CodingBat code practice

Java String Equals and Loops

- Compare two Strings: a.equals(b)
- **Do not** use ‘ == ’
- Sadly ‘ = = ’ compiles, but does not work reliably .. a real trap
- In retrospect, an error in the design of Java

String Equals

Use the equals() method to check if 2 strings are the same. The equals() method is case-sensitive, meaning that the string "HELLO" is considered to be different from the string "hello". The = = operator does not work reliably with strings. Use = = to compare primitive values such as int and char. Unfortunately, it's easy to accidentally use = = to compare strings, but it will not work reliably. Remember: use equals() to compare strings. There is a variant of equals() called equalsIgnoreCase() that compares two strings, ignoring uppercase/lowercase differences.

```java
public class StringComparisonExample {
    public static void main(String[] args) {
        String a = "hello";
        String b = "there";

        if (a.equals("hello")) {
            System.out.println("a is equal to 'hello'");
        }

        // Avoid using == for comparing Strings
        if (a == "hello") {
            // This block might not execute as expected
            System.out.println("This block might not execute as expected");
        }

        if (a.equals(b)) {
            System.out.println("a is equal to b");
        } else {
            System.out.println("a is not equal to b");
        }

        if (b.equals("there")) {
            System.out.println("b is equal to 'there'");
        }

        if (b.equals("There")) {
            System.out.println("This block won't execute due to case difference");
        }

        if (b.equalsIgnoreCase("THERE")) {
            System.out.println("b is equal to 'THERE' (case-insensitive)");
        }
    }
}
```

Please note that I've added System.out.println statements to the code so that you can see the output when you run the program. This way, you can observe the behaviour of the different comparisons.

String For Loop

- Super-common string for-loop
- Loop to hit each index number once:
- 0, 1, 2, ... length-1
- for (int i = 0; i < str.length(); i++) { }

```
for (int i = 0; i < str.length(); i++) {
    // Code to be executed for each index i
}
```

- Strategy 1: straight use of standard loop -- great
- Strategy 2: standard loop + some variation