

Immutable strings in Java

```
char aCharacter = 'a'; // char is an primitive type
String aString = "a"; // String is a class. aString is a
                      // reference to an object of type String
String emptyString = ""; // string with no characters, empty string
```

String objects are immutable (read only, can not be changed)

```
String str = "Hello World!"           is the same as
String str = new String("Hello World!");
```

str.length() gives the length of a string
str.charAt(i) gives the character at position i, e.g.

```
for(int i = 0; i < str.length(), ++i) {
    System.out.print(charAt(i));
}
```

```
String str2 = str + "13"; // str2 refers to string "Hello World!13"
```

```
"Hello".length() // length of string "Hello"
```

```
String str4 = "Ohio";
String str5 = str4.substring(0,3); // str5 refers to "Ohi"
str5 = str4.substring(1,3); // str5 refers to "hi"
"washington".substring(2,5) // gives the string "shi"
```

in substring(i,j) $j > i$, length of substring is always $j-i$ (ith position is included in the substring, jth position is not)

Testing strings for equality

```
String str1 = "OSU";
String str2 = "OSU"
```

str1 == str2 tests to see if str1 and str2 refer to the same object. This may or may not be true. This may be true if the compiler arranges it is so that equal strings share memory. This is implementation dependent.

str1.equals(str2) tests to determine if the characters in the two strings referred by str1 and str2 are the same. This is most often what we want to test for.

```
String str3 = "oSU";
```

```
str1.equalsIgnoreCase(str3); // returns true
"osu".equalsIgnoreCase(str3); // also returns true
```

Mutable strings in Java

StringBuffer objects are mutable (can be modified)

StringBuffer objects have a capacity, which is the length of the string it can store before it must allocate more space.

StringBuffer constructors:

```
public StringBuffer()           // initial value "", initial capacity 16
public StringBuffer(int length) // init value "", capacity length
public StringBuffer(String str) // initial value same as str,
                               // initial capacity str.length()+16
```

Some useful methods:

```
public StringBuffer append(String str)
public StringBuffer append(char[] chars)
```

```
public StringBuffer insert(int i, String str)
public StringBuffer insert(int i, char[] chars)
```

Example:

```
StringBuffer strBuffer = new StringBuffer("osu");
String insertString = "AAA";
strBuffer.insert(1, insertString); // strBuffer contains oAAAsu
```

```
public void setCharAt(int i, char character)
```

Example:

```
StringBuffer strBuff = new StringBuffer("osu");
char character = 'M';
strBuff.setCharAt(1, character); // strBuff contains the oMu
```