1 Bear

public class Bear {
    public static int num = 0;
    public int myNum;
    public String name;

    public Bear (int n, String str) {
        num += 1;
        myNum = n;
        name = str;
    }

    public void printNum() {
        System.out.println(myNum);
    }

    public void printInfo(String str) {
        System.out.println("I like " + str);
    }

    public void printInfo(int d) {
        System.out.println("Number: " + d);
    }
}

Take a look at the class and answer the questions below. Suppose we instantiate the following two objects:

Bear bear1 = new Bear(4, "Oski");
Bear bear2 = new Bear(2, "Clark");

What is the output after executing the following snippet of code:

System.out.println(bear2.num);
bear2.num -= 1;
System.out.println(bear1.num);
bear2.myNum -= 1;
System.out.println(bear1.myNum);
bear1.printInfo(2);
bear1.printInfo("apples");
2  Box and Pointer Diagrams

Answer the following questions about the Avatar class.

```java
public class Avatar {
    public static String electricity; public String fluid;

    public Avatar(String str1, String str2) {
        Avatar.electricity = str1;
        this.fluid = str2;
    }

    public static void main(String[] args) {
        Avatar foo1 = new Avatar("one ", "two");
        Avatar foo2 = new Avatar("three ", "four");
        /* a */
        foo1.electricity = "I declare ";
        foo1.fluid = "a thumb war";
        /* b */
        foo2 = foo1;
        /* c */
    }
}
```

Draw the box-and-pointer diagrams of the states of the program during the lines with the comments a, b, and c in the main method before exiting.
a) \[ \text{foo} \rightarrow \text{Avatar} \quad \text{fluid L"two"} \quad \text{Avatar class attributes} \quad \text{electricity L"three"} \]

\[ \text{foo} \rightarrow \text{Avatar} \quad \text{fluid L"fan"} \quad \text{electricity L"fan"} \]

b) \[ \text{foo} \rightarrow \text{Avatar} \quad \text{fluid L"a thumb war"} \quad \text{Avatar class attributes} \quad \text{electricity L"I declare"} \]

\[ \text{foo} \rightarrow \text{Avatar} \quad \text{fluid L"fan"} \quad \text{electricity L"fan"} \]

c) \[ \text{foo} \rightarrow \text{Avatar} \quad \text{fluid L"a thumb war"} \quad \text{Avatar class attributes} \quad \text{electricity L"I declare"} \]

\[ \text{foo} \rightarrow \text{Avatar} \quad \text{fluid L"fan"} \quad \text{electricity L"fan"} \]
3 Java Practice

1. Write a function that sums up all the digits in an integer recursively.

   ```java
   public static int sumDigits (int x) {
       if (x <= 0) {
           return 0;
       }
       return x % 10 + sumDigits(x/10);
   }
   
   Solution:
   ```

   ```java
   public static int sumDigits(int x) {
       if (x <= 0) {
           return 0;
       }
       return x % 10 + sumDigits(x/10);
   }
   ```

2. Write a function that sums up all the digits in an integer iteratively. For example, `sumDigits(31415)` should return `3 + 1 + 4 + 1 + 5 = 14`.

   ```java
   public static int sumDigits (int x) {
   }
   
   Solution:
   ```

   ```java
   public static int sumDigits(int x) {
       int total = 0;
       for (int num = x; num > 0; num /= 10) {
           total += num % 10;
       }
       return total;
   }
   ```

   Note: The "while" loop version may be more familiar for students, but going over the "/=" notation in the for loop can be good practice!

   Alternate Solution:

   ```java
   public static int sumDigits(int x) {
       int total = 0;
       while (x > 0) {
   ```
total += x % 10;
x /= 10;
}
return total;
}
4  Do you Git it?

Recall the Git commands init, add, commit, status, log, show, clone, pull, and push. Which of those commands allows you to:

- tell Git to start tracking a file: **add**
- save a snapshot of the files being tracked: **commit**
- see what files have changed since your last commit: **status**
- see a list of previous commits: **log**
- create a local copy of a remote repository: **clone**
- send your locally tracked files to a remote repository: **push**