The following are five methods you can use when manipulating a String.

The first one is ***String.toUpperCase().*** This method turns all the characters in a string to uppercase and returns it as a new String. This can be useful when you want to compare two Strings, such as last names. Refer to the example below:

String nameOne = "rowland";

String nameTwo = "Rowland";

System.***out***.println(nameOne == nameTwo); //outpust false

System.***out***.println(nameOne.toUpperCase() == nameTwo.toUpperCase()); //outputs true

The second one is ***String.*** ***substring(int startIndex, int endIndex).*** This method returns a string within a string from startIndex (inclusive) to endIndex(exclusive). This can be useful when you want to grab a word out of a sequence. Refer to the example below:

String pattern = "John Doe did a back flip.";

String name = pattern.substring(0, "John Doe".length()); //will return John Doe

The third one is ***String.replaceAll(String expression, String replacement).*** This method searches for each pattern in a string and replaces it with a replacement argument and returns the newly created String. This can be useful if a company changes managers and wants the website to display the new manager’s name. Refer to the example below:

String siteInfo = "John Doe is the manager of our great loan department. John Doe has the right background of experience needed to provide loans at a low APY of 0 percent.";

System.***out***.println(siteInfo.replaceAll("John Doe", "Mary Q")); // Outputs : “Mary Q is the manager of our great loan department. Mary Q has the right background of experience needed to provide loans at a low APY of 0 percent.”

The fourth one is ***String.repeat(int n).*** This method returns a string repeated n amount of times. This can be useful when we want to alert a user n amount of times when a passwords is invalid. We can repeat an annoying message like “Invalid Information.” three times. Refer to the example below:

System.***out***.println("Invalid Information.".repeat(3)); //”Invalid Information. Invalid Information. Invalid Information.”

The fith one is ***String.indexOf(String str).*** This method returns the index of the first match of the given string argument. This can be useful when wanting to find a topic in a book or long paragraph. Refer to the example below:

String sequence = "If Barry Allen coded all day, he would be javastest man alive.";

int badPunIndex = sequence.indexOf("javastest");

Lists, Sets, and Maps are part of the Collection class and are interfaces of it. There are a few key differences between these. A List is ordered, but a Set is not. A Map is not ordered, but is a special type of Collection in that a value is accessed by a key. Keys in a map can not be duplicated, but values can. A Set’s values can not be duplicated either, but in a List they can.

Two implementation of a List is an ArrayList and a LinkedList. An LinkedList will be used over an ArrayList if we want to keep track of the last element inputted into the List. Two implementation of a Set is an HashSet and a LinkedHashSet. We can use a LinkedHashSet over a HashSet when we want to keep order of a Set.

List<String> list = new ArrayList<>();

Set<StringBuilder> set = new HashSet<>();

Map<String, String> map = new HashMap<>();