Strings

	1st	2nd	3rd	4th			4th to Last	3rd to Last	2nd to Last	Last	END	str.substring(0, 3) → “abc” str.substring(1, 3) → “bc” str.substring(str.length()-2) → “yz” str.substring(str.length()-4, str.length()-1) → “wxy”
String str	**a**	**b**	**c**	**d**	**.**	**.**	**w**	**x**	**y**	**z**		
Index (position)	0	1	2	3	.	.	str.length()-4	str.length()-3	str.length()-2	str.length()-1	str.length()	

EVEN	**1st**	**2nd**	**3rd**	**4th**	**5th**	**6th**	**7th**	**8th**	**END**	**int mid = str.length() / 2; // 8/2 = 4** For even-length strings, there is **NO MIDDLE CHARACTER**. The character at index **mid** will be the first character in the right-half of the string. str.substring(mid-1, mid+1) gives the middle 2 characters.
String str	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**		
Index (position)	0	1	2	3 mid-1	4 mid	5 mid+1	6	7	str.length()	

ODD	**1st**	**2nd**	**3rd**	**4th**	**5th**	**6th**	**7th**	**8th**	**9th**	**END**	**int mid = str.length() / 2; // 9/2 = 4** For odd-length strings, mid gives the value of the index of the middle character. str.substring(mid-1, mid+2) gives the middle 3 characters.
String str	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**		
Index (position)	0	1	2	3 mid-1	4 mid	5 mid+1	6 mid+2	7	8	str.length()	

Strings

String Methods: These methods are called using **DOT** syntax using a String object, e.g.:
String + **DOT** + Method: word**.**length() “abcde”**.**substring(0,1)

Method	**Input Parameters**	**Description**	**Return Type**
length()	No Parameters	Returns the number of characters in the String.	**int** (NUMBER)
substring(int ***startIdx***, int ***stopIdx***)	***startIdx***: Index of the 1st character in the new string ***stopIdx***: Stopping index; the character at this index is NOT included. ***startIdx*** and ***stopIdx*** must satisfy the 3 conditions below: **(a) 0 <= *startIdx* <= str.length()** **(b) 0 <= *stopIdx* <= str.length()** **(c) *startIdx* <= *stopIdx***	Returns a ***NEW*** string containing (***stopIdx*-*startIdx***) # of characters. The 1st character of the new String is the character at position ***startIdx***. Characters are added up to, but do not include the character at index ***stopIdx***.	**String** (TEXT)
substring(***startIdx***)	***startIdx***: Index of the 1st character in the new string. ***startIdx*** must satisfy the following condition: **0 <= *startIdx* <= str.length()**	Returns a ***NEW*** string containing ***length()-startIdx*** # of characters. The new String includes all characters in the original string starting with the character at ***startIdx*** and continuing to the end.	**String** (TEXT)
equals(String ***strTest***)	***strTest***: The String used to compare the String object to.	Returns true if ***strTest*** matches the object String “abc” matches “abc” “abc” does **NOT** match “Abc”	**Boolean** (true/false)
equalsIgnoreCase(String ***strTest***)	***strTest***: The String used to compare the String object to. Matches strings regardless of uppercase or lowercase letters	Returns true if ***strTest*** matches the object String regardless of uppercase or lowercase, i.e. treats uppercase and lowercase letters as if they are the same. “abc” matches “abc” “abc” matches “ABC”	**Boolean** (true/false)
endsWith(String ***suffix***)	***suffix***: The String you want to match at the end of the String object.	Returns true if the String object ends with ***suffix***.	**Boolean** (true/false)
startsWith(String ***prefix***)	***prefix***: Does the object String start with ***prefix***?	Returns true if the String object begins with ***prefix***.	**Boolean** (true/false)
startsWith(String ***prefix***, int ***offsetIdx***)	***prefix***: Imagine that this creates a substring starting at position ***offsetIdx***. Does that substring start with ***prefix***?	Returns true if the imagined substring begins with ***prefix***.	**Boolean** (true/false)
isEmpty()	No Parameters	Returns true if the String object has no characters. This is the same thing as **return str.length() == 0;**	**Boolean** (true/false)