1 Strings

1.1 Creating a String Instance

- String str1 = "Hello"; // Using literal String
- str2 = new String("World"); // Using 'new' keyword

1.2 String Methods

- char **charAt**(int index)
  // Returns the character at a specified index. Throws IndexOutOfBoundsException.
- boolean **equalsIgnoreCase**(String str)
  // Compares the string values (ignoring case) and returns boolean value.
- boolean **equals**(Object obj)
  // Compares the string object and returns boolean value.
- int **compareTo**(String str)
  // Compares two strings based upon Unicode value of each character.
  - return 0 if both strings are equal.
  - return positive value if calling string is lexicographically greater than the parameterized string.
  - return negative value if parameterized string is lexicographically greater than the calling string.
  - if used to compare a string where $length \geq 1$ to an empty string, compareTo returns string length.
- int **compareToIgnoreCase**(String str)
  // Compares two strings based upon Unicode value of each character *ignoring case*.
- boolean **startsWith**(String prefix, int offset)
  // Checks whether a substring (starting at offset index) has the supplied prefix.
- boolean **startsWith**(String prefix)
  // Checks whether a string has the supplied prefix.
- boolean **endsWith**(String suffix)
  // Checks whether a string has the supplied suffix.
- int **hashCode**()
  // Returns the hash code of the string.
- int **indexOf**(int ch)
  // Returns the index of the first occurrence of the character 'ch' in the string.
- int **indexOf**(int ch, int fromIndex)
  // Same as indexOf but starts searching for 'ch' at the specified fromIndex.
- int **lastIndexOf**(int ch)
  // Returns the index of the last occurrence of 'ch' in the string
- int **lastIndexOf**(int ch, int fromIndex)
  // Same as above except beginning seach from 'fromIndex'.
- int **indexOf**(String str)
  // Returns the index (first letter) of the first occurrence of specified substring 'str'. If it doesn't exist, it returns -1.
- int **lastIndexOf**(String str)
  // Returns the index (first letter) of the last occurrence of specified substring 'str'. If it doesn't exist, it returns -1.
- String **substring**(int beginIndex)
  // Returns the substring starting at 'beginIndex' and ending at the end of the string.
- String **substring**(int beginIndex, int endIndex)
  // Returns the substring starting at 'beginIndex' and ending at 'endIndex'.
- String **concat**(String str)
  // Concatenates the specified string 'str' at the end of the calling string.
- String **replace**(char oldChar, char newChar)
  // Returns a new string after where each instance of 'oldChar' is replaced by an instance of 'newChar'.
- boolean **contains**(CharSequence s)
  // Checks if the calling string contains the specified sequence of char values. Throws NullPointerException if 's' is null.
- String **toUpperCase**(Locale locale)
  // Converts the string to upper upper-case using the rules defined by the specified locale.
- String **toUpperCase**()
  // Same as above and locale = Locale.getDefault().
- String **intern**()
  // Searches for the specified string in the memory pool and if found returns the reference to it. If the specified string is not found, the method allocates memory space to the specified string and assigns the reference to it. Java automatically interns string literals; this is useful when using 'new' keyword to make a string instance

```
String str1 = ``New String'';
String str2 = ``New String'';
String str3 = new String(``New String'');
String str4 = new String(``New String'').intern
System.out.println("Are str1 and str2 the
same: " + (str1 == str2)); // Returns true
System.out.println("Are str1 and str3 the
same: " + (str1 == str3)); // Returns false
System.out.println("Are str1 and str2 the
same: " + (str1 == str4)); // Returns true
System.out.println("Are str1 and str3 the
same: " + (str1 == str3.intern()); //
Returns true
```

- boolean **isEmpty**()
  // Method returns true if the given string has 0 length.
- public static String **join**()
  // Method joins the given strings using the specified delimiter and returns the concatenated Java String. CHECK AGAIN LATER.
- String **replaceFirst**(String regex, String replacement)
  // Replaces the first occurrence of substring that fits the given regular expression 'regex' with 'replacement'.
- String **replaceAll**(String regex, String replacement)
  // Replaces all occurrences of substrings that fit 'regex' with 'replacement'.
- String[] **split**(String regex, int limit)
  // Returns an array of substrings delimited by the given regular expression. 'limit' is a result threshold.
- String[] **split**(String regex)
  // Same as above, but without limit
- String **toLowerCase**(Locale locale)
  // Converts all of the characters in this String to lower case using the rules of the given Locale. If 'locale' is not specified, method uses Local.getDefault()
- public static String **format**(Locale l, String format, Object... args)
  // Returns a formatted string using the specified locale, format string, and arguments
- String **trim**()

// Returns a copy of the string with the leading and trailing whitespace omitted.

- char **toCharArray**()
  // Converts this string to a new character array.
- public static String **copyValueOf**(char[] data)
  // Returns a String that represents the character sequence in the array specified.
- public static String **copyValueOf**(char[] data, int offset, int count)
  // Returns a String, starting at the initial 'offset' index and continuing for 'count' characters, that represents the character sequence in the array specified.
- public void **getChars**(int srcBegin, int srcEnd, char[] dst, int dstBegin)
  // Copies characters from this string into the destination character array.
- public static String **valueOf**([argument])
  // Returns a string representation of argument, which can be boolean, char, int, long, float, double.
- boolean **contentEquals**(StringBuffer sb)
  // Argument can also be a CharSequence. Compares 'this' string to the specified StringBuffer (or CharSequence). The result is true if and only if this String represents the same sequence of characters as the specified StringBuffer (or CharSequence).
- boolean **regionMatches**(int srcoffset, String dest, int destoffset, int len)
  // Tests to see if two string regions are equal.
- boolean **regionMatches**(boolean ignoreCase, int srcoffset, String dest, int destoffset, int len)
  // Tests to see if two string regions are equal ignoring case.
- byte[] **getBytes**(String charsetName)
  // Encodes this String into a sequence of bytes using the named charset, storing the result into a new byte array. Can also take a Charset type argument.
- byte[] **getBytes**()
  // Encodes this String into a sequence of bytes using the platform's default charset, storing the result into a new byte array.
- int **length**()
  // Returns the length of 'this' string.
- boolean **matches**(String regex)
  // Tells whether or not this string matches the given regular expression.
- int **codePointAt**(int index)
  // Returns the character (Unicode code point) at the specified index.