**7 kyu**

**ANTISTRING**

10084% of 37112 of116[Jomo Pipi](https://www.codewars.com/users/Jomo%20Pipi)

Java

* [TRAIN AGAIN](https://www.codewars.com/kata/antistring/train/java)
* [NEXT KATA](https://www.codewars.com/trainer/java)

Details

[Solutions](https://www.codewars.com/kata/antistring/solutions/java)

[Discourse (5)](https://www.codewars.com/kata/antistring/discuss/java)

* Add to Collection
* |
* Share this kata:

**BEHOLD THE ANTISTRING**



given an input string **str**, return a string such that

* **all letters change case**
* **all digits are replaced with it's compliment to 9 (0 => 9, 1 => 8...)**
* **each letter is replaced by it's compliment in the alphabet**

**(a => z, B => Y, c => x...)**

* **the result is reversed**

Example:

"abcWXY123" => "678bcdXYZ"

only strings with letters and numbers will be passed in.

**Check out my other kata!**

[Matrix Diagonal Sort OMG](https://www.codewars.com/kata/5ab1f8d38d28f67410000090)

[String -> N iterations -> String](https://www.codewars.com/kata/5ae43ed6252e666a6b0000a4)

[String -> X iterations -> String](https://www.codewars.com/kata/5ae64f28d2ee274164000118)

[ANTISTRING](https://www.codewars.com/kata/5ab349e01aaf060cd0000069)

[Array - squareUp b!](https://www.codewars.com/kata/5a8bcd980025e99381000099)

[Matrix - squareUp b!](https://www.codewars.com/kata/5a972f30ba1bb5a2590000a0)

[Infinitely Nested Radical Expressions](https://www.codewars.com/kata/5af2b240d2ee2764420000a2)

[pipi Numbers!](https://www.codewars.com/kata/5af27e3ed2ee278c2c0000e2)

<https://www.codewars.com/kata/antistring/java>

*/\**

*\* To change this license header, choose License Headers in Project Properties.*

*\* To change this template file, choose Tools | Templates*

*\* and open the template in the editor.*

*\*/*

**package** javaapplication1;

**import** java.util.HashMap;

***/\*\****

***\****

***\* @author Usuario***

***\*/***

**public** **class** JavaApplication1 {

**public** **static** String antiString(String str)

        {

*//Dictionary<char, char> diccio = new Dictionary<char, char>();*

            HashMap<Character, Character> diccio =

**new** HashMap();

**for** (**char** ch = 'a'; ch <= 'z'; ch++) diccio.put(ch, (**char**)('z' - ch + 'A'));

**for** (**char** ch = 'A'; ch <= 'Z'; ch++) diccio.put(ch, (**char**)('Z' - ch + 'a'));

**for** (**char** ch = '0'; ch <= '9'; ch++) diccio.put(ch, (**char**)('9' - ch + '0'));

            String concat = "";

**for**(**int** i =0; i<str.length(); i++)

            {

                concat = diccio.get( str.charAt(i)) + concat;

            }

**return** concat;

        }

**public** **static** **void** main(String[] args) {

*// TODO code application logic here*

       System.out.println( antiString("abcWXY123"));

    }

}

**class** JomoPipi{

**static** String antiString(**final** String str) {

**final** StringBuilder sb = **new** StringBuilder(str.length());

**for** (**int** i = str.length() - 1; i >= 0; i--) {

**char** ch = str.charAt(i);

**if** (Character.isUpperCase(ch)) {

                ch = (**char**) ('z' - Character.toLowerCase(ch) + 'a');

            } **else** **if** (Character.isLowerCase(ch)) {

                ch = (**char**) ('Z' - Character.toUpperCase(ch) + 'A');

            } **else** {

                ch = (**char**) ('9' - ch + '0');

            }

            sb.append(ch);

        }

**return** sb.toString();

    }