUMN_LETTERS[randomIndex];  
    int randomNumber = getRandomNumber(randomLetter);  
    String result = "";  
  
    if (hasBeenCalled(randomNumber)) {  
        result = randomLetter + randomNumber + " has already been called";  
    }  
    else {  
        result = randomLetter + randomNumber;  
        numbersCalled[randomNumber - 1] = true;  
    }  
  
    return result;  
}
```



makeCall()



Scoring Guidelines

Uses `hasBeenCalled()` to check whether the bingo call has been previously made

```
public static String makeCall() {  
    int randomIndex = (int)(Math.random() * COLUMN_LETTERS.length);  
    String randomLetter = COLUMN_LETTERS[randomIndex];  
    int randomNumber = getRandomNumber(randomLetter);  
    String result = "";
```

```
    if (hasBeenCalled(randomNumber)) {  
        result = randomLetter + randomNumber + " has already been called";  
    }  
    else {  
        result = randomLetter + randomNumber;  
        numbersCalled[randomNumber - 1] = true;  
    }  
}
```

```
    return result;
```

```
}
```

makeCall()

Scoring Guidelines



Returns a String containing the correct result based on whether the bingo call has been previously made



```
public static String makeCall() {
    int randomIndex = (int)(Math.random() * COLUMN_LETTERS.length);
    String randomLetter = COLUMN_LETTERS[randomIndex];
    int randomNumber = getRandomNumber(randomLetter);
    String result = "";

    if (hasBeenCalled(randomNumber)) {
        result = randomLetter + randomNumber + " has already been called";
    }
    else {
        result = randomLetter + randomNumber;
        numbersCalled[randomNumber - 1] = true;
    }

    return result;
}
```



makeCall()



Scoring Guidelines

Marks the corresponding bingo call as `true` in `numbersCalled` (*no bounds error*)

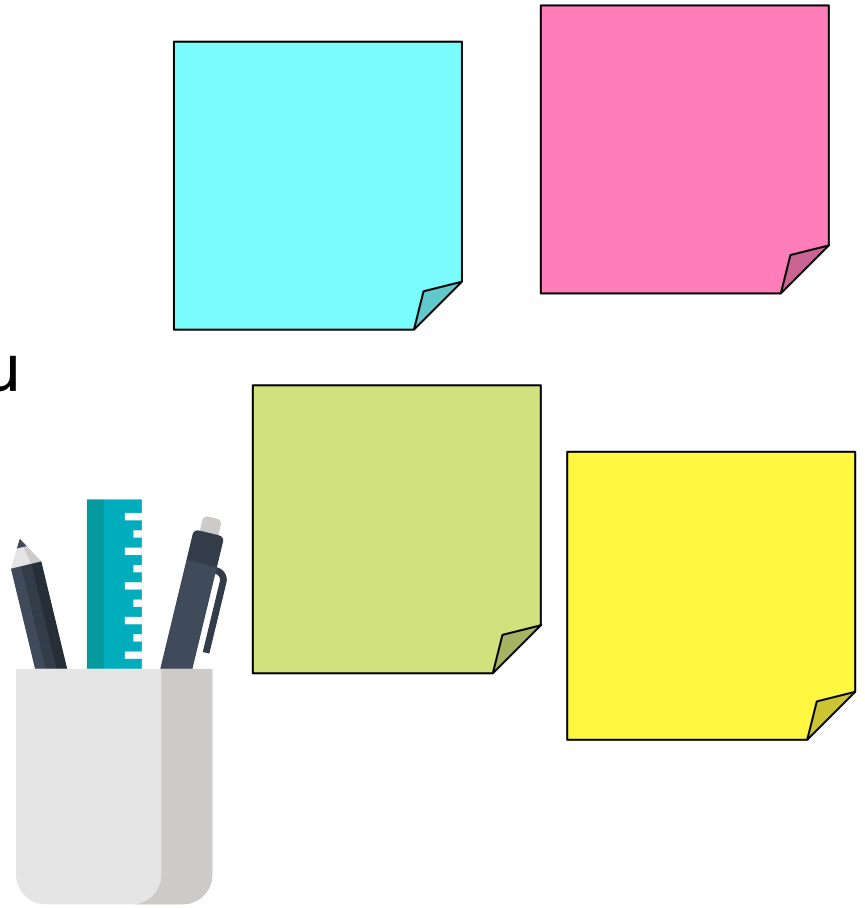
Wrap Up





 **Do This:**

On a sticky note or scrap piece of paper, write down the knowledge and skills you used on this FRQ that you learned in this unit.





Give One, Get One!

Share your **ideas** and **thoughts** with **your peers**.

- **Give** one of your ideas.
- **Get** one of your peer's ideas.

Find another classmate and repeat!

