String Methods

length()

↳ returns number of characters

→ String s = "Sergiy";
System.out.println(s.length()); //6

public int indexOf (char c)
public int indexOf (char c, int i)

↳ Allows to locate c in a String

→ If c is in String, leftmost occurence index is returned, otherwise -1

→ Integer argument allows to start looking at i (0 to i-1 is ignored)

→ String s = "Toronto";
s.indexOf('T') ← returns 0
s.indexOf('o', 3) ← returns 3
s.indexOf('r', 3) ← returns -1

trim()

↳ Removes leading or trailing ws

→ Creates new String obj & returns reference

→ s = "   lots   of space   ";
s.trim(); // "lots of space"

valueOf()

↳ Values of any type can be converted to string. For objects we use toString(), for primitive data, valueOf()

→ String.valueOf(123) ← "123"
String.valueOf('a') ← "a"
String.valueOf(2.5); ← "2.5"

public char charAt (int i)

↳ String s = "Jasper";
s.charAt (0) ← returns 'J'
s.charAt (5) ← returns 'r'
s.charAt (6) ← exception thrown

public String substring (int start)
public String substring (int start, int pastEnd)

↳ extracts part of a string

→ [start, end)

→ String s = "Brian Auyeung";
s.substring(3) ← "an Auyeung"
s.substring(2,5) ← "ian "
s.substring(20) ← exception thrown

public String toUpperCase()
public String toLowerCase()
boolean equalsIgnoreCase(String other)

↳ String s = "Bart & Lisa";
s.toLowerCase(); ← "bart & lisa"
s.toUpperCase(); ← "BART & LISA"
s.equalsIgnoreCase("BART & Lisa"); ← true

Chaining Instance Methods: <object>.<method1>.<method2>

↳ s.toUpperCase.charAt(0);