Offline Questions

1. Say you wanted to create a method that would check whether or not a certain character was a vowel or not.

   What are the possible ways to implement it?

   If statement with ors and contain.

   Can you do it without logical operators? (|| &&)

   I can do it with many different if statements

   Can you do it without if statements?

   I can't do it without an if statement.

   Could you implement this in ONE LINE?

   I can't. It's likely to be possible though.

Write the question based on logic

   a. String word = in.nextLine();
   b. If (word.contains(a) || word.contains(e) || word.contains(i) || word.contains(o) || word.contains(u))

2. In the code below, follow an initial call to methodE(). In what order do you access the methods?
   Fill in the method calls.

```
methodA(){
        // calls method B and then C
}
methodB(){
        // Calls method D and then C
}
methodC(){
        // Calls method D and then F
}
methodD(){
        // Calls no other method
}
methodE(){
```

```
        // Calls method A
}
methodF(){
        // Calls method D
}
```

Method E -> Method A -> Method B -> Method D
-> Method C -> Method D
-> Method F -> Method D
-> Method C -> Method D
->Method F -> Method D

Censorship

Driver: Froilan Zarate
Scribe: Alan Xiao

1. Driver: In folder lab07 create a new project (name it whatever you want). Create a class within it called Censorship.
   Censor should start like this:
   The Censorship class may look clunky right now, since you can only establish three words. In future labs, you'll implement loops and lists to provide more elegant solutions to these kinds of problems

```
public class Censorship
{
        public Censorship(String firstWord, String secondWord,
String thirdWord)
        {

      }

      public boolean containsBadWord(String str)
      {
              return false;
      }

              public boolean isBadWord(String str){
              return false;
     }
              public String censorBadWords(String str){
                      return "";
              }
   }
```

Add three instance variables **bannedWordOne, bannedWordTwo, and bannedWordThree** of class **String.** Remember to initialize these in the constructor!

Scribe: Did this compile?

Yes

Complete the isBadWord() method and compile again.

Remember, if you banned the word ‘apple’ then you wouldn’t want someone typing ApPLE in order to get around your censor!

Scribe: How many lines of code did it take?

One line

Scribe and Driver: What are the important differences between contains and equals?

Contains can look for a portion of a string, while equals must be the exact string

Test the isBadWord() method in the BlueJ workbench

Create an object with the following three strings:

apple, pear, lime // You have a severe fruit salad allergy
Scribe: Does ‘aPpLe’ return true? Write down the words you tested with (case sensitive in the report).

It does return true. We also tested initial object “lol” and “LOL” which also returned true

#javatip In the previous version of the lab, we introduced a way to turn a single Character into a String. Some details were removed, and we decided to leave this trick for you in this lab.

Strings can be created and appended by literally adding them together.
For example:

```
char c = ‘a’
String s = “” + c // will result in string s containing just “a”
String t = “hello”+ c // will result in string t containing
“helloa”
```

Your code currently can determine whether or not a string is a banned word. But what if someone puts a bad word inside of another word? Or disguises it?

Let’s assume that any version of the word ‘apple’ is illegal, which means that

‘pineapple’ would be a problem, but your isBadWord() won’t catch that since the word doesn’t EXACTLY match.

Complete the containsBadWord() method. To do this, you’re going to use a different method than .equals():

```
public boolean contains(CharSequence s)
```

For now, don’t worry about CharSequence, String is derived from CharSequence, and you can use String here instead.
Used in the following way:

```
boolean result = (stringname).contains("(desiredSequence)");
```

Remember, if you banned the word ‘apple’ then you wouldn’t want someone typing pInEApPLE in order to get around your censor!
Scribe: How many lines of code did it take?
One line

Scribe: did your partner use local variables? Did they use the `toLowerCase()` or `toUpperCase()`? Did they use if statements?

1 return statement, no case changes or if statements

Now you’re going to change the access modifier of **containsBadWord** from **public** to **private**.

Scribe: Is the containsBadWord method available to be directly called in the main method?

No

Classes will often have private methods that can be called within the class but not accessible from outside of the class. We call these private helper methods.

We will call **containsBadWord** from **censorBadWords** which is still public. This means that we can use the **containsBadWord** method through **censorBadWords.**

Complete the **censorBadWords** method using the String replace methods.

// When you censor something, you don’t need to worry about the number of characters being censored. *Replace it with **** (four asterisks)* every time.
Scribe: How many replacement methods are there? Which one(s) do you think you will use for this method? (A link to the Javadoc is provided below).
https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/String.html

There are 4. We used replaceAll.

Test your **censorBadWords** in the BlueJ workbench. Try to hide words within other words, or put whole sentences in.

Scribe: What words did you use?

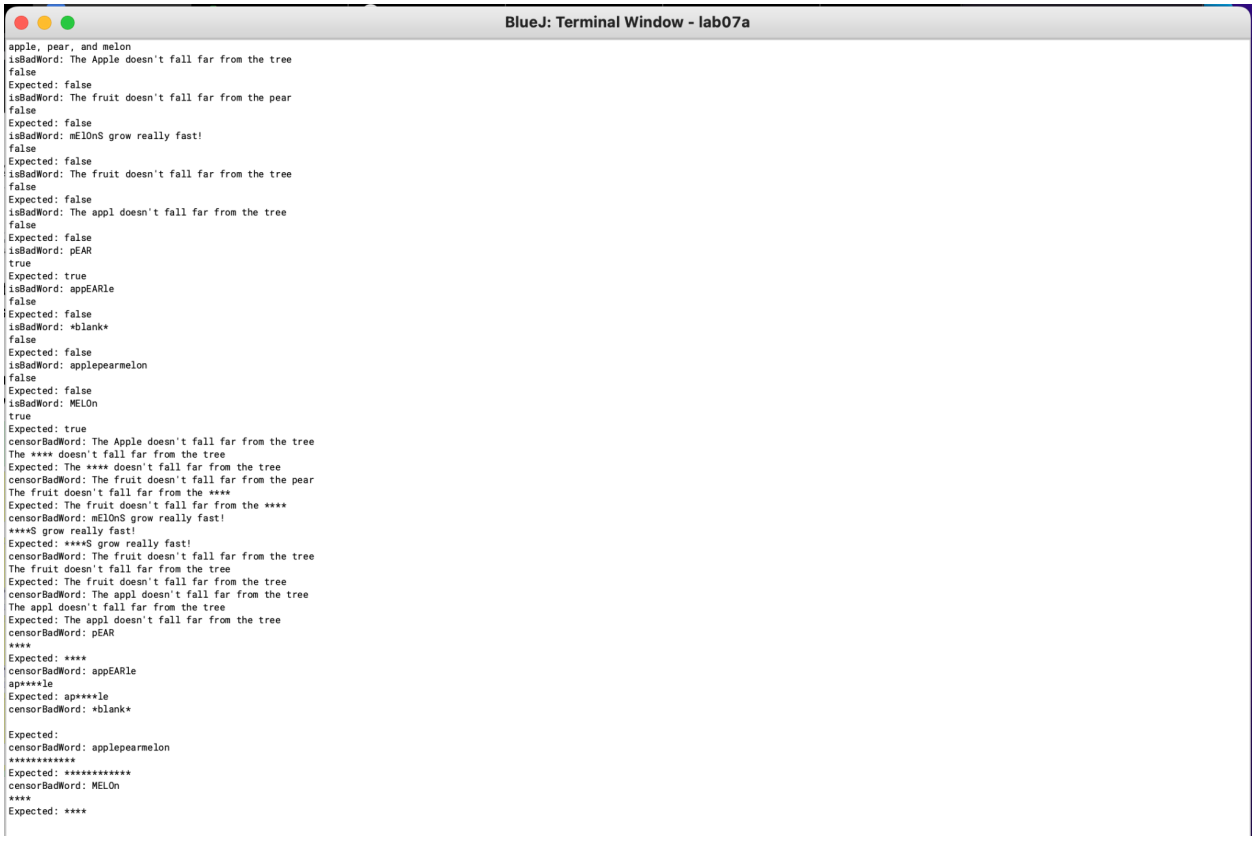
Object: Genshin, Impact, Sucks
Tested word: i love genshin impact

- Result: “I love ****”

Create a Tester class called CensorTester and copy the initial code from here.
Take a screenshot of your code successfully running all of the test cases.

2.

```
/**
 * The censored words
 *
 * @author Froilan Zarate & Alan Xiao
 * @version 07-10-2022
 */
public class Censorship
{
    private String bannedWordOne, bannedWordTwo, bannedWordThree;
    /**
     * The constructor for censorship
     *
     * @param firstWord  the first censored word
     * @param secondWord  the second censored word
     * @param third  the third censored word
     */
    public Censorship(String firstWord, String secondWord, String thirdWord)
    {
          bannedWordOne = firstWord;
          bannedWordTwo = secondWord;
          bannedWordThree = thirdWord;
    }
    /**
     * Checks if the string contains a bad word
     *
     * @param str  the string
     * @return a true or false if there is a bad word
     */
    private boolean containsBadWord(String str)
    {
        return (str.toLowerCase().contains(bannedWordOne.toLowerCase()) || str.toLowerCase().contains(bannedWordTwo.toLowerCase()) || str.toLowerCase().contains(bannedWordThree.toLowerCase()));
    }
    /**
     * Checks if the string is a bad word
     *
     * @param str  the string
     * @return a true or false if the string is a bad word
     */
    public boolean isBadWord(String str)
    {
        return (str.toLowerCase().equals(bannedWordOne.toLowerCase()) || str.toLowerCase().equals(bannedWordTwo.toLowerCase()) || str.toLowerCase().equals(bannedWordThree.toLowerCase()));
    }
    /**
     * Replaces the string with censored bad words
     *
     * @param str  the string
     * @return the censored string
     */
    public String censorBadWords(String str)
    {
        str = str.replaceAll("(?i)" + bannedWordOne, "****");
        str = str.replaceAll("(?i)" + bannedWordTwo, "****");
        str = str.replaceAll("(?i)" + bannedWordThree, "****");
        return str;
    }
}
```

3. 

Traded with Group Jerison and Victoria

Objects:epic, lmao, lol

Tested words:

- I would like to have a lol and lmao competition
- Result: I would like to have a **** and **** competition.